1980–1992 *Nucleic Acids Research, 2011, Vol. 39, No. 6* *Published online 10 November 2010*
doi:10.1093/nar/gkq1052

Protein multiple sequence alignment by hybrid bio-inspired algorithms

Vincenzo Cutello, Giuseppe Nicosia*, Mario Pavone and Igor Prizzi

Department of Mathematics and Computer Science, University of Catania, Viale A. Doria 6, 95125 Catania, Italy

Received March 10, 2010; Revised September 16, 2010; Accepted October 13, 2010

ABSTRACT

This article presents an immune inspired algorithm to tackle the Multiple Sequence Alignment (MSA) problem. MSA is one of the most important tasks in biological sequence analysis. Although this paper focuses on protein alignments, most of the discussion and methodology may also be applied to DNA alignments. The problem of finding the multiple alignment was investigated in the study by Bonizzoni and Vedova and Wang and Jiang, and proved to be a NP-hard (*non-deterministic polynomial-time* hard) problem. The presented algorithm, called Immunological Multiple Sequence Alignment Algorithm (IMSA), incorporates two new strategies to create the initial population and specific *ad hoc* mutation operators. It is based on the 'weighted sum of pairs' as objective function, to evaluate a given candidate alignment. IMSA was tested using both classical benchmarks of BAliBASE (versions 1.0, 2.0 and 3.0), and experimental results indicate that it is comparable with state-of-the-art multiple alignment algorithms, in terms of quality of alignments, weighted Sums-of-Pairs (SP) and Column Score (CS) values. The main novelty of IMSA is its ability to generate more than a single suboptimal alignment, for every MSA instance; this behaviour is due to the stochastic nature of the algorithm and of the populations evolved during the convergence process. This feature will help the decision maker to assess and select a biologically relevant multiple sequence alignment. Finally, the designed algorithm can be used as a local search procedure to properly explore promising alignments of the search space.

INTRODUCTION

The most effective method to discover structural or functional similarities among proteins is to compare multiple proteins of various 'phylogenetic' distances. Multiple Sequence Alignment (MSA) of proteins plays a central role in molecular biology, as it can unravel the constraints imposed by structure and function on the evolution of whole protein families (1). MSA has been used for building phylogenetic trees, for the identification of conserved motifs, to find diagnostic patterns families, and to predict secondary and tertiary structures of RNA and protein sequences (2).

In order to be able to align a set of biosequences, a reliable objective function is needed to quantify the performance of an alignment in terms of its biological plausibility through an analytical or computational function (3). The alignment quality is often the limiting factor in the analysis of biological sequences. Defining an appropriate and efficient objective function can remove this limitation, but this is still an active research field (3,4). A simple objective function used for this purpose is the 'weighted sums-of-pairs' (SP) with affine gap penalties (5). In this approach, each sequence receives a weight, which is proportional to the amount of independent information it contains (6), and the cost of the multiple alignment is equal to the sum of the costs of all the weighted pairwise substitutions. Since the knowledge about the structure of the search space for MSA is not enough to guide an effective search towards the best solution, several 'Evolutionary Algorithms' (EAs) have been developed to solve such a problem and, in general, computational biology problems (7,8). Evolutionary algorithms are applied to problems where exact methods and heuristics are not available, or where the size of the search space precludes an exhaustive search for the optimal solution. In this research work, we tackle MSA instances using a new Immunological Algorithm (IA), inspired by the

*To whom correspondence should be addressed. Tel: +39 095 738 3030; Fax: +30 095 330094; Email: nicosia@dmi.unict.it

The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

Clonal Selection Principle (9–11), called Immunological Multiple Sequence Alignment (IMSA). IMSA incorporates specific perturbation operators for MSA of amino acid sequences, and the results obtained show that the designed IA is comparable to the state-of-the-art MSA algorithms. It is important to highlight that IMSA is able to produce several optimal or suboptimal alignments, comparable to those obtained by other approaches. This is the crucial feature of EAs, in general, and of the algorithm, IMSA, used in this research work in particular.

The article is structured as follows: multiple sequence alignment of proteins section presents a short description on the features of multiple sequence alignment, including also the objective function used in this work; state-of-the-art methods for MSA section presents instead a brief review on the best methods for MSA problem; in IMSA section we describe the proposed IMSA, focusing on the description of its features and its operators; results section presents a large set of experiments, comparing IMSA with the state-of-the-art algorithms on all three versions of BAliBASE; final remarks section emphasizes the conclusions on the use of IMSA algorithm in multiple sequence alignment problems.

MULTIPLE SEQUENCE ALIGNMENTS OF PROTEINS

One of the most important and popular computational sequence analysis problems is to determine whether two, or more, biological sequences have common subsequences. However, two primary issues need to be faced to check the similarities between two or more sequences: the choice of an objective function to assess the biological alignment quality, and the design of an effective algorithm to optimize the given objective function.

The alignment quality is often the limiting factor in biological analyses of amino acid sequences; defining a proper objective function is a crucial task. Our research work focuses on the key issue of designing an efficient algorithm to find optimal and suboptimal alignments of protein sequences. Of course, the technique is also applicable to DNA alignments. The most popular objective function used to measure the biological alignment quality is the weighted SP with affine gap penalties (5), where each sequence receives a weight that is proportional to the amount of independent information it contains (6) and the cost of the multiple alignment is equal to the sum of the costs of all the weighted pairwise substitutions. Formally it is defined as:

$$\max_{\hat{S}}\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\mathrm{WSS}(\hat{S}_i,\hat{S}_j)+\sum_{i=1}^{n}\mathrm{AGPS}(\hat{S}_i)\right) \quad (1)$$

where n is the number of the sequences; AGPS is the 'affine gap penalty function' that is one of the most appropriate penalty score from a biological point of view; and WSS is the 'weighted symbol score', defined as

$$\mathrm{WSS}(\hat{S}_i,\hat{S}_j) = W_{ij}\sum_{k=1}^{\hat{\ell}} M(\hat{s}_{i,k},\hat{s}_{j,k}).$$

Sequence weights are determined by constructing a guide tree from known sequences.

For multiple protein sequence alignment, the weighted SP with affine gap penalties is a popular objective function included in many MSA packages. The problem of finding the multiple alignment was investigated in (12) and (13), and proved to be a NP-hard problem. Results presented in (13) were proven using a 'nonmetric scoring matrix' (zero distance between two identical residues), which is different from the actual scoring matrices used in multiple alignments. Moreover, in (12) the authors improved the previous investigation by using a fixed metric score matrix through a reduction from the 'Minimum Vertex Cover', a classical NP complete problem (14).

STATE-OF-THE-ART METHODS FOR MSA

Although the most popular method to solve MSA is based on 'Dynamic Programming' (DP) (15), which guarantees a mathematically optimal alignment, this method is limited to a small number of short sequences. Such a limitation is due to the growth of the problem space with the number of sequences and the length of the proteins. To overcome this problem, several heuristic approaches (16–18) based on different strategies have been developed to effectively deal with the complexity of the problem (3,19). All current methodologies of multiple alignment are heuristic and can be classified under three main categories: 'progressive alignments', 'exact algorithms' and 'iterative alignments'.

Progressive alignments

Progressive alignment is the most commonly used approach to multiple sequence alignment. This kind of methodology works by aligning the closest sequences first, and then the more distant ones are added. Although this approach has the advantage of being simple and very fast, it does not guarantee any level of optimization. Therefore, the main drawback of this approach is that once a sequence has been aligned it cannot be modified, even if it produces possible conflicts with subsequently added sequences. Alignment programs based on this approach are MULTALIGN (20), PILEUP (21), CLUSTALX (22), CLUSTALW (23), T-COFFEE (24). Their strategy is to align sequences in a progressive manner, by using either a consistency-based or a SP objective function in order to minimize possible errors. In contrast to the previous approach, PIMA (25), which is also a progressive alignment method, uses local dynamic programming to align only the most conserved motifs. In the default setting, it makes use of two alignment methods, 'maximum linkage' (ML_PIMA) and 'sequential branching' (SB_PIMA), to decide the order of alignments. Sequence and Secondary-structure Profiles Enhanced Multiple alignment (SPEM) (26) combines a

sequence-based method with a consistency-based refinement for pairwise alignment, and is also a progressive algorithm for multiple alignment. PROBCONS (27) is a practical tool for progressive protein multiple sequence alignment that is based on 'probabilistic consistency', which is a novel scoring function for measuring alignment quality. It also incorporates an iterative refinement process.

