Methods in Molecular Biology™

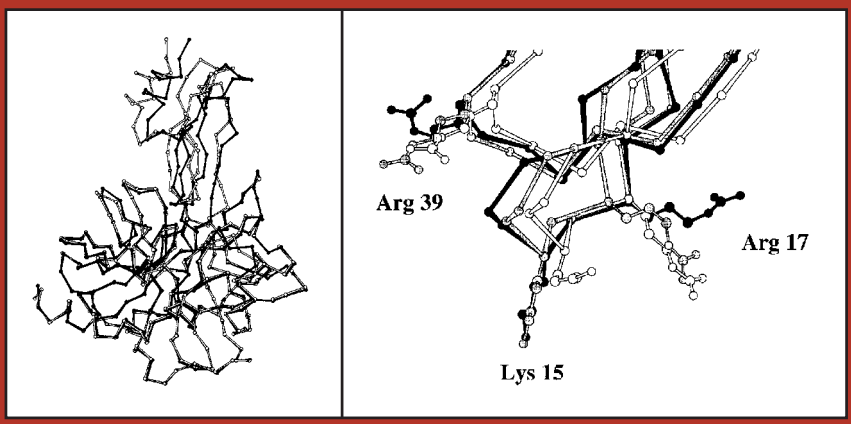
VOLUME 143

Protein Structure Prediction

Methods and Protocols

Edited by

David M. Webster



HUMANA PRESS

1

Multiple Sequence Alignment

Desmond G. Higgins and William R. Taylor

1. Introduction

The alignment of protein sequences is the most powerful computational tool available to the molecular biologist. Where one sequence is of unknown structure and function, its alignment with another sequence that is well characterized in both structure and function immediately reveals the structure and function of the first sequence. This ideal transfer of information is, unfortunately, not always attained and can fail either because the two sequences are equally uncharacterized (although they might align quite well) or because the alignment is too poor to be trusted. Both these situations can be helped if the analysis is extended to incorporate more sequences. In the former case, the addition of further sequences can reveal portions of the protein that are important in structure and function (even if that structure or function is unknown), whereas in the latter, the revelation of conserved patterns can help add confidence in the alignment.

In this chapter, we describe two methods that can be used to produce multiple sequence alignments. Both are based on the simple heuristic that it is best to align the most similar sequences first and gradually combine these, in a hierarchic manner, into a multiple sequence alignment.

2. MULTAL

2.1. Outline of the Algorithm

The Program MULTAL was originally devised to deal with large numbers of protein sequences that are typically encountered in the analysis of large families (such as the immunogobulins or globins) or in sifting out the often extensive collections of sequences produced as the result of a search across the

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

sequence databanks. These applications are the main topic considered in this section. Those who wish to use the program only as an alignment/editor for a small number of sequences would be best to seek out the program CAMELON <http://www.oxmol.co.uk/prods/camelon/> (which is an implementation of MULTAL by Oxford Molecular) or CLUSTAL (*see* **Subheading 3.**).

Where CLUSTAL takes a more rigorous phylogenetic approach to ordering of sequences prior to alignment, MULTAL uses a simple single-linked clustering iterated over several cycles. On each cycle, only sequences that have a pairwise similarity greater than a predefined cutoff (specified of each cycle) are aligned. If more than two sequences are mutually similar above the current cutoff score, then all are brought together in one step using a fast concatenation algorithm (*see* **ref. *1***). However, as this is only robust for closely related sequences, later cycles are restricted to pairwise combinations.

In each cycle, all subalignments and all single sequences are again compared with each other. Here the algorithm differs significantly from CLUSTAL, which adheres to the original guide tree and is more similar to the GCG program PILEUP (http://www.gcg.com/products/software.html) that developed out of a simpler approach (2). When aligning a sequence with an alignment or an alignment with an alignment, MULTAL calculates a pairwise sum over the similarity of each amino acid in one alignment with each amino acid in the other alignment. MULTAL retains this simple sum, whereas CLUSTAL provides a weighting scheme to down-weight the contribution from similar sequences. This feature was not provided in MULTAL, as the alternate approach (which is more practical with large numbers of sequences) is simply to remove one of a pair of similar sequences. A protocol for this is described as follows.

