Using a β-Contact Predictor to Guide Pairwise Sequence Alignments for Comparative Modelling

Filippo Ledda, Giuliano Armano, and Andrew C. R. Martin

Abstract—With the exponential rise in the number of available protein sequences, prediction of protein tertiary structure has become one of the most important tasks in bioinformatics; "comparative", or "homology", modelling is able to provide accurate models, but sequence alignment is a critical task. A strong correlation holds between the RMS deviation of models and the occurrence of errors in the alignment. In order to correct such errors, we developed BCAlign, based on an optimization procedure taking into account the correctness of the assignments of β-contacts, together with a standard scoring system. A β-contact evaluator (BCEval), based on a mixture of neural networks, is used to evaluate the assignments. The evaluation of β-contacts has proved to be a useful measure in improving alignments for comparative modelling. Considering the fraction of useful alignments below 3Å, the models generated by BCAlign have shown a significant overall improvement compared with Needleman and Wunsch's pairwise and multiple alignments obtained with MUSCLE. Further improvements were observed where BCEval shows high confidence in the alignments generated. The method has been made available as a web server at http://iasc.diee.unica.it/bcserver, with a REST-style interface also available.

Index Terms—Homology Modelling, Sequence Alignment, β-contacts, Ensemble Architectures, Artificial Neural Networks.

1 Introduction

The difference between the number of protein sequences translated from sequences held in GenBank [3] and the number of protein structures held by the PDB (Protein DataBank) [4] is vast. Only recently have high throughput methods started to be put in place to solve protein structure. Comparative modelling [5] offers a way to bridge the gap between the number of sequences and structures.

Comparative modelling generally relies on knowing the structure of a homologous protein and using that as a template to build the structure of a protein. Methods include 3D-JIGSAW [2], FAMS [28], ESyPred3D [20], RAPPER [8]. COMPOSER [37], [38] and the particularly popular SwissModel [1], [31] and MODELLER [10], [32], [41].

However, the limiting factor in all these methods is obtaining the correct alignment. This is the most important stage of comparative modelling [7], [24], but unfortunately, particularly at low sequence identity, it can be the most difficult to get right. The sequence alignment one wishes to achieve is the alignment that would be obtained by performing a structural alignment and reading off the resulting sequence alignment. Of course the structure of the target is not available, so one must rely on a sequence alignment. While multiple alignment can help, the sequence alignment can often differ substantially from the structural alignment.

There are numerous methods for performing structural alignment which often differ in the precise details of their results (e.g. CE [34], SSAP [39], STRUCTAL [36], DALI [14], MATRAS [17], VAST [12], SSM [19]). Since there are many different ways to superimpose two or more protein structures, if the proteins are not identical (or at least extremely similar), then there can be no single optimal superposition [27]. For our purposes, we have chosen SSAP as the gold standard, "correct" alignment.

The most extreme types of misalignment (Misleading Local Sequence Alignments, MLSAs) are areas where the sequence alignment for a region is very clear, yet it does not match the structure-derived alignment [33]. We define less extreme misalignments, where the sequence and structural alignments do not agree, as SSMAs ("Sequence-Structure MisAlignments"). For example, Figure 1 shows the sequence and structural alignment of a region from 1igmH00 and 1ap2A00 (a human and mouse antibody heavy chain variable region respectively) where an SSMA can clearly be seen.

In their analysis of the CASP2 comparative modelling section, Martin *et al.* [24] showed that there was a relationship between the percentage of correctly aligned residues and the sequence identity (Figure 2 of their paper). We have reproduced that analysis using approximately 56,000 pairs of homologous protein domains from CATH [29], [30], each of which was

G. Armano and F. Ledda are with the Department of Electrical and Electronic Engineering, University of Cagliari, Italy e-mail: armano@diee.unica.it, filippo.ledda@gmail.com.
A. C. R. Martin is with Institute of Structural and Molecular Biology, University College, London.
Manuscript received November 4, 2012; revised Month day, year.

```
1ap2A00                          DIVMTQSPSSLTVTAGEKVTM
1igmH00 Sequence alignment       EVHLLESGGNL-VQPGGSLRL
1igmH00 Structural alignment EVHLLESG-GNLVQPGGSLRL
                                          ****
```

Fig. 1. An example of an SSMA found between CATH domains 1igmH00 and 1ap2A00. The SSMA is indicated with asterisks.

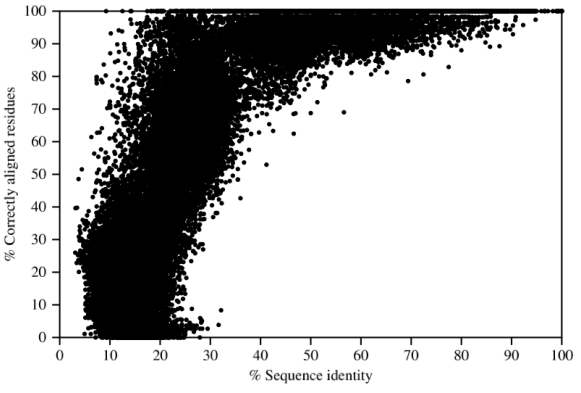

Fig. 2. The relationship between the percentage of correct sequence alignment and the percentage of sequence identity. Each pair of NRep domains in each CATH homologous family has been structurally aligned by SSAP and sequence aligned using a Needleman and Wunsch global alignment. The structural alignment is taken as the correct alignment. Twelve outlying points have been removed after being identified as occurring, owing to errors in the CATH database.

aligned on the basis of structure using SSAP and on sequence using a Needleman and Wunsch sequence alignment [26]. Figure 2 clearly shows that if there is a high sequence identity between two sequences, then the sequence alignment is likely to match the structural alignment. However as sequence identity decreases, particularly below 30%, the accuracy of the alignment decreases and the sequence-based alignment can be completely different from the structural alignment. In this paper, we concentrate on improving the alignment in β-sheets and therefore hope to improve the models obtained.

Previous work by Lifson & Sander [22], Wouters & Curmi [42], Hutchinson *et al.* [15] and Fooks *et al.* [11] has shown clear residue pairing preferences between adjacent β-strands. With this in mind, we believe that some sequence misalignments can be detected and corrected by detecting errors in the assignment of β-contacts. Given a pair of β-strands (a "β-pair") assigned to a target from a template after initial sequence alignment, a measure of the likelihood of the register between the paired being formed in a real protein can be used as part of a scoring system of an alignment algorithm. Thus we developed BCEval, a β-contact evaluator based on a mixture of neural networks, able to predict whether a pair of β-strands is in the correct register. In addition, a pairwise sequence alignment method (BCAlign) has been developed able to take into account the β-contact evaluations. A search algorithm controlled by an iterative procedure had to be adopted to find the alignment instead of a classical dynamic programming technique such as Needleman and Wunsch. This is because the score of a substitution in the alignment will depend on the mutual register with another substitution along the sequence (because the register will affect the β-pairing), thus breaking the basic assumption of dynamic programming. In other words, while searching for the best alignment, the contacts of the parent template are assigned to the target; the scoring system then takes into account both of the assigned β-strands at the same time, so that the substitutions within a strand cannot be scored without taking into account the information about the neighbouring strand.