Exact algorithms

Exact algorithms were developed to align multiple sequences simultaneously (28). They are high-quality heuristics able to produce alignments near to optimal ones, but they are limited to handle a small number of sequences. Thus, high memory requirement, high computational effort and limitation on the number of sequences limit their usage. A new divide and conquer algorithm (29) extending their capabilities was developed.

Iterative alignments

Iterative alignment methods depend on algorithms able to produce an alignment and to refine it through a series of iterations until no further improvements can be made. They are based on the idea that the solution to a given problem can be computed by modifying an already existing 'suboptimal solution'. DIALIGN (30,31), a consistency-based algorithm, attempts to use local information in order to guide a global alignment, i.e. to construct multiple alignments based on segment-to-segment comparisons—such segments are incorporated into a multiple alignment by using an iterative procedure. PRRP (32) optimizes a progressive global alignment by iteratively dividing the sequences into two groups, which are realigned by using a global group-to-group alignment algorithm. HMMT (33) a Hidden Markov Model (HMM) using simulated annealing and dynamic programming for correctly sampling suboptimal multiple alignments is better able to find global optima than other HMM methods. Multiple sequence comparison by log-expectation (MUSCLE) (34) is based on similar strategies as those used by PRRP. Sequence Alignment by Genetic Algorithm (SAGA) (35) is a genetic algorithm based on the Consistency Objective Function For alignmEnt Evaluation COFFEE objective function (36). The approach described in SAGA has received considerable interest in the evolutionary computation community. PRofile ALIgNEment (PRALINE) (37) begins with a pre-processing of the sequences to align. As of today, it also provides a choice of seven different secondary structure prediction programs that can be used either individually or in combination as a consensus for integrating structural information into the alignment process.

In general, EAS tend to be suitable tools for MSA (8,38) and can be used to effectively search large solution spaces. However, they spend a lot of time in gradually improving potential solutions before reaching a solution that is comparable to those obtained by deterministic methodologies (39). This is due to a random initialization of the candidate alignments.

IMSA: IMMUNOLOGICAL MULTIPLE SEQUENCE ALIGNMENT

Clonal Selection Algorithms (CSA) are a special class of IA, which are inspired by the clonal selection principle (10,11) to produce effective mechanisms for search and optimization (40–42). The proposed algorithm, called IMSA, is population-based, where each individual of the population is a 'candidate solution' belonging to the fitness landscape of a given MSA instance. This work presents an extended and more robust version of IMSA than those proposed in (7,43). The algorithm has been tested on a larger test case (BAliBASE versions 1.0, 2.0 and 3.0), and several metrics have been used to assess the quality both of alignments and of comparisons.

IMSA incorporates two different strategies to create the initial population, as well as new hypermutation operators, which are specific operators for solving protein MSA that insert or remove gaps in the given sequences. Gap columns, which have been matched, are moved to the end of the sequence. The remaining elements (i.e. amino acids) and existing gaps are shifted into the freed space. Like the classical IAS, IMSA considers antigens (Ags) and B cells. An Ag is a given MSA instance, i.e. the protein sequences to align, while B cells are a population, a multi-set, of alignments that have solved (or approximated) the initial problem (44–46).

In tackling the MSA, Ags and B cells are represented by a sequence matrix. Let $\Sigma = \{A,R,N,D,C,E,Q,G,H,I,L,K,M,F,P,S,T,W,Y,V\}$ be the twenty amino acid alphabet, and let $S = \{S_1,S_2,\ldots,S_n\}$ be the set of $n \geq 2$ sequences (strings), with respective lengths $\ell_1,\ell_2,\ldots,\ell_n$, such that $S_i \in \Sigma^*$. An Ag, hence, is represented by a matrix of n rows and $max\{\ell_1,\ldots,\ell_n\}$ columns (see the upper matrix of Figure 1). Each B cell is represented by an $(n \times \ell)$ binary matrix (see the middle matrix of Figure 1), where $\ell = (\frac{3}{2} \times max\{\ell_1,\ldots,\ell_n\})$, with $\frac{3}{2}$ a fixed parameter, ℓ is a maximum string length in order to properly manage a given problem instance. In such a matrix, an entry $s'_{ij} = 1$ indicates that the corresponding amino acid of the sequence S_i will be locate in the j-th position; otherwise, if $s'_{ij} = 0$, a gap will be placed in the j-th position of S'_i. The overall representation of genotype and phenotype is showed in Figure 1.

Initial population strategies

To create the initial population of d candidate alignments, we used two different strategies. The first strategy is based on the use of random 'offsets' to shift the initial sequences and it is called 'random_initialization'. Such a model works by randomly choosing an offset in the range $[0,(\ell-\ell_i)]$ with uniform distribution, and then by shifting the sequence S_i offset towards the right side of the row i of the current B cell.

Figure 2 shows an example of the scheme used to initialize the population by using random 'offsets'. Plot (b) represents how such process works using different 'offset' values, considering the initial multiple sequence alignment shown in plot (a). The second way to initialize the population is to use the CLUSTALW algorithm (23).

However, to increase the diversity of the initial population, we have used both strategies together; a percentage of the initial alignments were generated by CLUSTALW, while the remaining ones were determined by the creation of random offsets. We called this new method 'CLUSTALW-seeding'. All results shown in this article were obtained using 80% of the initial population generated by CLUSTALW, and the remaining 20% by 'random_initialization' (i.e. using the random 'offsets'). We have used 'CLUSTALW-seeding' to initialize the population of alignments, to avoid that the algorithm could be trapped in local optima during the early phases of the convergence process.

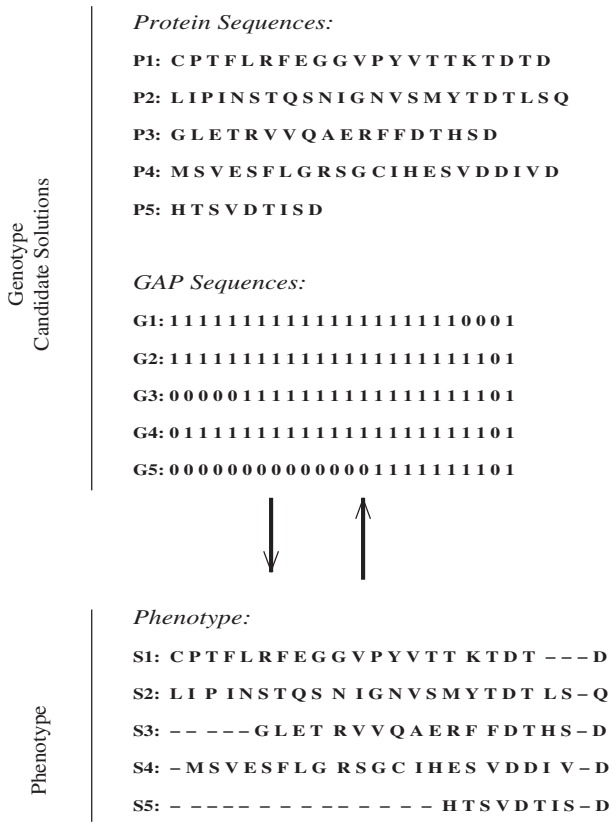

Figure 1. How the genotype and the phenotype are represented in IMSA.

Cloning and hypermutation operators

The clonal expansion process in IMSA was represented by the classical 'static cloning operator', which clones each B cell *dup* times, thus producing an intermediate population $P^{(clo)}$ of $N_c = d \times dup$ B cells, where d is the population size. The basic mutation processes that are considered in pairwise alignment and multiple sequence alignments are as follows: 'substitutions' that change sequences of amino acids, as well as 'insertions' and 'deletions', which respectively, add and remove amino-acids and/or gaps (19). In a first version of the algorithm, the classical hypermutation and hypermacromutation operators (47–49) were used; the first operator flips a bit, by using a number of mutations that is inversely proportional to the fitness function value, whereas hypermacromutation simply swaps two randomly chosen subsequences. However, the first experiments produced non-optimal alignments leading to frequent premature convergence to a local optimum during the convergence process. Therefore, we developed two new hypermutation operators, specifically for multiple sequence alignments that insert or remove gaps in the sequences. Such operators are the 'GAP operator' and the 'BlockShuffling operator'. Both of them act on the cloned B cells ($P^{(clo)}$) and generate two new populations, $P^{(gap)}$ and $P^{(block)}$, respectively.