2.2. Strategies for Large Numbers of Sequences

MULTAL contains numerous methods to deal with large numbers of sequence (where large is considered to be hundreds or thousands of sequences). Although very valuable, this aspect can require understanding and careful treatment if the program is not to miss expected similarities. Generally, there is a trade-off between time spent and the chance of missing a relationship.

2.2.1. The Span Parameter

The greatest saving in time that can be made when dealing with a large number of sequences is to avoid the costly comparison of all against all (this is especially true for MULTAL, where this calculation is performed on each cycle). If the sequences were presented in an optimal order in which the most similar sequences were adjacent, then MULTAL would only need to consider adjacent sequences on each cycle — transforming a time dependency that was

proportional to the square of the sequences into a time dependency that is linear in the number of sequences. As such an optimal order cannot easily be obtained, MULTAL considers the pairwise similarity over a number of adjacent sequences, specified by a parameter called the **span**, which can be varied from cycle to cycle, as can all the MULTAL parameters.

In general, the span starts small (comparing only local sequences) and expands from cycle to cycle. However, even if it remains fixed at a small number, there is still a good chance of obtaining a complete multiple alignment, because, as the cycles progress, the number of "sequences" (which now includes subalignments) decreases relative to the span so that by the final cycles, the number of subalignments plus unaligned sequences (referred to jointly as *blocks*) is less than the span and so all are eventually compared to all.

2.2.2. The Window Parameter

A related saving can be made at the level of the detailed calculation of the alignment. If the initial cycles are only aligning relatively similar sequences, then the size of relative insertion and deletion needed to obtain the optimal alignment can be expected to be relatively small. If restrictions are placed on the alignment path, then a calculation of time dependent on the product of the sequences becomes approximately linear in sequence length. The parameter that controls this is called the **window** and its value specifies a diagonal stripe (placed symmetrically) through the matrix (dot-plot) constructed from placing each sequence on the sides of a rectangle. As a safeguard, however, if the difference in sequence length is greater than the size of the window parameter value, then the sequences are not compared on that cycle. In general (as with the span parameter), the value of the window parameter should be increased through successive cycles.

2.2.3. Peptide Presort

The efficient operation of both the span and window parameters rely on having a well-ordered starting list of sequences. Often, sequences are fouund preordered in existing databanks or as the result of a previous alignment using MULTAL or some other program. (Both MULTAL and CLUSTAL record the resulting alignment to be used in this way.) However, if this is not avaliable, then MULTAL can (optionally) attempt to create it based on a rough measure of similarity based on an analysis of the peptide composition of each sequence — specifically, the number of common peptides between sequences. This can be calculated very quickly using a simple hash-table or as in the current versions of MULTAL, using a dynamic radix tree structure that can accommodate any peptide size. The size of peptide that is used for this analysis can be specified but, in general, less than three is too general and over four is too specific

(too few common peptides are found in all but the most similar sequences). Originally, a tetrapeptide was used ***(3)*** and it was also shown ***(4)*** that a tripeptide measure can capture sequence similarity quite well down to roughly the level of 50% identity.

2.3. Alignment Parameters

As in all alignment methods, it is necessary to specify a measure of similarity between amino acids to provide an alignment score and, in addition, specify both a model and parameters for the penalty attached to relative insertions and deletions (gaps). As in other aspects of MULTAL, these aspects are kept very simple as it is the general philosophy of the approach that the important contribution to the alignment is the number and quality of the sequences (with respect to their phylogenetic distribution) that makes a good alignment and not the fine tuning of parameters. For example, if a good selection of sequences are obtained, then these effectively define their own local amino acid exchange matrix at every position.