In this paper, we introduce both BCEval and BCAlign. The accuracy of BCAlign is assessed against (i) the standard Needleman and Wunsch pairwise sequence alignment, (ii) multiple alignments obtained with MUSCLE [9] and (iii) an equivalent of BCAlign without the use of the evaluator (NoBCAlign). Additionally, the RMSD of models built using the different alignments is compared. The method has been made available as a web server at http://iasc.diee.unica.it/bcserver.

2 METHODS

When the homology modelling target and template sequences are aligned, the structural characteristics of the template are assigned to the target. Thus the secondary structure and the relative position within the structure (including interactions with other residues) are immediately known for the target sequence. A mis-alignment will lead to a wrong structural assignment. Thus we are able to examine contacts between residues in adjacent β-strands in an attempt to detect misalignments using an evaluation of an assigned β-pair being correct based on machine learning (BCEval).

At first glance, including these evaluations in the scoring system of a typical dynamic programming algorithm seems straightforward. Unfortunately, the main dynamic programming assumption (that the optimal solution of the problems should depend on the optimal solution of its sub-problems) is broken. In order to overcome this limitation, we developed a

```
----AA-----AA------BBBB-----CCCCC-----CCCCC-------BBBB------
----12-----12------1234-----12345-----54321-------4321------
QSPVDIDTHTAKYDPSLKPLSVSYDQATSLRILNNGHAFNVEFDDSQDKAVLKGGPLDGT
CCCCEECCCCCEECCCCCCEEEECCCCCEEEEEECCCCEEEEECCCCCCCEEEECCCCCC
```

Fig. 3. An example chain indicating the residues in contact. The letters in the first line indicate the β-strand pairs. The numbers in the second line indicate the residues in contact within the same pair. For example, the two residues labelled B1 form a contact.

technique which adopts a heuristic search algorithm (BCAlign).

2.1 Developing the β-Contact Evaluator (BCEval)

The evaluation of β-contacts can be tackled as a prediction problem, similar to contact map prediction. We must (i) define the training data, (ii) find a suitable representation of the input and output data and (iii) set up a proper architecture and learning algorithm(s). Methods were implemented using the GAME framework [21], written in Java 6.0.

Generic contact predictors such as those by Cheng & Baldi [6] and Tegge *et al.* [40] have low accuracy owing to the difficulty of predicting all possible contacts occurring in a protein (including between α-helices). Even more specific predictors, specialized in β-contacts, report accuracies below 50% [23]. Fortunately, we already know which strands are in contact and we can concentrate on small shifts around a given position. Thus, we developed a new system specialized in recognizing a contact from the "shifted" versions that could be identified from an alignment procedure.

2.1.1 Data Representation

The β-pairs must be represented in a fixed-length vector to obtain an input suitable for a neural network. The input vector must contain the residues of the two strands involved in the pairing and shifted versions of the same pair must be clearly recognizable.

Figure 3 shows an example in which the contacting residues belonging to different β-strands are indicated. While the length of the β-segments is variable, a fixed-length vector is needed for the data representation. A window of N residues would be perfectly suited to strands of length N, while information would be lost for pairings of longer strands and shorter strands would include residues not involved in contacts.

In addition, one must account of both parallel and anti-parallel strands. For instance, taking a window of four residues along the anti-parallel strands, *B*, in Figure 3, the encoding must indicate that the leucine at the first position in the first strand is in contact with the glycine in the last position of the second strand, not the valine in the first position. The different hydrogen-bonding patterns observed in parallel and anti-parallel sheets also result in different propensities in the contacts between residues, as shown by Hutchinson *et al.* [15] and Fooks *et al.* [11]. For these reasons, a "mixture of experts" approach has been adopted: one expert only deals with strands of one type and length.

Profiles, obtained after three iterations of a PSI-BLAST search of the whole protein against *uniref90*[1], (inclusion threshold $= 10^{-3}$; defaults for other parameters) were used to encode the residues in the window. A simple position-independent coding of the residues gave worse performance.

2.1.2 The Architecture

Figure 4 shows the architecture of BCEval. The "core evaluation module" of BCEval consists of a mixture of 13 neural networks, each one specialized for a specific length (1,2,3,4,5,6,7+) and type (parallel or anti-parallel) of β-pairing. The window length includes all (and only) the residues involved in each pairing, such that each neural network has a fixed-length vector as input, representing the residues involved in the contact. Simpler architectures with only one neural network and fixed input length (1, 2, 3) were tried first, but gave lower accuracy. The final output is obtained by averaging three core evaluation modules trained separately.

2.2 Training and Test Data Composition

A reference test set, TESTDOM, was built by selecting 10% of the total codes in the CATH database [29] at the homologue level and extracting the corresponding domains. A subset of the possible pairs of homologous domains in TESTDOM, mostly distant homologues (67% were below 30% sequence identity), was used to build a set of domain pairs. The resulting set, TESTALIGN, consists of 743 proteins, which have been used to test the alignment algorithms. In the same way, another set, TRAINALIGN, was obtained from the domains excluded from TESTDOM in order to train the parameters of the alignment algorithms. Finally, a set of protein chains, TRAINCH, consisting of protein chains from a dataset with identity $< 25\%$, [2]. selected in order not to include any chain containing the domains in TESTDOM, was used as a starting point to obtain contacts used in training BCEval; whole chains were used rather than domains

1. http://www.ebi.ac.uk/uniref/
2. http://bio-cluster.iis.sinica.edu.tw/~bioapp/hyprosp2/dataset_8297.txt