GAP operator. The 'GAP operator' is based on two procedures: one inserts adjacent sequences of gaps (*InsGap*) while the other one removes them (*RemGap*). Initially, the GAP operator chooses which procedure to apply by using a random uniform distribution, i.e. it is randomly decided whether a number of adjacent gaps is to be inserted into the sequences or removed. Then a number k, in the range $[1,\theta]$, of (adjacent) gaps is randomly chosen, where θ represents a percentage of the length of the alignments (ℓ). Results shown in this article were obtained by setting $\theta = 2\% \cdot \ell$.

The INSGAP PROCEDURE can be summarized by the following steps. First, split the n sequences in z groups; from experimental results, $z = 2$ is the best setting for the performance of IMSA. Hence, we can rephrase this step as follows: randomly choose a value $m \in [1,n]$, and split the n sequences into two groups, respectively, from sequence 1 to m, and from $(m+1)$ to n. Second, randomly choose two integer values x and y, in such a way that k adjacent gaps are inserted beginning from column x for the first group, and from column y for the second one.

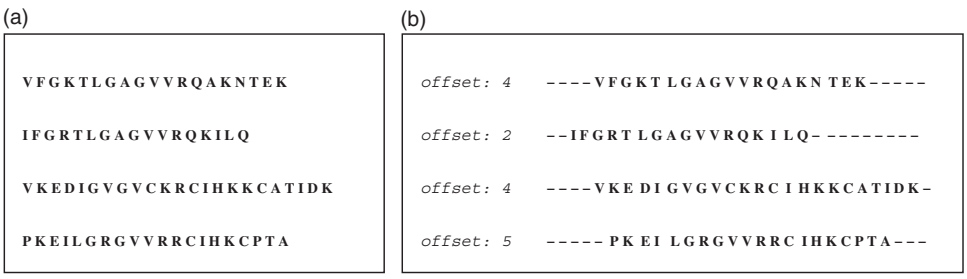

Figure 2. Initialize the population by 'offsets'.

Third, choose a random shift direction D for the subsequence, either left or right. Finally, insert the k adjacent gaps in the respective positions for each sequence, and then shift the subsequence to the D direction. During the shifting phase, it is possible to miss $n \geq 0$ bits with value 1 (it is similar to the left and right shift operators in the C programming language family); in this case, InsGap will select n bits with value 0, different from the k gaps inserted, and they will be flipped to 1, thereby rebuilding the correct sequence.

The RemGap procedure simply removes k adjacent gaps and moves the subsequences towards a randomly chosen direction either left or right. Figure 3 shows how the 'GAP operator' works. In particular, it shows the InsGap [plot (a) of Figure 3] and RemGap [plot (b) of Figure 3] procedures. They, respectively have the purpose to insert and remove adjacent gaps into the proposed alignment. In plot (a), an example of the InsGap procedure is shown using $k = 3$, $m = 2$ and the right shift direction.

BlockShuffling operator. The BlockShuffling operator is based on the block definition, and it moves aligned blocks to the left or to the right; a block is selected in each alignment starting from a random point in a sequence. IMSA includes three different approaches:

(1) *BlockMove* moves whole blocks, either to the left or to the right;

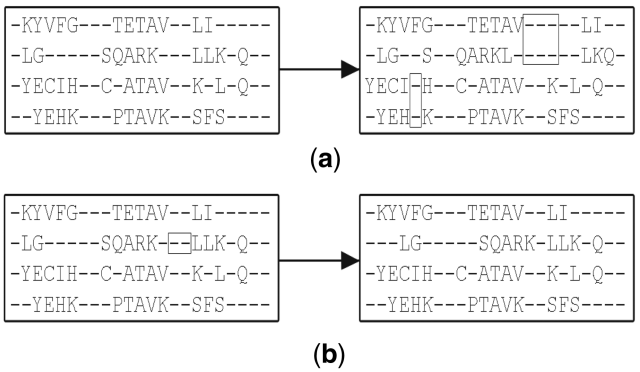

Figure 3. The GAP operator has the purpose to insert, by the *InsGap* procedure (**a**), or to remove, by the 'RemGap' procedure (**b**), adjacent gaps into the proposed alignment.

(2) *BlockSplitHor* divides the blocks into two parts, upper and lower, and shifts only one part, chosen randomly;
(3) *BlockSplitVer* randomly chooses a column in the block, divides the block into two sides (left and right) and shifts only one side, randomly chosen as well.

Figure 4 summarizes the three operators: the upper plot shows the *BlockMove* operator; the middle plot depicts how *BlockSplitHor* works, by choosing the 4th row to divide the block into two parts; and the lower plot shows the *BlockSplitVer* operator performing a right shift.

After the two hypermutation operators are used, IMSA moves the only-gap columns (columns made of gaps only) to the right end side of the matrix, with the Strip_Gaps($P^{(*)}$) function. This function is always applied before the fitness function is evaluated. Figure 5 shows an example of how Strip_Gaps($P^{(*)}$) function works.

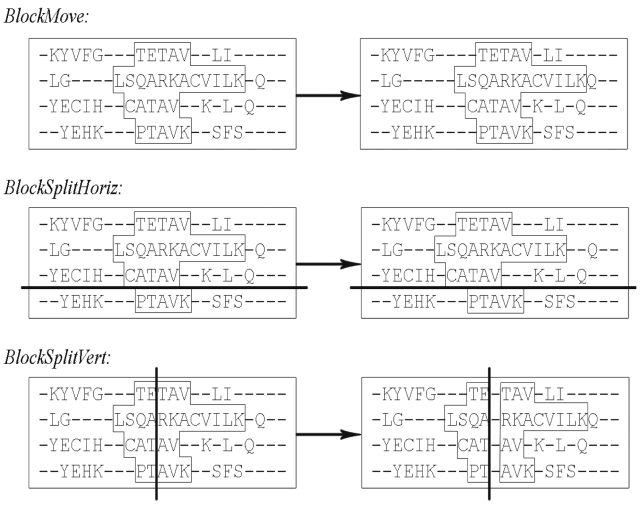

Figure 4. The *BlockShuffling operator* has the purpose to shift blocks of amino acids or gaps. The upper plot shows the *BlockMove* operator; the middle plot depicts how *BlockSplitHor* works, by choosing the 4th row to divide the block into two parts; the lower plot shows the *BlockSplitVer* operator performing a right shift.

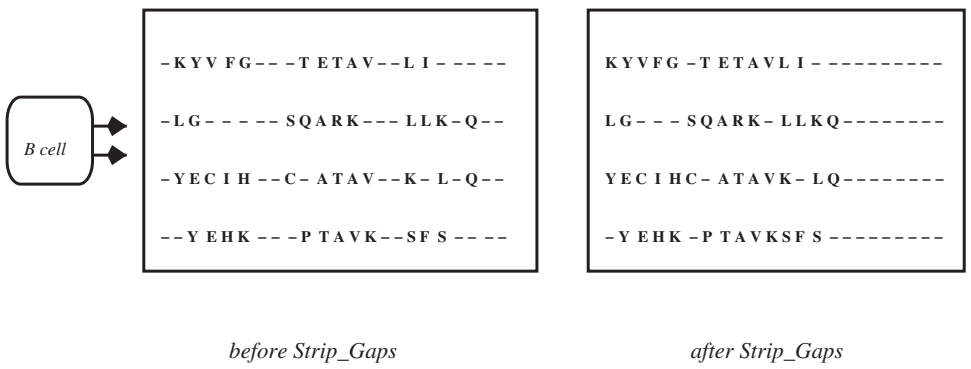

Figure 5. The *Strip_Gaps* operator moves only-gap columns to the right end side of the matrix.