2.3.1. Amino Acid Exchange Matrix

MULTAL allows two matrices to be used in each run and these can be combined in varying proportions on each cycle. Generally, the two matrices used are the identity matrix (in which amino acid identites score 10 and all else 0) and the PAM_{120} matrix ***(5)***. These are stored in the files id.mat and md.mat but can be substituted for any other matrix, e.g., Dayhoff's PAM_{250} matrix, a BLOSUM matrix ***(15)***, or even the JTT matrix ***(4)***. Through the different cycles, the current matrix is a linear interpolation between the two given matrices, specified by the parameter matrix that gives the porportion (out of 10) that the matrix in md.mat contributes. For example, if matrix = 3, then (with the PAM_{120} matrix in md.mat), the values used in the alignment calculation are 30% of the PAM_{120} values augmented by 7 on the diagonal (being 70% of the values in the identity matrix in id.mat). The same overall effect might have been attained by using a series of PAM or BLOSUM matrices (as can be used in the CLUSTAL program), however, the fine specification of values makes little difference to the alignment and the use of an identity matrix produces values that are more familiar.

In the past, the matrix parameter was increased from cycle to cycle, with the expectation that later alignments would be composed of more distant sequences and should therefore have a matrix suited to their degree of divergence (e.g., the PAM_{250} matrix). However, although this is still true for isolated sequences that have not aligned, it does not apply to subalignments, as these have already effectively created their own individual amino acid exchange matrix at every position composed out of the sum of amino acid pairwise similarities. This

effect combined with a "soft" matrix (one that scores general similarity) leads to too much flexibility in the match and tends to diminish the importance of highly conserved positions (of which there are often relatively few) and can lead to both misalignment and the false incorporation of sequences that do not belong in the family.

2.3.2. Gap Penalties

Adhering to the philosophy that the simplest alignment principles are sufficient, MULTAL has only one gap penalty that is paid once for a gap of any size — but not at the beginning or end of a sequence. This is justified in the context of the alignment of distant protein sequences by the expectation (1) that the locations where insertions can occur in the protein structure are generally on the surface and (2) that if a small insertion can be made, there are probably few constraints on this forming a linker out to a larger insertion that might even comprise a complete domain. As with the matrix parameter, the gap penalty can be varied over the cycles, but little justification has been seen for this and, generally a constant gap value in the range 20–30 is maintained over the full run.

Some later and more experimental versions of MULTAL embody more complex gap functions. These were designed to take account of the structural expectation that matches in a sequences alignment are correlated, often being found in runs (typical of a conserved secondary structure) ***(6,7)***, or having an overall distribution that cannot be adequately controlled by a penalty applied independently at each insertion point ***(8)***. These more subtle aspects have also been reviewed in a less technical volume ***(9)***.

2.4. When to Stop Aligning

Programs such as MULTAL or CLUSTAL (or any of their ilk) contain no inherent method to detect when two sequences (or subalignments) should not be aligned together. The various algorithms can produce an alignment even when the sequences are random. Rough guidelines, such as percentage sequence identity can be used, or statistics such as those employed in databank search methods. However, there are no adequate statistics that can be applied to the more complex situation of aligning alignments. Even the percentage identity is not a good guide as the pairwise similarity among sequences that can be reliably aligned using multiple sequence alignment methods extends far into what would be considered random were the two sequences to be extracted and assessed as a pair. These scores are also directly derived from the current matrix and gap penalty, which is also difficult to allow for.

Strategies, that can be employed with MULTAL are to allow the alignment to go to completion (one big family) but then to backtrack up the cycles (using careful visual assessment) until the point at which the subfamilies last seemed

Table 1
MULTAL Parameter Files for Alignment

Matrix	Gap	Span	Win.	Cutoff
5	20	3	30	700
5	20	5	40	600
5	20	7	50	500
5	20	9	60	400
5	20	9	70	300
5	20	9	80	250
5	20	9	90	200
5	20	9	100	150
5	20	9	100	150

The columns are, respectively, the *matrix* parameter (5 = 50% PAM_{120}), the *gap* penalty, the number of adjacent sequences considered (*span*), boundary (window) on alignment deviation (*win.*), and the score *cutoff*. Each line of parameters is used in successive cycles. (*See* and **ref. *3*** for details.)

to be credible. This places considerable burden on the method used for "visual assessment" and in the absence of any structural or functional knowledge, this can only be judged by the conservation of groups that might be involved in structure or function. The former are generally interesting residues, such as arginine, aspartate, histidine, or any charged amino acid that might be capable of catalysis or binding. The residues of structural importance are generally hydrophobic, with glycine, proline, and cysteine often conserved because of their unique properties.