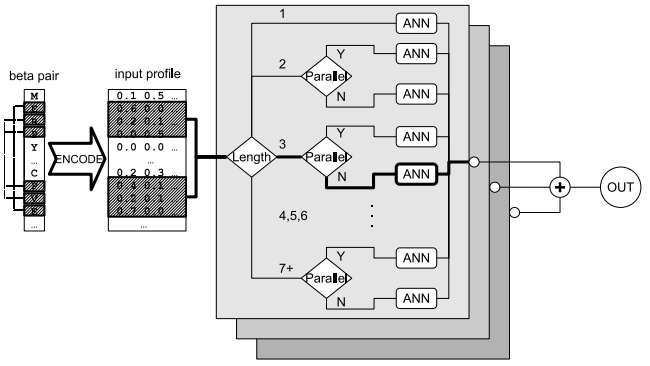

Fig. 4. The BCEval architecture. The guarding functions ensure that only one neural network is activated at a time. The "parallel" guard is able to distinguish between parallel and anti-parallel strand pairs, while the "length" guard dispatches based on the length. In the example, an anti-parallel pair of length 3 is given so activating the path shown in bold. Three core units consisting of independently trained neural networks are averaged to obtain the final evaluation.

in order to use DSSP [16] outputs from the EBI[3] directly. Contacts included in the training of BCEval were obtained from TRAINCH DSSP files; these files include the position of the contacts between paired residues in β-strands. Negative examples (i.e. pairings not observed to be in a β-contact) were obtained by a synthetic sampling around the actual contacts. A Gaussian distribution ($\mu = 0, \sigma = \sqrt{5}$) around the positive examples was used to perform the sampling. The negative and positive samples were balanced, without taking into account the observed distribution. As seen in Figure 2, the extent of the observed shifts depends greatly on the sequence identity, making it hard to model the observed distribution correctly and, in any case, balanced inputs generally result in better learning. This partial synthetic sampling was preferred to sampling real data in order to obtain more, and more varied, samples. In practice, the negative data are randomly generated at each training iteration, so improving the diversity given to the training algorithm.

As for the training technique and parameter setting, each expert is a 3-layer feed-forward neural network, trained with a variant of back-propagation, with initial learning rate = 0.001 and momentum = 0.1. The learning rate is adjusted between iterations with an inverse-proportion law. The number of input neurons is $20N$, (where N is the size of the input window) and the number of hidden neurons is 75 for each neural network. A single output neuron indicates whether the given input is a contact or not. To help the training algorithm avoid local minima, the training set was randomly shuffled at each iteration. Furthermore, each protein provides only a subset of its inputs, according to a random choice performed in accordance with a parameter, n. In particular, a random value k is generated in the range $[0, n-1]$ and the inputs with index k, $k+n$, $k+2n, \ldots$ are provided to the learning algorithm.

To prevent the training process from stopping with a local oscillation of accuracy (evaluated on a validation set consisting of a 10% of TRAINCH, not used in the back-propagation process), weights are recorded when a minimum is encountered on the validation set, but the training continues until the error on the validation set increases for 10 consecutive iterations.

2.3 Developing the Pairwise Sequence Alignment Algorithm (BCAlign)

The definition of an alignment algorithm includes two separate parts: (i) the cost function i.e. a scoring scheme used to evaluate an alignment; (ii) the alignment strategy, i.e. a strategy which gives the succession of substitutions, insertions and deletions which minimize the cost function. Here we describe a cost function which includes the evaluation of β-pairings made by BCEval and an alignment strategy suitable for use with the given cost function. Full details of both are provided in Appendix A.

2.3.1 Defining the Cost Function

In brief, given a pairwise sequence alignment, A, between a template and a target sequence (the structure of the template being known), its cost[4] $c(A, S_{tpl})$, evaluated with respect to the structure of the template sequence, in BCAlign consists of the sum of the following three main contributions:

$$c(A, S_{tpl}) = c_{nw}(A) + c_{\beta i}(A, S_{tpl}) + c_{bc}(A, S_{tpl}) \quad (1)$$

where S_{tpl} is the structure of the template sequence.

The component c_{nw} is the result of a classical similarity-based scoring scheme with affine gap penalties. The cost of the substitutions is obtained from a similarity scoring matrix M_s (e.g. BLOSUM62), reversed so as to obtain a cost matrix $M_c = -(M_s - \max(M_s))$.

The term $c_{\beta i}$ in Equation 1 is related to the total number of gaps within the β-strands in the template. This is similar to the approach adopted in PIMA [35]. This component has been included in order to increase the number of β-pairs available for the evaluation: the larger costs help to avoid insertions and deletions inside β-strands, which rarely occur during evolution.

The last term in Equation 1, c_{bc}, results from the evaluation of β-pairs in the target sequence (assigned from the template, based on the alignment). This has two effects:

3. ftp://ftp.ebi.ac.uk/pub/databases/dssp/

4. Scores, usually preferred in the scoring systems of sequence alignments, can also be viewed as the opposite of costs.

- to increase the cost of β-pairs that appear to be mistakenly assigned (i.e. shifted),
- to decrease the cost of β-pairs that appear to be assigned correctly.

The first of these changes the equilibrium of the alignment space, moving away from the solutions suggested by the other two terms that lead to wrong β-pair assignments. The second, although not directly improving the alignment, prevents drifts when correct assignments are found with the standard scoring scheme. The change of a pair of assignments may affect the solution in many different places within the alignment.

The term c_{bc} is composed of two elements, summed over all β-pairs:

1) A term proportional to $-\tilde{p}(bp_{tgt})$, where $\tilde{p}(bp_{tgt})$ is the estimation of the probability of the β-pair bp_{tgt} being formed, given by BCEval (see Appendix A),
2) A term to stabilize the algorithm in the presence of wrong estimations which also takes account of the corresponding β-pair in the template (bp_{tpl}), for which the estimation error is known to be $1-\tilde{p}(bp_{tpl})$. The requirement for this term derives from the assumption that, with the correct alignment, the errors in the estimation of p on the template and target are correlated. Therefore, if $\tilde{p}$ for the template is significantly larger than for the target, we have a strong indicator of a probable error in the alignment (see Appendix A).

2.3.2 Minimizing the Cost Function

Dynamic programming is generally used to minimize a cost function for sequence alignments and is the best choice when the optimal solution can be built incrementally by calculating the best solution for its sub-problems. With the proposed cost function, this assumption is broken, since the cost of a substitution is related to other substitutions along the sequences. A natural generalization of dynamic programming is represented by a search algorithm, which allows to evaluate the path dynamically.

Using a global-search algorithm, the best alignment can be found by searching for the path in a tree which optimizes a score or cost function, leading to the end of the sequences. Figure 5 gives an example of a simple alignment performed with a best-first search strategy.