Table 1. Pseudo-code of IMSA

```
IMSA (d, dup, τ_B, T_max)
t ← 0;
FFE ← 0;
N_c ← d × dup;
P^(t) ← Initialize_Population(d);
Strip_Gaps(P^(t));
Evaluate(P^(t));
FFE ← FFE + d;
while (FFE < T_max)do
    P^(clo) ← Cloning (P^(t), dup);
    P^(gap) ← Gap_operators (P^(clo));
    Strip_Gaps(P^(gap));
    Evaluate(P^(gap));
    FFE ← FFE + N_c;
    P^(block) ← BlockShuffling_operators (P^(gap));
    Compute_Weights();
    Normalize_Weights();
    Strip_Gaps(P^(block));
    Evaluate(P^(block));
    FFE ← FFE + N_c;
    (P_a^(t), P_a^(gap), P_a^(block)) ← Elitist-Aging(P^(t), P^(gap), P^(block), τ_B);
    P^(t+1) ← (μ+λ)-Selection(P_a^(t), P_a^(gap), P_a^(block));
    t ← t + 1;
end_while
```

Aging and (μ + λ)-selection operators

The aging operator eliminates the old B cells in the populations $P^{(t)}$, $P^{(gap)}$ and $P^{(block)}$, with the goal to maintain high diversity in order to avoid premature convergence. The number of generations that a B cell can remain into the populations is determined by parameter τ_B; when a B cell reaches $\tau_B + 1$ generations, it is erased from the current population, even if it is a good candidate solution (Static-Aging). The only exception is made for the best B cell present in the current population; we call this scheme Elitist-Static-Aging.

A new population $P^{(t+1)}$ of d B cells is obtained by selecting the best survivors from the aging operator, using the $(\mu+\lambda)$-selection operator (with $\mu = d$ and $\lambda = 2N_c$). The selection operator reduces an offspring B cell population of size $\lambda \geq \mu$ to a new parent population of size μ; it guarantees monotonicity in the evolution dynamics.

Table 1 (see Section IMSA: Immunological Multiple Sequence Alignment) shows the pseudo-code of the IMSA algorithm, where the function *Evaluate(P)* computes the SP objective function [see Equation (1) in Section] of each B cell in the population P [i.e. the proposed alignment quality, using Equation (1)]. For our research purpose the used objective function was the 'weighted' SP with affine gap penalties (5). The functions COMPUTE_WEIGHTS() and NORMALIZE_WEIGHTS(), respectively, compute and normalize the weights of the sequences by using a rooted tree, which is used for the evaluation of the objective function.

Finally, with $T_{\max}$ we indicate the maximum number of fitness function evaluations that we have used as termination criterion of the optimization process. The variable Fitness Function Evaluations (FFE) computes the number of the objective function evaluations, after immunological operators are applied.

RESULTS

To evaluate the biological alignment quality produced by IMSA, we have used the classical benchmark BALIBASE versions 1.0, 2.0 and 3.0. Benchmark Alignment dataBASE (BALIBASE) is a database that has been developed to evaluate and to compare multiple alignment programs containing high-quality (manually refined) multiple sequence alignments. BALIBASE version 1.0 (50) contains 141 reference alignments, and is divided into five hierarchical reference sets, containing 12 representative alignments. For each alignment the 'core blocks' are defined; they are the regions that can be reliably aligned and represent 58% of residues in the alignments. The remaining 42% are in ambiguous regions which cannot be reliably aligned. Reference 1 contains alignments of equidistant sequences with similar length; reference 2 contains alignments of a family (closely related sequences with >25% identity) and three orphan sequences with 20% identity; reference 3 consists of up to four families with <25% identity between any two sequences from different families; and references 4 and 5 contain sequences with large N/C-terminal extensions or internal insertions. For an extensive explanation of all references please refer to (4). The second version, BALIBASE v.2.0 (51), includes all alignments present in the first version, where all alignments have been verified and 'hand-constructed' from the literature (http://bips.u-strasbg.fr/fr/Products/Databases/BAliBASE2/). Moreover, test alignments are scored with respect to BALIBASE core blocks, regions for which reliable alignments are known to exist (27). Finally, the third version of BALIBASE (52) contains 218 alignments, and it is organized in the same way as the version 2.0, but with a larger sequence collections that contain more outlier sequences (http://www-bio3d-igbmc.u-strasbg.fr/balibase/). One interesting and favourable feature of IMSA is its ability to produce several optimal or suboptimal alignments. In this way, IMSA gives to biologists more tools to better study and understand the protein sequences.

Figure 6 shows two different alignments produced by IMSA for the BALIBASE instances on Reference 1: *1ad2*, in the left plot, and *1aym3*, in the right one. The left plot in Figure 6 shows two different alignments with the same SP and CS scores, while in the right plot two alignments are shown with different SP and CS scores. CS represents the Column Score, which is defined as the number of correctly aligned columns on the generated alignments, divided by the total number of aligned columns in the core blocks of the reference alignment. In both plots, the different alignment subsequences are highlighted in grey.

The left plot in Figure 7 shows two different alignments produced by IMSA for the *1hfh* instance of Reference 1, with different SP and CS score values, while the right plot shows three alignments with the same SP and CS score for the *2mhr* instance of Reference 1. We highlight the difference between the alignments in grey.

Finally, Figures 8 and 9 show different alignments produced by IMSA on instances of Reference 3 (*1uky*) and Reference 5 (*1qpg*).

Figure 6. Optimal and suboptimal alignments produced by IMSA for the *1ad2* (left) and *1aym3* (right) instances of Reference 1 (V2). The SP and CS scores show the difference between the two alignments (it is highlighted in grey).

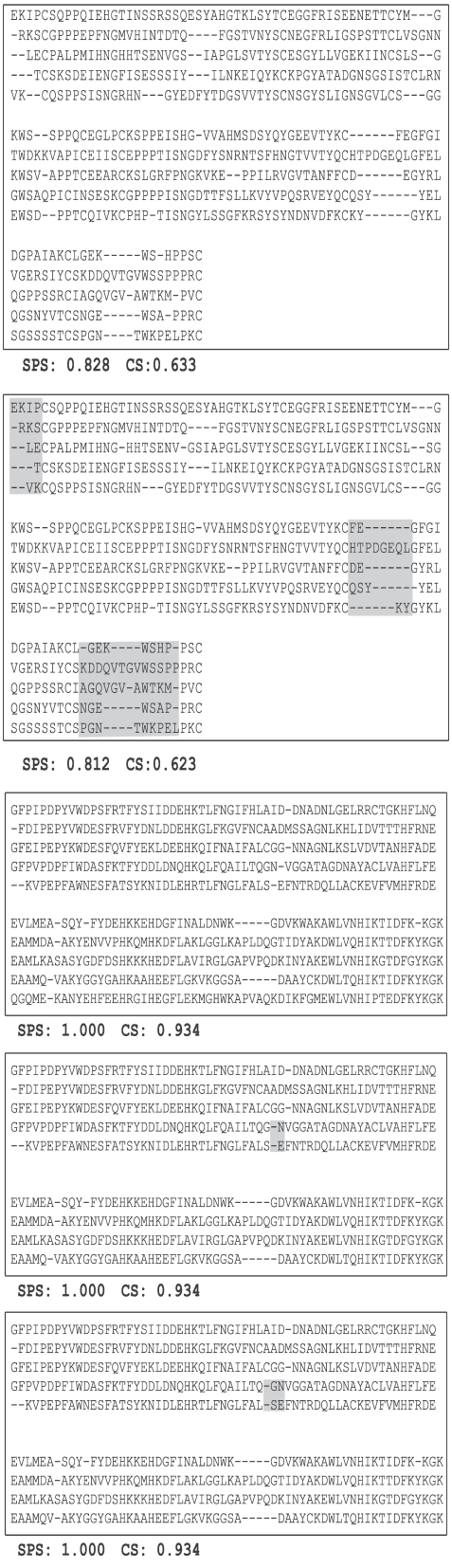

Figure 7. Optimal and suboptimal alignments produced by IMSA for *1hfh* (left) and *2mhr* (right) instances of Reference 1 (V2 and V3, respectively). The left plot shows two different alignments for the *1hfh* instance with different SP and CS score values, while the right plot shows three different optimal alignments for the *2mhr* instance. The differences between the alignments are highlighted in grey.