Visual assessment cannot be employed in automatic family compilation or where the user has little "feel" for the data. In this situation, it has been found (through accumulated experience) that with a matrix value of 3 and a gap penalty of 20–30, the recommended lower limit on the score cutoff is 150. At this level, in repeated trials, there are roughly as many family members that do not align as there are false alignments. A value of 200 or 250 would be recommended as a safer choice for those who have little or no feel for the quality of sequence alignments (*see* **Table 1** for an example of parameter file).

2.5. Sequence Selection with MULTAL

2.5.1. Sequence Criteria

Sequences can be selected using the program MULTAL as a prefilter to form subfamilies above a preset degree of similarity (details in **Tables 1** and **2**). From each subfamily, a representative sequence was chosen according to the weighting scheme that valued sequences with a respresentative length that did not contain any nonstandard amino acids. A measure *r* was calculated:

Table 2
MULTAL Parameter Files for Filtering

Matrix	Gap	Span	Win.	Cutoff
		(A) Filter to 90%		
0	20	1	1	990
0	20	2	1	980
0	20	4	2	960
0	20	8	3	940
0	20	10	4	920
0	20	10	5	900
0	20	10	5	900
		(B) Filter to 80%		
Matrix	Gap	Span	Win.	Cutoff
0	20	1	5	890
0	20	2	6	880
0	20	4	7	860
0	20	8	8	840
0	20	10	9	820
0	20	10	10	800
0	20	10	10	800
		(C) Filter to 70%		
Matrix	Gap	Span	Win.	Cutoff
0	20	1	10	790
0	20	2	12	780
0	20	4	14	760
0	20	8	16	740
0	20	10	18	720
0	20	10	20	700
0	20	10	20	700

The columns are, respectively, the *matrix* parameter (0 = identity), the *gap* penalty, the number of adjacent sequences considered (*span*), boundary (window) on alignment deviation (*win.*), and the score *cutoff*. Each line of parameters is used in successive cycles. (*See* above and **ref. *3*** for details.)

$$r = \log(d^2 + 1) + s \quad (1)$$

where d is the difference in length of an individual sequence from the mean length of the subfamily in which it is aligned and s is the number of nonstandard amino acid symbols (included, B J O U X Z). To this basic score, penalties and bonus points were added as defined in **Table 3** and the sequence with the lowest score was selected.

Table 3
Structure Selection Penalties

Attribute	Penalty
MODEL	999
NMR	5
MUTANT	2
FRAGMENT	1

If the protein description contained the *attribute* key word, the *penalty* was added.

Table 4
Sequence Selection Penalties

Attribute	Penalty
PROBABLE	1
PRECURSOR	2
HYPOTHETICAL	5
MUTANT	40
FRAGMENT	50
Special	–100
Structure	–60

If the description line contained the *attribute* key word (in capitals) the *penalty* was added to the base score r (**Eq. 1**). The bonus points (below the line) were added if the sequence has some special significance (determined by the used), or had a known structure.

The sequences can be filtered (using the foregoing criteria) in successive cycles, first to eliminate any sequences with more than 90% similarity, then 80%, and finally 70% similarity. (*See* **Table 2** for alignment parameter details.)

2.5.2. Structural Criteria

A set of protein structures can be filtered using the same approach but with a different set of criteria. With this data, the base score (r) was taken as the atomic resolution plus the average B-value over the α-carbons divided by 100. If the resolution was not defined a value of 5 was taken and similarly an undefined B-value contribution was taken as 1 (i.e., an average of 100/residue). Onto this base score were added the penalties and bonus scores defined in **Table 4**.

2.6. Installation and Operation

2.6.1. Installation

MULTAL can be downloaded by ftp from <http://mathbio.nimr.mrc.ac.uk/>. It is currently implemented on Silicon Graphics computers (SIG, Mountain View, CA), but the source code (which is in standard C language) is provided and can be easily recompiled on other machines. Note that this version is the user-unfriendly version for use by acedemics. Commercial companies and those who need a friendly interface or user support should contact Oxford Molecular (Web site <http://www.oxmol.co.uk/prods/cameleon/>) to investigate purchasing CAMELEON.