However, search algorithms may lead to an explosion in computational cost; in Figure 5, a blind (brute-force) search strategy is adopted, with the consequence that many nodes are expanded unnecessarily before finding the solution. The expected number of expanded nodes grows exponentially with the length of the path, which grows linearly with the length of the sequences. Consequently, to reduce the number of expanded nodes, heuristic search strategies,

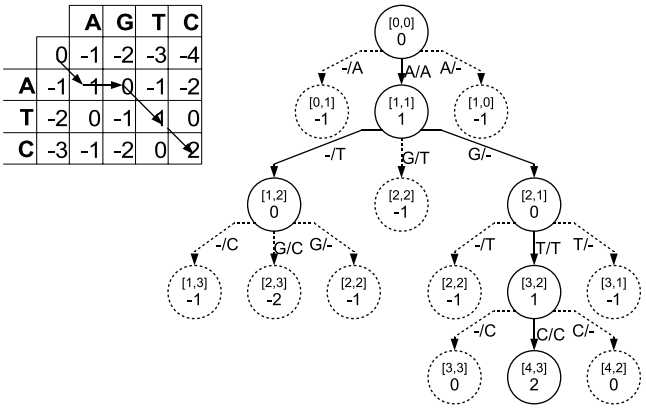

Fig. 5. An example of alignment performed with a search algorithm. The search can be represented by a tree, in which the edge score is given by a simple scoring system (-1 for gaps, 1 match, -2 mismatch). Each circle represents a node, indicating the position in the two sequences and the path score. With a best-first search (i.e. the most promising nodes are opened first), the nodes shown with solid lines are expanded. In addition, nodes outside the solution path (in dashed lines) are explored, according to the local score. On the left, the corresponding Needleman and Wunsch matrix is indicated: note that the values in the Needleman and Wunsch matrix correspond to the scores of a node only when the best path to that node is followed.

such A* [13], can be adopted. A perfect heuristic (i.e. one which provides perfect estimates) for the components of the cost c_{nw} and $c_{\beta i}$ (Equation 1) can be obtained by adapting the approach used by dynamic programming algorithms. Hence, with only these two components, only the nodes in the optimal path are expanded by the A* algorithm, making this equivalent to a global dynamic programming approach. However, the component c_{bc} in Equation 1 cannot take advantage of any heuristic cost estimator and is relatively expensive to compute — a search algorithm computing this component dynamically would be computationally too expensive. Consequently, an iterative approach was adopted: after each iteration (the first being run without the component c_{bc}), the resulting β-pairs are collected and evaluated for use in the next iteration. The additional information is thus introduced step-by-step, permitting the algorithm, at each iteration, to escape from misleading pairings reached by following the other two components of the cost (c_{nw} and $c_{\beta i}$). The Iterative-Deepening A* (IDA*) [18] algorithm is used to perform the search. For further details, see Appendix A.

2.4 Evaluation Criteria

Two criteria were used to evaluate the results: (i) the fraction of correct substitutions (FCS) was measured by comparing the sequence alignment against a reference structural alignment obtained using SSAP [39],

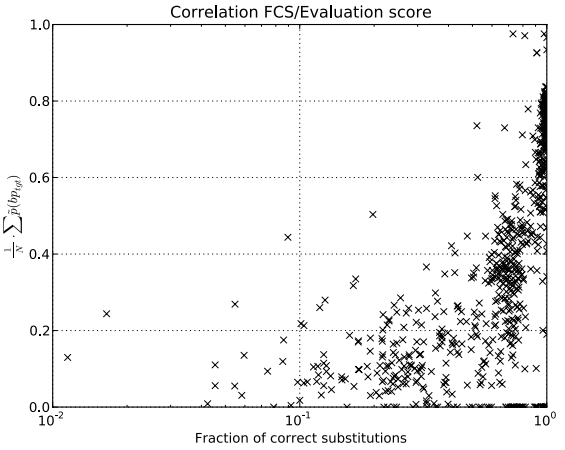

Fig. 6. Plot of the β-contact predictor (BCEval) score *vs.* the "fraction of correct substitutions" (FCS). Where BCEval scores zero, no β-pairs were assigned after the alignment because no contacts were present or because all were broken by gaps in the alignment.

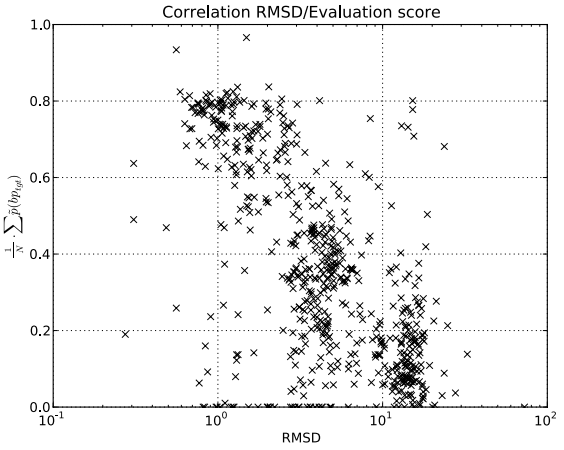

Fig. 7. Plot of the β-contact (BCEval) score *vs.* RMSD (Å) for three-dimensional comparative models generated using MODELLER.

(ii) the RMSD of models generated from the alignments using MODELLER [10], [41] in fully automatic mode with default parameters[5]. Fitting of models to the crystal structures was performed using the McLachlan algorithm [25] as implemented in the program ProFit (Martin, A.C.R., http://www.bioinf.org.uk/software/profit/).

For each experiment, we calculated the average RMSD of the models obtained as well as the percentage of "acceptable" models, i.e. those with RMSD below 3Å, which is considered to be quite a strict criterion for distant homologues. The percentage of acceptable models is more indicative of the utility of the alignments than is the mean, since the latter can easily be skewed by very bad models. In practical terms there is no difference between a "bad" and a "very bad" model.

An additional parameter, the "SSMA distance" (SSMAD), defined as the mean distance of each residue from its correct position in the reference structural alignment, as used in our earlier work [24] was also tested, but was found to correlate less well with RMSD than the simpler FCS measure.

3 Results

3.1 BCEval

On average over a 7-fold cross validation on TRAINCH, BCEval achieved an accuracy of 0.785, precision of 0.771, recall of 0.811 and Matthews Correlation Coefficient (MCC) of 0.571 –full results are shown in Table 1. In order to assess the use of BCEval in the evaluation of alignments, the correlation between the actual performance for a series of alignments and an evaluation metric from BCEval for that alignment was analysed. This metric was the mean of the evaluations for the target protein: $\tilde{p}(bp_{tgt})$. Needleman and Wunsch alignments were also generated, scored with the BLOSUM45 matrix and gap opening/extension penalty of $13/1$. Figures 6 and 7 plot the BCEval metric against the fraction of correct substitutions (FCS) and the RMSD respectively. The existence of a considerable correlation between the scores and the alignment quality suggests that BCEval scores can be used effectively to chose the best alignment in a set and that β-pairs can be exploited to enhance pairwise sequence alignments.