Figure 8. Optimal and suboptimal alignments produced by IMSA for the *1uky* instance for Reference 3 (medium). We show the SP and CS scores. The difference between the two alignments is highlighted in grey.

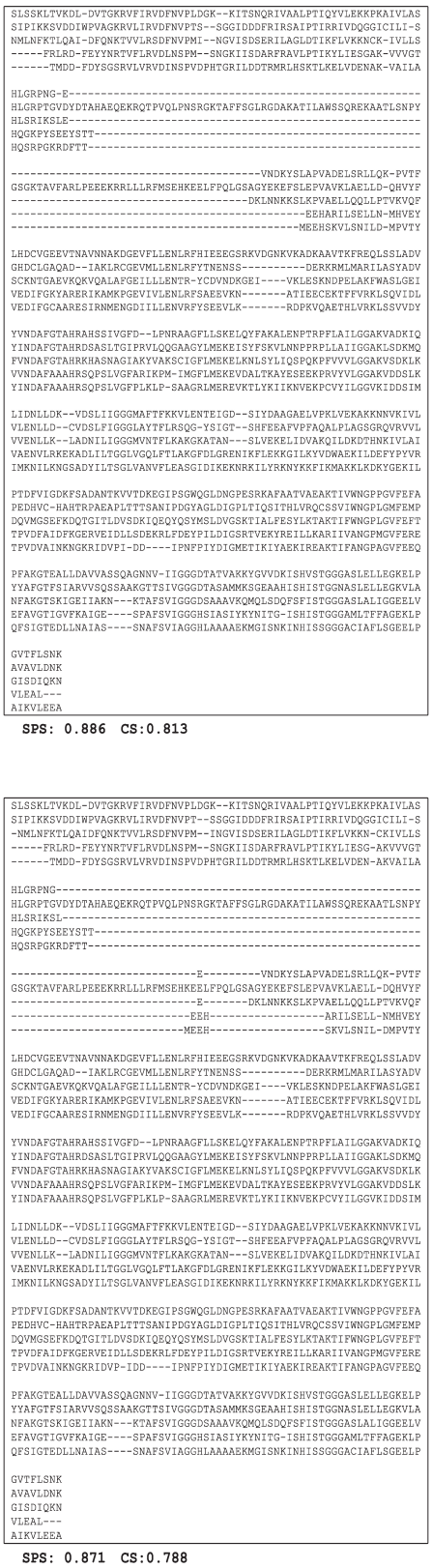

Figure 9. Optimal and suboptimal alignments for the *1qpg* instance of Reference 5.

These figures highlight the capability of IMSA to produce optimal and suboptimal alignments. Thanks to this ability, more tools are available to the biologists to better understand and study the proteins evolution process.

The results shown in all experiments were obtained by using the following experimental protocol: population size $d = 10$, cloning parameter $dup = 1$, age parameter $\tau_B = 33$, maximum number of objective function evaluations $T_{\max} = 2 \times 10^5$ and 50 independent runs. The parameter values have been selected inspecting the literature on the clonal selection algorithms (45,49,53). Moreover, we used the following substitution matrices:

- BLOSUM45 for Ref1v1 and Ref 3, with $GOP = 14, GEP = 2$;
- BLOSUM62 for Ref1v2, Ref 2, Ref 4 and Ref 5, with $GOP = 11, GEP = 1$;
- BLOSUM80 for Ref1v3, with $GOP = 10, GEP = 1$.

Table 2 shows the average SP score obtained by the described alignment tools on every instance set of BAliBASE v.1.0. As it can be seen in this table, IMSA performs well on the Reference 2 and Reference 3 sets. The values obtained help to raise the overall score, which is higher compared with the results published by the Bioinformatics platform of Strasbourg (http://bips.u-strasbg.fr/en/presentation.php). In Table 3, we show the ability of IMSA to improve and to refine the best initial alignment produced by *CLUSTALW-seeding* on the BAliBASE v.1.0 benchmark. In all references, IMSA improves the initial alignments, producing an overall average SP score 10.5 times better than the initial ones. By this feature, IMSA yields an effective refinement methodology.

In Table 4, we show the average SP and CS values obtained by the tools on every group of instances belonging to the BAliBASE v.2.0 database. The table also represents the accuracies of the produced alignments. The values used in Table 4 are drawn from data reported in (26). IMSA obtains comparable values of SP score on Reference 1, Reference 2 and Reference 5—despite the fact that the value obtained on Reference 3 is the fourth best value. This table also shows that future efforts should focus on improving the CS metric. The last column of the Table 4 (see Results Section) indicates the average number of the improved alignments (NIA), with respect to the initial population produced by *CLUSTALW-seeding*, which was described in Section.

To further evaluate the real performance of the proposed IMSA, Table 5 reports how many best alignments are produced by IMSA, with respect to *CLUSTALW-seeding*. We present the average SP and CS values obtained on each reference belonging to the BAliBASE v.2.0 database. Even in this version of

Table 2. SP values given by several methods on the BAliBASE v.1.0 benchmark (http://bips.u-strasbg.fr/fr/Products/Databases/BAliBASE/) for multiple sequence alignment

Aligner	Ref. 1 (82)	Ref. 2 (23)	Ref. 3 (12)	Ref. 4 (12)	Ref. 5 (12)	Overall (141)
DIALIGN (30)	77.7	38.4	28.8	**85.2**	**83.6**	62.7
CLUSTALX (22)	85.3	58.3	40.8	36.0	70.6	58.2
PILEUP8 (21)	82.2	42.8	33.3	59.1	63.8	56.2
ML_PIMA (25)	80.1	37.1	34.0	70.4	57.2	55.7
PRRP (34)	**86.6**	54.0	48.7	13.4	70.0	54.5
SAGA (35)	70.3	58.6	46.2	28.8	64.1	53.6
SB_PIMA (25)	81.1	37.9	24.4	72.6	50.7	53.3
MULTALIGN (20)	82.3	51.6	27.6	29.2	62.7	50.6
IMSA	80.7	**88.6**	**77.4**	70.2	82.0	(**79.7** ± 5.6) (78.47,80.92)

For IMSA we report mean and standard deviation ($\mu \pm \sigma$), and confidence interval, about 95% of the data are within 1.96 SD of the mean. Best results are in boldface.

Table 3. Performance of IMSA with respect to the initial population $P^{(t=0)}$ produced by *CLUSTALW-seeding*, on BAliBASE v.1.0 benchmark

Aligner	Ref. 1 (82)	Ref. 2 (23)	Ref. 3 (12)	Ref. 4 (12)	Ref. 5 (12)	Overall (141)
CLUSTALW-seeding	77.1	63.1	63.7	65.7	78.4	69.2
IMSA	80.7	88.6	77.4	70.2	82.0	79.7
Improvement	+3.6	+25.5	+13.7	+4.5	+3.6	+**10.5**

Best results are in boldface.

Table 4. Alignment accuracies given by several methods on the BAliBASE v.2.0 benchmark (http://bips.u-strasbg.fr/fr/Products/Databases/BAliBASE2/) for multiple sequence alignment (26)

Aligner	Ref. 1 (82)		Ref. 2 (23)		Ref. 3 (12)		Ref. 4 (12)		Ref. 5 (12)		Overall (141)		NIA
	SP	CS	SP	CS	SP	CS	SP	CS	SP	CS	SP	CS	
SPEM (26)	**90.8**	83.9	93.4	57.3	81.4	56.9	**97.4**	**90.8**	97.4	**92.3**	**91.5**	78.6	1
MUSCLE (34)	90.3	**84.7**	64.4	60.9	82.2	61.9	91.8	74.8	**98.1**	92.1	91.0	**78.7**	1
ProbCons (27)	90.0	83.9	**94.0**	**62.6**	**82.3**	**63.1**	90.9	73.6	**98.1**	91.7	90.8	78.4	1
T-Coffee (24)	86.8	80.0	93.9	58.5	76.7	54.8	92.1	76.8	94.6	86.1	88.2	74.6	1
PRALINE (37)	90.4	83.9	**94.0**	61.0	76.4	55.8	79.9	53.9	81.8	68.6	88.2	73.9	1
ClustalW (23)	85.8	78.3	93.3	59.3	72.3	48.1	83.4	62.3	85.8	63.4	85.7	70.0	1
IMSA	83.4	65.3	92.1	41.3	78.6	36.2	73.0	31.9	83.6	56.9	(82.1 ± 9.2) (79.55,84.64)	(46.3 ± 6.9) (44.38,48.21)	52

BAliBASE v.2.0 (51) includes all alignments present in the first version, where all alignments have been verified and 'hand-constructed' from the literature. Test alignments are scored with respect to BAliBASE core blocks, regions for which reliable alignments are known to exist (27). For IMSA, we report mean and standard deviation ($\mu \pm \sigma$), and confidence interval, about 95% of the data are within 1.96 SD of the mean. Best results are in boldface.