1. In the internet location <http://mathbio.nimr.mrc.ac.uk/>, click on the MULTAL-FTP name to go to the MULTAL directory. Here, two files will be found: README.txt and multal.tar.gz.
2. Click on MULTAL.tar.gz and provide a local directory name into which it can be copied.
3. Unpack the file in the local directory by typing gunzip -c multal.tar.gz | tar xvof - . This will create a directory called MULTAL containing the program and a subdirectory data containing some amino acid similarity matrices.
4. MULTAL can be run simply by typing multas. All parameters and sequences are specified in the file called test.run, of which an example is provided along with some test sequences. The sequence selection version (which differs only in its output) is called MULSEL.

2.6.2. Operation

A good example on which to test MULTAL is the small β/α protein flavodoxin. These bacterial proteins are widely diverged, having large insertions and deletions, but they still retain some relatively clear motifs by which to judge the quality of the alignment. This is aided in the test sequences provided (in the flavo.seq), which have been edited to include a lowercase residue in the motifs that should align. In the final alignment these lowercase letters should be aligned. It is a useful exercise to vary the matrix, gap penalty, and number of sequences to get a feel for the effect that these variables have on the accuracy of the alignment. The sequence file contains 13 sequences (with three of the known structure from which the motif alignment can be checked) and the start of the default run is shown in **Fig. 1**.

In **Fig. 1** the names and lengths of the input sequences are echoed, along with the parameters for the first cycle. Following this, a top-triangle matrix of scores is presented for all the pairwise comparisons. Here, sequence paris outside the range of the span parameter (3) are not calculated, and this is indicated by the entry >s. Similarly, those not calculated because of the length difference

```
Mutation Data Matrix (120 PAMs)
ARNDCQEGHILKMFPSTWYVBZX
matrix constant = 8
Residue identity scores 10
ARNDCQEGHILKMFPSTWYVBZX
matrix constant = 0

55 sequences to be read in length range 1 -> 1000
Reading user file of seq.s: flavo.seq
 user_in = 1
1 USER>1FX1 147 2.00 flavodoxin - Desulfovibrio vulgaris
2 USER>2FCR 173 1.80 flavodoxin - red alga (Chondrus crispus)
3 USER>4FXN 138 1.80 flavodoxin (semiquinone form) - Clostridium sp.
4 USER>FLAV_ANASP 169 FLAVODOXIN. - ANABAENA SP. (STRAIN PCC 7120), AND ANABAENA SP.
5 USER>FLAV_AZOVI 179 FLAVODOXIN. - AZOTOBACTER VINELANDII.
6 USER>FLAV_CLOAB 160 FLAVODOXIN. - CLOSTRIDIUM ACETOBUTYLICUM.
7 USER>FLAV_DESDE 148 FLAVODOXIN. - DESULFOVIBRIO DESULFURICANS.
8 USER>FLAV_DESGI 146 FLAVODOXIN. - DESULFOVIBRIO GIGAS.
9 USER>FLAV_DESSA 146 FLAVODOXIN. - DESULFOVIBRIO SALEXIGENS.
10 USER>FLAV_DESVH 148 FLAVODOXIN. - DESULFOVIBRIO VULGARIS (STRAIN HILDENBOROUGH).
11 USER>FLAV_ECOLI 175 FLAVODOXIN. - ESCHERICHIA COLI.
12 USER>FLAV_ENTAG 177 FLAVODOXIN. - ENTEROBACTER AGGLOMERANS.
13 USER>FLAV_MEGEL 137 FLAVODOXIN. - MEGASPHAERA ELSDENII.
End of user sequences
13 sequences read into 13 blocks, (13 are keyed)


PARAMETERS for next cycle
 mul_wt = 0 damp = 0 mat_wt = 5
 gap_pen = 20 span = 3 window = 30
 minscore = 700 pepscore = 0 list_limit = 2
 circles = 0 output = 2 outline = 80
score_pairs

    0=block 479 519 500  >s  >s  >s  >s  >s  >s  >s  >s  >s
        1=block  >w 631 620  >s  >s  >s  >s  >s  >s  >s  >s
            2=block  >w  >w 480  >s  >s  >s  >s  >s  >s  >s
                3=block 717 462 445  >s  >s  >s  >s  >s  >s
                    4=block 447  >w  >w  >s  >s  >s  >s  >s
                        5=block 447 421 445  >s  >s  >s  >s
                            6=block 720 700 713  >s  >s  >s
                                7=block 808 797 368  >s  >s
                                    8=block 792 379  >w  >s
                                        9=block 446 447 485
                                           10=block 638  >w
                                               11=block  >w
                                                   12=block
average score over cutoff = 749
(7 pairs) cluster_pairs total list_alloc = 6
pack_lists
update_blocks
      USER>FLAV_DESGI
808
      USER>FLAV_DESSA


      USER>FLAV_ANASP
717
      USER>FLAV_AZOVI


      USER>FLAV_DESDE
713
      USER>FLAV_DESVH


block 3 = 2 seqs
*USER>FLAV_ANASP :      FLAVODOXIN. - ANABAENA SP. (STRAIN PCC 7120), AND ANABAENA SP.
*USER>FLAV_AZOVI :      FLAVODOXIN. - AZOTOBACTER VINELANDII.

SKKIGLFYGTQTGKTESVaEIIRDEFGNDVVT LHDVSQAEVTDLNDYQYLIIgCPTWNIGELQS        DWEGLYS
 AKIGLFFGSNTGKTRKVaKSIKKRFDDETMSDALNVNNRVSAEDFAQYQFLILgTPTLGEGELPGLSSDCENESWEEFLP

ELDDVDFNGKLVAYfGTGDQIGYADNFQDAIGILEEKISQRgGKTVGYWSTDGYDFNDSKALRNGKFVGLALDEDNQSDL
KIEGLDFSGKTVALfGLGDQVGYPENYLDALGELYSFFKDRgAKIVGSWSTDGYEFESSEAVVDGKFVGLALDLDNQSGK

TDDRIKSwVAQLKSEFGL
TDERVAAwLAQIAPEFGLSL
```