5. Only 637 models of the 743 of TESTALIGN were obtained from the alignments owing to problems in the automatic process which extracted the indexes for the domains from the PDB files. The problem is often caused by fragmented domains which include non-consecutive parts of sequence.

3.2 BCAlign

Preliminary experiments were run on a subset of 500 domain pairs from TRAINALIGN to optimize parameters and the following were then used for all runs: $c_{go} = 22$, $c_{gx} = 9$ $c_{g\beta} = 6$, $\gamma_{abs} = 75$ and $\gamma_{rel} = 5$ (see Appendix A). Substitutions are scored using BLOSUM45; the algorithm is best suited to distant homologues since, for sequence alignments between close homologues, a standard sequence alignment is usually sufficiently reliable, and very few SSMAs are detected. The maximum number of iterations was set to 5 with a limit of one minute imposed on the search algorithm at each iteration using an Intel SU9600 CPU. The main code was written in Java and experiments were scripted using Python via the Jython 2.5 interpreter.

The performance of BCAlign was assessed in three comparisons: (i) with a Needleman and Wunsch alignment (scored using the BLOSUM45 matrix and gap opening/extension penalties 13/1, optimized as

Fold	Accuracy	Precision	Recall	MCC
1	0.781	0.765	0.807	0.563
2	0.783	0.774	0.797	0.565
3	0.795	0.777	0.821	0.592
4	0.784	0.777	0.802	0.568
5	0.784	0.768	0.812	0.569
6	0.790	0.764	0.840	0.582
7	0.778	0.770	0.797	0.556
AVG	0.785	0.771	0.811	0.571

TABLE 1
Results for BCEval in a 7-fold cross validation test on the dataset TRAINCH.

above), (ii) with multiple alignments obtained using MUSCLE [9], and (iii) with the same search technique, but without the use of the evaluator ("NoBCAlign") –i.e. using optimized parameters as above, but setting $\gamma_{abs} = 0$ and $\gamma_{rel} = 0$. MUSCLE was run using standard parameters, including all the CATH homologous sequences contained in TESTDOM in the multiple alignments.

On the TESTALIGN dataset, BCAlign shows a relative improvement[6] of 11.3% ($0.628 vs.\ 0.703$) in FCS compared with Needleman and Wunsch, 1.4% compared with MUSCLE and 6.2% compared with NoBCAlign. The RMSD improves by 7.14% (5.99Å $vs.\ 6.43$Å) compared with Needleman and Wunsch, and by 6.59% compared with NoBCAlign. However BCAlign performs slightly worse than MUSCLE (-2.6%) when assessed on RMSD. The large values of RMSD result from the fact that the majority of the alignments in the test set have sequence identity below 25%. In addition, as seen in Figure 7, a few models have extremely large RMSDs, skewing the mean value.

The percentage of acceptable models (i.e. with RMSD < 3.0Å) is probably a more useful measure of the success of an alignment method. In this experiment, this was 42% for BCAlign, 36% for Needleman and Wunsch, 35% for MUSCLE and 39% for NoBCAlign. Unexpectedly, multiple alignment using MUSCLE performed worst in this evaluation.

Better results are obtained by restricting comparisons to data for which we expect BCAlign to perform well, i.e. where a large number of β-pairings are present and the BCEval score improves. For structures with at least 8 β-pairs (58% of the alignments) the RMSD improvement is 1.22% over MUSCLE and 8.83% over NoBCAlign. The percentage of acceptable models improves to 48% for BCAlign, compared with 41% for Needleman and Wunsch, 39% for MUSCLE and 44% for NoBCAlign, evaluating the same set of models.

In addition, the BCEval scores can be used to select those cases where BCEval makes confident predictions. Figures 8 and 9 show the average relative improvement in RMSD and FCS between BCAlign pairwise alignment and MUSCLE multiple alignment, when varying an inclusion threshold based on the improvement in the assignment of β-pairs when comparing MUSCLE and BCAlign alignments, as evaluated using BCEval. The graphs clearly show that, by using an inclusion threshold of less than -0.3 (thus including up to 20% of alignments), substantial improvements in FCS and RMSD can be obtained compared with other methods. For example, taking alignments with at least 8 β-pairings and an inclusion threshold of -0.3 (Figure 9), the percentage of proteins with RMSD lower than 3Å is 39% for BCAlign, 21% for Needleman and Wunsch, 15% for MUSCLE and 26% for NoBCAlign.

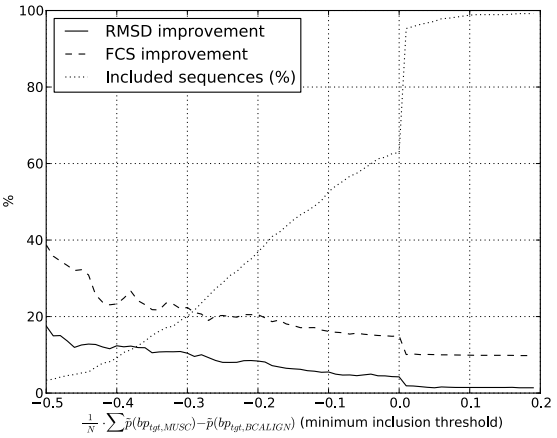

Fig. 8. Average improvement in FCS and RMSD compared with MUSCLE at different inclusion thresholds. The threshold is an a priori measure of the confidence about the alignment, which consists of the difference in the BCEval score between alignments obtained with BCAlign and MUSCLE. At each point in the plot, the alignments below the given threshold are included. The percentage of included alignments at each threshold is also shown.

Overall, BCAlign showed a considerable improvement compared with conventional pairwise Needleman and Wunsch alignment of 11.3% in FCS on a set of 743 alignments of domains not showing homology with the data used to train the evaluator. Three-dimensional models obtained from the alignments show an average RMSD improvement of 7.14%, compared with standard Needleman and Wunsch sequence alignments. In addition, BCAlign results are, on average, comparable with multiple alignments obtained with MUSCLE. However, Figures 8 and 9 show that choosing the 20% best-scoring alignments according to the evaluator, models obtained with BCAlign show a considerable improvement in the RMSD of about 10% over MUSCLE. The percentage

6. Relative improvements are calculated with the formula: $RI(a,b) = \frac{(a-b)}{(a+b)/2} \cdot 100$

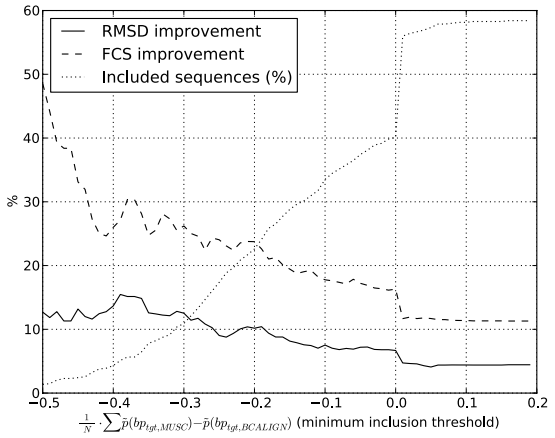

Fig. 9. Average improvement in FCS and RMSD compared with MUSCLE at different inclusion thresholds, for proteins containing at least 8 β-pairs. The threshold consists of the difference in the BCEval score between alignments obtained with BCAlign and MUSCLE. At each point in the plot, the alignments below the given threshold are included. The percentage of included alignments at each threshold is also reported.

of acceptable models shows an improvement of about 22% over MUSCLE when all proteins are considered and about 23% when only proteins containing at least 8 β-pairs are considered.