BALIBASE, IMSA produces better alignments than the initial ones, thus again showing its refinement ability.

For sake of completeness, we tested IMSA with two immunological aligners, *ClonAlign* and *AIS*. Tables 6 and 7 show the comparisons. In both cases, IMSA outperforms *ClonAlign* and *AIS*.

Table 5. Performance of IMSA with respect to the initial population $P^{(t=0)}$ produced by *CLUSTALW-seeding*, on BALIBASE v.2.0 benchmark BALIBASE v.2.0 (51) includes all alignments present in the first version, where all alignments have been verified and 'hand-constructed' from the literature (27)

Aligner	*CLUSTALW-seeding*		IMSA		Improvement	
	SP	CS	SP	CS	SP	CS
Ref. 1 (82)	77.1	64.9	83.4	65.3	+6.3	+0.4
Ref. 2 (23)	85.5	40.7	92.1	41.3	+6.6	+0.6
Ref. 3 (12)	68.3	34.9	78.6	36.2	+10.3	+1.3
Ref. 4 (12)	64.1	29.9	73.0	31.9	+8.9	+2.0
Ref. 5 (12)	73.8	51.4	83.6	56.9	+9.8	+5.5
Overall (141)	73.7	44.3	82.1	46.3	**+8.4**	+2

Best results are in boldface.

Table 6. IMSA versus AIS (57), each entry reports the SP value

Instance	Sequences	BW (58)	AIS (58)	IMSA
1aboA	5	0.622	0.646	**0.759**
451c	5	0.321	0.538	**0.773**
9rnt	5	0.783	0.804	**0.954**
kinase	5	0.308	0.399	**0.644**
2cba	5	0.653	**0.761**	0.754
1ppn	5	0.605	0.623	**0.987**
2myr	4	0.236	**0.385**	0.285
1eft	4	0.728	0.739	**0.880**
1taq	5	0.747	0.817	**0.946**
1ubi	17	0.267	0.393	**0.897**
kinase	18	0.186	0.270	**0.905**
1idy	27	0.295	0.346	**0.854**
Average		0.479	0.560	**0.810**

Best results are in boldface.

In Table 8 is shown the comparison among IMSA and some of the most popular alignment algorithms, as PROBCONS (27), PCMA (54), MUSCLE (34), CLUSTALW (23) and COBALT (55). These experiments have been done on BALIBASE version 3.0 (52), which containing 218 alignments, and it is organized in the same way as the version 2.0, but with larger sequence collections that contain more outlier sequences. Tables 8 shows 'quality assessment score (Q-score)', that is an average over all datasets in the benchmark, and the relative running time.

Looking the results with respect the Q-score column is possible to see as IMSA is comparable with all alignment algorithms, showing the third best performance, behind only to PROBCONS and PCMA algorithms, although it seems to be slower from a running time point of view. In fact, as shown in the last column of the same table, IMSA presents a larger running time with respect to the other algorithms, except PROBCONS whose running time is the highest.

Tables 9 and 10 show, respectively, the average SP and *Total-Column* (TC) scores, obtained on BALIBASE 3.0 data set. For this kind of comparison, as done in (56), we labelled the five categories of the BALIBASE benchmark as $RV1*$, $RV20, \ldots, RV50$, where the first class is further divided into two subcategories ($RV11$ and $RV12$).

Also on this kind of comparisons is possible to see as IMSA provides comparable alignments in term of quality, as determined via the used score metrics.

Table 8. IMSA versus COBALT (55), PROBCONS (27), PCMA (54), MUSCLE (34) and CLUSTALW (23)

Aligner	Q-score	Running time
PROBCONS (27)	**86.41**	32 h 11 min
PCMA (54)	85.75	5 h 39 min
IMSA	84.68	30 h 58 min
COBALT (55)	84.44	4 h 38 min
MUSCLE (34)	82.35	1 h 18 min
CLUSTALW (23)	75.37	1 h 21 min

The comparison was done using BALIBASE 3.0 (53) as benchmark. Best results are in boldface.

Table 7. IMSA versus ClonAlign (58), each entry reports the pair of values (SP, CS)

Instance	ClonAlign (59)	Clustal (59)	Muscle (59)	T-Coffee (59)	IMSA
1aab	(1.000, 1.000)	(0.940, 0.881)	(1.000, 1.000)	(1.000, 1.000)	(1.000, 1.000)
1aho	(1.000, 1.000)	(0.920, 0.857)	(1.000, 1.000)	(1.000, 1.000)	(1.000, 1.000)
2trx	(0.707, 0.500)	(0.707, 0.500)	(0.644, 0.386)	(0.752, 0.591)	(0.996, 0.614)
1tgxa	(0.849, 0.833)	(0.914, 0.933)	(0.785, 0.700)	(0.753, 0.667)	(0.991, 0.921)
1wit	(1.000, 1.000)	(0.873, 0.683)	(1.000, 1.000)	(0.980, 0.951)	(0.898, 0.871)
1ar5a	(0.977, 0.957)	(0.986, 0.976)	(0.995, 0.994)	(0.982, 0.970)	(1.000, 0.968)
gal4	(0.666, 0.459)	(0.698, 0.541)	(0.746, 0.430)	(0.683, 0.422)	(0.584, 0.506)
glg	(0.907, 0.845)	(0.956, 0.908)	(0.986, 0.982)	(0.987, 0.986)	(0.880, 0.830)
1amk	(0.993, 0.982)	(0.996, 0.991)	(0.996, 0.991)	(0.996, 0.991)	(1.000, 1.000)
1gdoa1	(0.779, 0.679)	(0.908, 0.835)	(0.862, 0.732)	(0.934, 0.884)	(0.882, 0.763)
451c	(0.707, 0.469)	(0.649, 0.429)	(0.622, 0.367)	(0.717, 0.469)	(0.773, 0.619)
Average	(0.871, 0.793)	(0.868, 0.752)	(0.876, 0.780)	(0.889, 0.812)	(**0.909, 0.826**)

Best results are in boldface.

Table 9. Average SP score for IMSA and several alignment algorithms, on BALIBASE 3.0 data-set

Aligner	*RV*11	*RV*12	*RV*20	*RV*30	*RV*40	*RV*50
MUSCLE (fast) (34)	0.4904	0.8303	0.8359	0.7076	0.6904	0.6823
MAFFT (fast) (59)	0.4801	0.8161	0.8404	0.7345	0.7187	0.7089
MAFFT *v*6 (parttree, $n = 50$) (59)	0.4790	**0.8066**	**0.8096**	**0.6801**	**0.6610**	0.6985
MAFFT (59)	0.4914	0.8258	0.8459	0.7437	0.7347	0.7253
GRAMALIGN (56)	0.5089	0.8328	0.8270	0.6855	0.7239	0.6903
KALIGN (60)	0.5029	0.8504	0.8410	0.7389	0.7259	0.7299
CLUSTALW (fast) (23)	**0.4748**	0.8367	0.8258	0.6843	0.6705	0.6715
MUSCLE (34)	0.5578	0.8583	0.8548	0.7492	0.7623	0.7384
CLUSTALW (23)	0.4908	0.8197	0.8219	0.6841	0.6950	**0.6698**
PSALIGN (61)	**0.5924**	**0.8804**	**0.8720**	0.7554	**0.7937**	**0.7739**
T-COFFEE (24)	0.5181	0.8650	0.8660	**0.7588**	0.7452	0.7715
IMSA	0.5638	0.8660	0.8690	0.7565	0.7534	0.7628

Best results are in boldface.

Table 10. Average Total-Column (TC) score for IMSA and several alignment algorithms, on BALIBASE 3.0 data set

Aligner	*RV*11	*RV*12	*RV*20	*RV*30	*RV*40	*RV*50
MUSCLE (fast) (34)	0.2421	0.6349	**0.2599**	**0.2457**	**0.2614**	0.2719
MAFFT (fast) (59)	0.2354	**0.6209**	0.3094	0.2910	0.3108	0.3087
MAFFT *v*6 (parttree, $n = 50$) (59)	0.2461	0.6320	0.2978	0.2987	0.3104	0.3435
MAFFT (59)	0.2532	0.6256	0.3168	0.3158	0.3073	0.3303
GRAMALIGN (56)	0.2993	0.6701	0.2917	0.2503	0.3292	0.3006
KALIGN (60)	0.2538	0.6749	0.2765	0.2955	0.3253	0.3223
CLUSTALW (fast) (23)	**0.2317**	0.6651	0.2680	0.2513	0.2808	0.2752
MUSCLE (34)	0.3217	0.6961	0.3077	0.3087	0.3484	0.3397
CLUSTALW (23)	0.2395	0.6417	0.2602	0.2478	0.3024	**0.2658**
PSALIGN (61)	**0.3503**	**0.7384**	**0.3517**	0.2992	**0.3951**	0.3816
T-COFFEE (24)	0.2716	0.6986	0.3257	**0.3637**	0.3659	**0.3974**
IMSA	0.3397	0.7119	0.3335	0.3245	0.3899	0.3823

Best results are in boldface.

Table 11. Running time necessary to align all data set of BALIBASE 3.0

Aligner	Running time (s)
MUSCLE (fast) (34)	200
MAFFT (59)	376
GRAMALIGN (56)	453
KALIGN (60)	886
MUSCLE (34)	4086
CLUSTALW (23)	5813
PSALIGN (61)	101 403
IMSA	110 073
T-COFFEE (24)	252 384

Finally, in Table 11 are presented the running times necessary to align all data set for all alignment algorithms shown in Tables 9 and 10.

FINAL REMARKS

We have designed a Clonal Selection Algorithm, called IMSA, to address the Multiple Sequence Alignment problem. This algorithm includes a new method to generate the initial population (*CLUSTALW-seeding*), and two specific *ad-hoc* mutation operators. To measure the alignment quality produced by IMSA, we have used the classical benchmark BALIBASE versions 1.0, 2.0 and 3.0. A favourable feature of IMSA is the ability of generating more than a single suboptimal alignment, for every MSA instance. This behaviour is due to the stochastic nature of the algorithm and of the populations evolved during the convergence process. This feature will help the decision maker to assess and select the biologically relevant multiple sequence alignment. The alignment process is not affected by the presence of distant sequences, and this can be considered another advantage of IMSA. Another important feature of the designed algorithm is that IMSA can be used by other aligners as a local search procedure to properly explore promising candidate solutions or regions of the search space.

Experimental results on BALIBASE v.1.0 show that IMSA is superior to PRRP, CLUSTALX, SAGA, DIALIGN, PIMA, MULTIALIGN and PILEUP8; while on BALIBASE v.2.0 the algorithm shows interesting results in terms of SP score with respect to established and leading methods, e.g. CLUSTALW, T-COFFEE, MUSCLE, PRALINE, PROBCONS and SPEM. Although the scoring function used by IMSA produces high SP values and low CS scores, future work will focus on the improvement of the CS score values using the T-Coffee scoring function. Using the same benchmark (BALIBASE v.2.0) IMSA was also compared with two immunological aligners. From these comparisons, IMSA shows best performances, and hence best alignments, than both *ClonAlign* and *AIS*.

For completeness, IMSA has been compared with the state-of-the-art alignment algorithms also on the BALIBASE v.3.0 benchmark. Also in this new testbed, IMSA shows good alignments, which are comparable with the state-of-the-art methods, as MUSCLE (FAST), MAFFT (FAST), MAFFT *v*6 (PARTTREE, $n = 50$), MAFFT, GRAMALIGN, KALIGN, CLUSTALW (FAST), MUSCLE, CLUSTALW, PSALIGN, and T-COFFEE.

ACKNOWLEDGEMENTS

The anonymous reviewers provided feedback that measurably improved the manuscript.

FUNDING

Funding for open access charge: EU projects.

Conflict of interest statement. None declared.

REFERENCES

1. Eidhammer,I., Jonassen,I. and Taylor,W.R. (2004) *Protein Bioinformatics*. Wiley, Chichester, West Sussex, UK.
2. Durbin,R., Eddy,S., Krogh,A. and Mitchison,G. (2004) *Biological Sequence Analysis*. Cambridge University Press, Cambridge, UK.
3. Notredame,C. (2007) Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput. Biol.*, **3**, 1405–1408.
4. Thompson,J.D., Plewniak,F., Ripp,R., Thierry,J.C. and Poch,O. (2001) Towards a reliable objective function for multiple sequence alignments. *J. Mol. Biol.*, **301**, 937–951.