In conclusion, BCAlign appears to perform best when used in a mixed environment, in which different techniques compete while taking into account the scores assigned by BCEval. Restricting the use of BCAlign to those cases where BCEval makes the most confident predictions greatly increases its effectiveness. Even including the best 50% of the alignments shows BCAlign to be a good strategy (5% improvement over MUSCLE).

4 Conclusions and Future Work

Sequence alignment is the most critical task in comparative modelling: a strong correlation holds between the RMS deviation of models and the occurrence of errors in the alignment. In order to improve alignments, we have exploited the likelihood of a given pairing between β-strands being correct. Since the location of β-strands is known for the template it can be assigned to the target sequence after the alignment. Our β-contact evaluator, BCEval, estimates the likelihood of assigned β-pairings occurring in real proteins by using a mixture of neural networks.

BCEval has then been exploited in a novel sequence alignment technique, BCAlign. We have presented a scoring system which combines a normal system based on a substitution matrix with BCEval. Since it is not possible to use standard dynamic programming with this scoring system, BCAlign resorts to a search algorithm, guided by an external loop to control the maximum run time.

Experiments confirm the validity of the approach: BCEval predictions show a considerable correlation with correct β-pair assignments and alignments obtained with BCAlign show that the evaluation of assigned β-pairs can be successfully exploited to enhance sequence alignments.

Finally, the implementation of the algorithms can probably be further improved. The computation is still not sufficiently efficient, frequently reaching the time limit for long sequences which, on average, will have more β-strands that can be exploited by the method and therefore are likely to show the best improvements. The search algorithm could be improved, particularly by enhancing the heuristic function to decrease the alternative paths that are explored. Alternatively, it may be possible to design a better control loop able to include the evaluations without overloading the search algorithm, or to use stochastic local search algorithms, including genetic algorithms.

Acknowledgments

This work was supported by the Italian Ministry of Education – Investment funds for basic research, under the project ITALBIONET (Italian Network of Bioinformatics) and by the Visiting Professor Programme of the University of Cagliari.

Appendix A

A.1 Defining the Cost Function In Detail

Given a pairwise sequence alignment, A, between a template and a target sequence (the structure of the template being known), let us recall that the cost $c(A, S_{tpl})$ is defined as the sum of three contributions, i.e. $c_{nw}(A)$, $c_{\beta i}(A, S_{tpl})$, and $c_{bc}(A, S_{tpl})$, where S_{tpl} denotes the structure of the template sequence (see also Equation (1)).

The component c_{nw} is the result of a classical similarity-based scoring scheme with affine gap penalties:

$$c_{nw}(A) = c_{go} \cdot n_{go} + c_{gx} \cdot n_{gx} + \sum_{i,j} M_c(P_{tpl_i}, P_{tgt_j}) \quad (2)$$

where i and j indicate the position of the substituted residues, n_{go} the number of gap openings, and n_{gx} the number of gap extensions, all according to the alignment A. c_{go} is the cost of opening of a new gap, c_{gx} is the cost of extending a gap, and M_c is a substitution cost matrix obtained by reversing a similarity scoring matrix M_s (such as BLOSUM62):

$$M_c = -(M_s - \max(M_s)) \quad (3)$$

The second term, $c_{\beta i}$, in Equation 1 is an additional gap penalty to penalize gaps within the β-strands of the template:

$$c_{\beta i} = n_{g\beta} \cdot c_{g\beta} \quad (4)$$

where $c_{g\beta}$ is β-specific gap penalty and $n_{g\beta}$ is the number of gaps within β-strands.

The last term in Equation 1, c_{bc}, results from the evaluation of β-pairs in the target sequence (assigned from the template, based on the alignment). Considering $\tilde{p}(bp_{tgt})$ as the estimation of the probability of the β-pair bp_{tgt} being formed, it is reasonable that the cost $c(bp_{tgt})$ should be proportional to $-\tilde{p}(bp_{tgt})$. The value of the cost should be large enough to allow the overall cost function to escape from misleading minima obtained with the standard scoring system. Thus, the cost should also be proportional to the number of residues involved in the pairing (n_{bp}). The following quadratic formula gave the best stable performance:

$$c_{ev_{abs}}(bp_{tgt}) = \begin{cases} +(0.5 - \tilde{p}(bp_{tgt}))^2 \cdot n_{bp} & \text{if } \tilde{p} \leq 0.5 \\ -(0.5 - \tilde{p}(bp_{tgt}))^2 \cdot n_{bp} & \text{if } \tilde{p} > 0.5 \end{cases} \quad (5)$$

An additional term has been included in order to stabilize the algorithm in the presence of wrong estimations. This contribution also takes account of the corresponding β-pair in the template (bp_{tpl}), for which the estimation error is known to be $1 - \tilde{p}(bp_{tpl})$. The requirement for this term derives from the assumption that, with the correct alignment, the errors in the estimation of p on the template and target are correlated. Therefore, if $\tilde{p}$ for the template is significantly larger than for the target, we have a strong indicator of a probable error in the alignment. With $\tilde{p}(bp_{tpl}) - \tilde{p}(bp_{tgt}$ denoted by $\delta_{\tilde{p}}$, this relative contribution is given by:

$$c_{ev_{rel}}(bp_{tgt}, bp_{tpl}) = \begin{cases} \delta_{\tilde{p}}^2 \cdot n_{bp} & \text{if } \delta_{\tilde{p}} > 0.1 \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