5. Altschul,S.F. and Lipman,D.J. (1989) Trees stars and multiple biological sequence alignment. *SIAM J. Appl. Math.*, **49**, 197–209.
6. Altschul,S.F., Carroll,R.J. and Lipman,D.J. (1989) Weights for data related by a tree. *J. Mol. Biol.*, **207**, 647–653.
7. Cutello,V., Nicosia,G., Pavone,M. and Prizzi,I. (2007) Proteomic multiple sequence alignments: refinement using an immunological local search. *Applied and Industrial Mathematics in Italy, series on Advanced in Mathematics for Applied Sciences*, World Scientific, Vol. 75, pp. 291–302.
8. Lones,M.A. and Tyrrell,A.M. (2007) Regulatory motif discovery using a population clustering evolutionary algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 4, pp. 403–414.
9. Bevilacqua,V., Menolascina,F., Alves,R.T., Tommasi,S., Mastronardi,G., Delgado,A., Paradiso,M.R., Nicosia,G. and Freitas,A.A. (2008) Artificial immune systems in Bioinformatics. In Smolinski,T.G., Milanova,M.G. and Hassanien,A. (eds), *Computational Intelligence in Biomedicine and Bioinformatics: Current Trends and Applications*. Springer, New York, NY, USA, pp. 305–330.
10. Cutello,V., Nicosia,G. and Pavone,M. (2007) An immune algorithm with stochastic aging and kullback entropy for the chromatic number problem. *J. Combi. Optimi.*, **14**, 9–33.
11. Cutello,V., Nicosia,G., Pavone,M. and Timmis,J. (2007) An immune algorithm for protein structure prediction on lattice models. *IEEE Trans. Evol. Comput.*, **11**, 101–117.
12. Bonizzoni,P. and Vedova,G.D. (2001) The complexity of multiple sequence alignment with SP-score that is a metric. *Theor. Comput. Sci.*, **259**, 63–79.
13. Wang,L. and Jiang,T. (1994) On the complexity of multiple sequence alignment. *J. Comput. Biol.*, **1**, 337–348.
14. Garey,M.R. and Johnson,D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York.
15. Gupta,S.K., Kececioglu,J.D. and Schaffer,A. (1995) Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment. *J. Comput. Biol.*, 459–472.
16. Joo,K., Lee,J., Kim,I., Lee,S.-J. and Lee,J. (2008) Multiple sequence alignment by conformational space annealing. *Biophys. J.*, **95**, 4813–4819.
17. Lu,Y. and Sze,S.-H. (2008) Multiple sequence alignment based on profile alignment of intermediate sequences. *J. Comput. Biol.*, **15**, 767–777.
18. Rausch,T., Emde,A.-K., Weese,D., Döring,A., Notredame,C. and Reinert,K. (2008) Segment-based multiple sequence alignment. *Bioinformatics*, **24**, i187–i192.
19. Löytynoja,A. and Goldman,N. (2008) A model of evolution and structure for multiple sequence alignment. *Philos. Tran. R. Soc. Lond. B, Biol. Sci.*, **363**, 3913–3919.
20. Corpet,F. (1988) Multiple sequence alignment with hierarchical clustering. *Nucleic Acids Res.*, **16**, 10881–10890.
21. Devereux,J., Haeberli,P. and Smithies,O. (1984) A comprehensive set of sequence analysis programs for the VAX. *Nucleic Acids Res.*, 387–395.
22. Thompson,J.D., Gibson,T.J., Plewniak,F., Jeanmougin,F. and Higgins,D.G. (1997) The ClustalX windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Res.*, **24**, 4876–4882.
23. Thompson,J.D., Higgins,D.G. and Gibson,T.J. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, **22**, 4673–4680.
24. Notredame,C., Higgins,D.G. and Heringa,J. (2000) T-Coffee: a novel method for fast and accurate Multiple Sequence Alignment. *J. Mol. Biol.*, **302**, 205–217.
25. Smith,R.F. and Smith,T.F. (1992) Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Eng.*, **5**, 35–41.
26. Zhou,H. and Zhou,Y. (2005) SPEM: improving multiple sequence alignment with sequence profiles and predicted secondary structures. *Bioinformatics*, **21**, 3615–3621.
27. Do,C.B., Mahabhashyam,M.S.P., Brudno,M. and Batzoglou,S. (2005) ProbCons: probabilistic consistency-based multiple sequence alignment. *Genome Res.*, **15**, 330–340.
28. Carrillo,H. and Lipman,D.J. (1988) The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, **48**, 1073–1082.
29. Stoye,J., Moulton,V. and Dress,A.W. (1997) DCA: an efficient implementation of the divide-and conquer approach to simultaneous multiple sequence alignment. *Bioinformatics*, **13**, 625–626.
30. Morgenstern,B., Frech,K., Dress,A. and Werner,T. (1998) DIALIGN: finding local similarities by multiple sequence alignment. *Bioinformatics*, **14**, 290–294.
31. Morgenstern,B., Frech,K., Dress,A. and Werner,T. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, **15**, 211–218.
32. Gotoh,O. (1994) Further improvement in methods of group-to-group sequence alignment with generalized profile operations. *Bioinformatics.*, **10**, 379–387.
33. Eddy,S.R. (1995) Multiple alignment using hidden Markov models. *3rd International Conference on Intelligent Systems for Molecular Biology*, Vol. 3, pp. 114–120.
34. Edgar,R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, **32**, 1792–1797.
35. Notredame,C. and Higgins,D.G. (1996) SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.*, **24**, 1515–1539.
36. Notredame,C. (1998) COFFEE: an objective function for multiple sequence alignments. *Bioinformatics*, **14**, 407–422.
37. Simossis,V.A. and Heringa,J. (2005) PRALINE: a multiple sequence alignment toolbox that integrates homology-extended and secondary structure information. *Nucleic Acids Res.*, **33**, 289–294.
38. Shyu,C., Sheneman,L. and Foster,J.A. (2004) Multiple sequence alignment with evolutionary computation. *Genetic Prog. Evol. Mach.*, **5**, 121–144.
39. Nguyen,H.D., Yoshihara,I., Yamamori,K. and Yasunaga,M. (2002) Aligning multiple protein sequences by parallel hybrid genetic algorithm. *Genome Inform.*, **13**, 123–132.
40. Cutello,V. and Nicosia,G. (2006) A clonal selection algorithm for coloring, hitting set and satisfiability problems. *International Workshop on Natural and Artificial Immune Systems (NAIS)*, Vol. 3931 of *LNCS*, Springer, pp. 324–337.
41. Cutello,V., Narzisi,G., Nicosia,G. and Pavone,M. (2006) An immunological algorithm for global numerical optimization. *7th International Conference on Artificial Evolution (EA)*, Vol. 3871 of LNCS, Springer, pp. 284–295.
42. Cutello,V., Lee,D., Leone,S., Nicosia,G. and Pavone,M. (2006) Clonal selection algorithm with dynamic population size for bimodal search spaces. *2nd International Conference on Natural Computaion (ICNC)*, Vol. 4221 of LNCS, Springer, pp. 949–958.
43. Cutello,V., Lee,D., Nicosia,G., Pavone,M. and Prizzi,I. (2006) Aligning multiple protein sequences by hybrid clonal selection algorithm with insert-remove-gaps and blockshuffling operators. *5th International Conference on Artificial Immune Systems (ICARIS)*, Vol. 4163 of LNCS, Springer, pp. 13–28.
44. Cutello,V., Nicosia,G. and Pavone,M. (2003) A hybrid immune algorithm with information gain for the graph coloring problem. *Genetic and Evolutionary Computaion COnference (GECCO)*, Vol. 2723 of LNCS, Springer, pp. 171–182.
45. Cutello,V., Narzisi,G., Nicosia,G. and Pavone,M. (2005) Clonal selection algorithms: a comparative case study using effective mutation potentials. *4th International Conference on Artificial Immune Systems (ICARIS)*, Vol. 4163 of LNCS, Springer, pp. 13–28.
46. Cutello,V., Morelli,G., Nicosia,G. and Pavone,M. (2005) Immune algorithms with aging operators for the string folding problem and the protein folding problem. *5th European Conference on*

Comput.ion in Combinatorial Optimization (EvoCOP), Vol. 3448 of LNCS, Springer, pp. 80–90.
47. Cutello,V., Nicosia,G. and Pavone,M. (2004) Exploring the capability of immune algorithms: a characterization of hypermutation operators. *3rd International Conference on Artificial Immune Systems (ICARIS)*, Vol. 3239, of LNCS, Springer, pp. 263–276 (2004).
48. Cutello,V., Nicosia,G. and Pavone,M. An immune algorithm with hyper-macromutations for the dill's 2d hydrophobic–hydrophilic model. *Cong. Evol. Comput.*, **1**, 1074–1080.
49. Cutello,V., Nicosia,G., Pavone,M. and Narzisi,G. (2006) Real coded clonal selection algorithm for unconstrained global numerical optimization using a hybrid inversely proportional hypermutation operator. *21st Annual ACM Symposium on Applied Computing (SAC)*, Vol. 2, ACM, 950–954.
50. Thompson,J.D., Plewniak,F. and Poch,O. (1999) BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*, **15**, 87–88.
51. Bahr,A., Thompson,J.D., Thierry,J.C. and Poch,O. (2001) BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Res.*, **29**, 232–326.
52. Thompson,J.D., Koehl,P., Ripp,R. and Poch,O. (2005) BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins Struct. Funct. Bioinformatics*, **61**, 127–136.
53. Castrogiovanni,M., Nicosia,G. and Rascuna,R. (2007) Experimental analysis of the aging operator for static and dynamic optimisation problems. *11th International Confernce on Knowledge-Based and Intelligent Information and Engineering Systems - KES 2007*, Vietri sul Mare, Italy. Springer, Vol. 4694 of LNCS, pp. 804–811.
54. Pei,J., Sadreyev,R. and Grishin,N.V. (2003) PCMA: fast and accurate multiple sequence alignment based on profile consistency. *Bioinformatics*, **19**, 427–428.
55. Papadopoulos,J.S. and Agarwala,R. (2007) COBALT: constraint-based alignment tool for multiple protein sequences. *Bioinformatics*, **23**, 1073–1079.
56. Russell,D.J., Out,H.H. and Sayood,K. (2008) Grammar-based distance in progressive multiple sequence alignment. *BMC Bioinformatics*, **9**.
57. Ge,H., Zhong,W., Du,W., Qian,F. and Wang,L. (2007) A hybrid algorithm based on artificial immune system and hidden markov model for multiple sequence alignment. *International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2007)*. Atlantis Press.
58. Layeb,A. and Deneche,A.H. (2007) Multiple sequence alignment by immune artificial system. *ACS/IEEE International Conference on Computer Systems and Applications*, pp. 336–342.
59. Katoh,K., Misawa,K., Kuma,K. and Miyata,T. (2002) MAFFT: a novel method for rapid multiple sequence alignment based on fast fourier trasform. *Nucleic Acids Res.*, **30**, 3059–3066.
60. Lassmann,T. and Sonnhammer,E. (2005) Kalign – an accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics.*, **6**.
61. Sze,S., Lu,Y. and Yang,Q. (2006) a polynomial time solvable formulation of multiple sequence alignment. *J. Comput. Biol.*, **13**, 309–319.