The total cost for a single β-pair is then:

$$c_{ev}(bp_{tgt}, bp_{tpl}) = \gamma_{abs} \cdot c_{ev_{abs}} + \gamma_{rel} \cdot c_{ev_{rel}} \quad (7)$$

where γ_{abs} and γ_{rel} are given as parameters (note that in NOBCalign, γ_{abs} and γ_{rel} are set to zero.) The total cost for an alignment is given by the sum of c_{ev}, for all the β-pairs resulting from the assignment of the template structure (S_{tpl}) to the target (S_{tgt}):

$$c_{bc}(A, S_{tpl}) = \sum_{\substack{bp_{tpl} \in S_{tpl} \\ bp_{tgt} \in \tilde{S}_{tgt}}} c_{ev}(bp_{tgt}, bp_{tpl}) \quad (8)$$

A.2 Minimizing the Cost Function In Detail

A search algorithm is completely defined by:

- *The search problem.* This is defined by the tuple: *(S_0, operator-set, goal-test, f)*,
  where S_0 is the start state; *operator-set* defines the set of states that can be reached from a given state; *goal-test* can say whether a given state is the goal or not; f is an evaluation function which gives a score (or cost) for a given path (sequence of states).
- *The search strategy.* This determines the order in which the nodes are expanded. For instance, a best-first strategy always expands a node with the best value of f.

With a global-search algorithm, the best alignment can be found by searching for the path in a tree which optimizes f, leading to the end of the sequences.

Using a blind (brute-force) search strategy, many nodes are expanded unnecessarily before finding the solution; this may lead to an explosion in computational cost. The number of expanded nodes is greatly reduced by adopting a heuristic search algorithm such as A* where the path cost of a given node n is the sum of two terms:

$$f(n) = g(n) + h(n) \quad (9)$$

g being a path cost function, and h being a heuristic function, expected to estimate the cost from that node to the solution. If the cost increases monotonically along the path and the heuristic function is "admissible" (i.e. it is an underestimation of the real cost of the solution) the A* algorithm is guaranteed to find the path with minimum cost. The complexity becomes linear if the estimation given by the heuristic function is exact.

Using the cost function defined in Equation 1 as $g(n)$, a heuristic function can be devised to estimate exactly the first two terms of the cost, but an exact estimate cannot be given for $c_{bc}(A, S_{tpl})$; the complexity rapidly increases with the length of the sequences to be aligned. In order to control the run time, rather than dynamically apply the evaluator during the search, an iterative procedure has been used: at the *i-th* iteration, the pairings found in all $i-1$ previous iterations are evaluated, until a reasonable trade-off is found.

Figure 10 presents the pseudo code for the iterative procedure. The function *bc_search* performs a search, applying the costs for the pairs obtained for the β-pairs encountered in the solutions of previous iterations. The iterations continue until convergence, or until an iteration limit is reached.

The A* evaluation function has been used in the alignment search algorithm, with the Iterative-Deepening A* (IDA*) search strategy. In our case, a search state is indicated by $S = [i, j]$, where i and j represent the relative position in the sequences, and can also be viewed as coordinates in a Needleman and Wunsch-like cost matrix. The heuristic search problem is given by:

- S_0 **(start state):** $[0, 0]$
- **goal-test:** $[i_0, j_0], i_0 = length(P_1) \wedge j_0 = length(P_2)$
- **operators:** $[i_0, j_0] \rightarrow \{[i_0 + 1, j_0 + 1], [i_0 + 1, j_0], [i_0, j_0 + 1]\}$ (substitution, insertion, and deletion respectively). Each operation is defined provided that $i \leq length(P_1) \vee j \leq length(P_2)$.
- g **(cost function):** depends on the path from the start state to the current state. It includes the costs

```
def bc_align(template_seq, target_seq, template_str):
   c_pairs = set()
   for MAX_ITERATIONS times:
      solution = bc_search(template_seq, target_seq, c_pairs)
      new_pairs = c_pairs + get_c_pairs(solution, template_str)
      if len(new_pairs) > len(c_pairs):
        c_pairs = new_pairs
      else: #no new pair found
        break
   return solution
```

Fig. 10. The BCAlign iterative algorithm pseudo-code, in Python-like syntax

for the substitutions and gaps along the path, and the costs arising from β-pair evaluations. The cost function is basically c as defined in Equation 1, but for performance reasons, the values are rounded to the nearest integer.

- h **(heuristic function):** the heuristic function is the estimated cost for the remaining part of the sequences from the current state. A matrix H_{P_1,P_2} gives the estimation: $h([i,j]) = H_{P_1,P_2}(i,j)$ and is constructed similarly to a Needleman and Wunsch matrix, minimizing the sum of the terms c_{nw} and $c_{\beta i}$.

REFERENCES

[1] K. Arnold, L. Bordoli, J. Kopp, and T. Schwede. The SWISS-MODEL workspace: A web-based environment for protein structure homology modelling. *Bioinformatics*, 22:195–201, 2006.

[2] P. A. Bates and M. J. Sternberg. Model building by comparison at CASP3: using expert knowledge and computer automation. *Proteins: Struct., Funct., Genet.*, 37:47–54, 1999.

[3] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, B. A. Rapp, and D. L. Wheeler. GenBank. *Nucleic Acid Research*, 30:17–20, 2002.

[4] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acid Research*, 28:235–242, 2000.

[5] Tom L. Blundell, Devon Carney, Stephen Gardner, Fiona Hayes, Brendan Howlin, Tim Hubbard, John Overington, Diljeet Athwal Singh, B. Lynn Sibanda, and Michael J. Sutcliffe. Knowledge-based protein modelling and design. *Eur. J. Biochem.*, 172:513–520, 1988.

[6] Jianlin Cheng and Pierre Baldi. Improved residue contact prediction using support vector machines and a large feature set. *BMC Bioinformatics*, 8:113–113, 2007.

[7] S. Cristobal, A. Zemla, D. Fischer, L. Rychlewski, and A. Elofsson. A study of quality measures for protein threading models. *Bioinformatics*, 2:5–5, 2001.

[8] Mark A. DePristo, Paul I. W. De Bakker, Reshma P. Shetty, and Tom L. Blundell. Discrete restraint-based protein modeling and the Calpha-trace problem. *Protein Sci*, 12:2032–2046, 2003.

[9] Robert C. Edgar. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, 5:113–113, 2004.

[10] A. Fiser, R. K. Do, and A. Šali. Modeling of loops in protein structures. *Protein Sci.*, 9:1753–1773, 2000.

[11] H. M. Fooks, A. C. R. Martin, D. N. Woolfson, R. B. Sessions, and E. G. Hutchinson. Amino acid pairing preferences in parallel beta-sheets in proteins. *J. Mol. Biol.*, 356:32–44, 2006.

[12] J. F. Gibrat, T. Madej, and S. H. Bryant. Surprising similarities in structure comparison. *Curr. Opin. Struct. Biol.*, 266:540–553, 1996.

[13] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100 –107, 1968.

[14] L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.*, 233:123–138, 1993.

[15] E. G. Hutchinson, R. B. Sessions, J. M. Thornton, and D. N. Woolfson. Determinants of strand register in antiparallel beta-sheets of proteins. *Protein Sci*, 7:2287–2300, 1998.

[16] W. Kabsch and C. Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

[17] Takeshi Kawabata. MATRAS: A program for protein 3D structure comparison. *Nucleic Acid Research*, 31:3367–3369, 2003.

[18] R.E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27(1):97–109, 1985.

[19] E. Krissinel and K. Henrick. Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallogr.*, 60:2256–2268, 2004.

[20] Christophe Lambert, Nadia Léonard, Xavier De Bolle, and Eric Depiereux. ESyPred3D: Prediction of proteins 3D structures. *Bioinformatics*, 18:1250–1256, 2002.

[21] F. Ledda, L. Milanesi, and E. Vargiu. Game: A generic architecture based on multiple experts for predicting protein structures. *International Journal Communications of SIWN*, 3:107–112, 2008.

[22] S. Lifson and C. Sander. Specific recognition in the tertiary structure of beta-sheets of proteins. *J. Mol. Biol.*, 139:627–639, 1980.

[23] Marco Lippi and Paolo Frasconi. Prediction of protein -residue contacts by Markov logic networks with grounding-specific weights. *Bioinformatics*, 25(18):2326–2333, 2009.

[24] A. C. R. Martin, M. W. MacArthur, and J. M. Thornton. Assessment of comparative modeling in CASP2. *Proteins: Struct., Funct., Genet.*, Suppl. 1:14–28, 1997.

[25] A.D. McLachlan. Rapid comparison of protein structres. *Acta Cryst*, A38:871–873, 1982.

[26] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

[27] M. Novotny, D. Madsen, and G. J. Kleywegt. Evaluation of protein fold comparison servers. *Proteins: Struct., Funct., Genet.*, 54:260–270, 2004.

[28] K. Ogata and H. Umeyama. An automatic homology modeling method consisting of database searches and simulated annealing. *J. Mol. Graph.*, 18:305–306, 2000.

[29] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton. CATH–a hierarchic classification of protein domain structures. *Structure*, 5:1093–1108, 1997.

[30] Frances Pearl, Annabel Todd, Ian Sillitoe, Mark Dibley, Oliver Redfern, Tony Lewis, Christopher Bennett, Russell Marsden, Alistair Grant, David Lee, Adrian Akpor, Michael Maibaum, Andrew Harrison, Timothy Dallman, Gabrielle Reeves, Ilhem Diboun, Sarah Addou, Stefano Lise, Caroline Johnston, Antonio Sillero, Janet Thornton, and Christine Orengo. The CATH Domain Structure Database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis. *Nucleic Acid Research*, 33:D247–D251, 2005.

[31] M. C. Peitsch. ProMod and Swiss-Model: Internet-based tools for automated comparative modelling. *Biochem. Soc. Trans. (London)*, 24:274–279, 1996.

[32] R. Sánchez and A. Šali. Evaluation of comparative protein

structure modeling by MODELLER-3. *Proteins: Struct., Funct., Genet.*, 1:50–58, 1997.
[33] M. A. S. Saqi, R. B. Russell, and M. J. E. Sternberg. Misleading local sequence alignments: implications for comparative modelling. *Protein Eng.*, 11:627–630, 1998.
[34] I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Eng.*, 11:739–747, 1998.
[35] R. F. Smith and T. F. Smith. Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Eng.*, 5:35–41, 1992.
[36] S. Subbiah, D. V. Laurents, and M. Levitt. Structural similarity of DNA-binding domains of bacteriophage repressors and the globin core. *Curr. Biol.*, 3:141–148, 1993.
[37] M. J. Sutcliffe, I. Haneef, D. Carney, and T. L. Blundell. Knowledge based modelling of homologous proteins. 1. Three-dimensional frameworks derived from simultaneous superposition of multiple structures. *Protein Eng.*, 1:377–384, 1987.
[38] M. J. Sutcliffe, F. R. F. Hayes, and T. L. Blundell. Knowledge based modelling of homologous proteins. 2. Rules for the conformations of sustituted side chains. *Protein Eng.*, 1:385–392, 1987.
[39] W. R. Taylor and C. A. Orengo. Protein structure alignment. *J. Mol. Biol.*, 208:1–22, 1989.
[40] Allison N. Tegge, Zheng Wang, Jesse Eickholt, and Jianlin Cheng. NNcon: improved protein contact map prediction using 2D-recursive neural networks. *Nucleic Acids Res*, 37:W515–W518, 2009.
[41] A. Šali and T. L. Blundell. Comparative protein modelling by satisfaction of spatial restraints. *J. Mol. Biol.*, 234:779–815, 1993.
[42] M. A. Wouters and P. M. Curmi. An analysis of side chain interactions and pair correlations within antiparallel beta-sheets: the differences between backbone hydrogen-bonded and non-hydrogen-bonded residue pairs. *Proteins*, 22:119–131, 1995.

Giuliano Armano obtained his Ph.D. degree in Electronic Engineering from the University of Genoa, Italy, in 1990. He is currently associate professor of computer engineering at the University of Cagliari, and Head of the "Intelligent Agents and Soft-Computing" group. His educational background ranges from expert systems to machine learning, whereas his current research activities focus on soft-computing architectures and systems, in particular for bioinformatics and information retrieval tasks. As for bioinformatics, his primary research interests are on protein secondary structure prediction and on protein encoding based on multiple alignment.

Filippo Ledda was awarded a degree in electronic engineering from the University of Cagliari in 2006. From 2008 to 2011 he has been Ph.D. student in electronic engineering and computer science at the University of Cagliari, where he has currently a post doc. position. His current research interests are focused on the development of machine learning algorithms for ensemble architectures applied to bioinformatics. Within this field he has extensively investigated the problems of protein secondary structure prediction and protein encoding based on multiple alignment.

Andrew C. R. Martin obtained his PhD degree in Biochemistry at Oxford, in 1990. He joined Janet Thornton's group at UCL in 1994 as a post doc. for 4 years, working primarily on analysis and modelling of protein loops and the CATH database. From May 2005 he is Senior Lecturer in Bioinformatics, at University College London. His current research activities focus on structural bioinformatics and are aimed at developing tools to investigate and understand the relationship between protein sequence, structure and function. Within this general area, his main interests are protein modelling, structural analysis, structural immunology, effects of mutation on protein structure and disease, application of relational databases, automation and software development, with a particular interest in the sequence and structure of antibodies.