Fig. 1. Initial text output from MULTAL comparing 13 flavodoxin sequences.

condition (window) are indicated as <w. The highest scoring pairs of sequences are selected and the alignment of two of these is shown at the bottom of **Fig. 1**.

This process is repeated through each cycle until, at the final cycle, all the sequences have aligned (the final alignment scores more than the 150 cutoff in test.run). The final result is shown in **Fig. 2**, in which the two current subalignments are brought together with a score of 389. The crude "graphic" (of "p-b---"s) is a (fallen) treelike record of the order in which the sequences were brought together. For example, the three pairs aligned in the first cycle (**Fig. 1**) are bridged by a p-b- graphic on the part closest to the sequence codes, whereas further condensations progress progress to the left. The parameters producing this result (in which all motifs align) are shown in **Table 1**, which is an amplification of the file test.run. (Details of the options can be found in the README.txt file on the Web server.)

2.6.3. Execution Time

Using the test sequences provided in the flavo.seq (along with the parameters provided in test.run), the time taken to align the sequences was measured when running on a single Silicon Graphics R10000 processor (174 MHz) by typing the command time multas > /dev/null. The times returned by the UNIX time utility were 0.825u 0.072s 0:01.10 80.9% 0+ok 14+3io 5pf+0w; this specifies under one second in the user field (u).

3. CLUSTAL

CLUSTAL is the generic name for a family of programs that have been produced to carry out multiple alignments since 1988 ***(10–13)***. The most recent versions are CLUSTAL W ***(12)***, which uses a simple text menu interface, and CLUSTAL X ***(13)***, which uses a portable windowing system. Both programs are freely available for academic use and may also be used from within some of the main sequence analysis packages as well as from a number of sites on the Internet. The algorithmic details for the two programs are more or less identical, but CLUSTAL X does have some extra features for selecting subsets of sequences for realignment and for viewing misaligned regions. It also looks nicer and provides the user with multicolored alignments.

The basic method is similar to that of MULTAL (**ref. *3*** and **Section 2**). Each pair of sequences is aligned in turn and the similarity of the sequences is recorded as the percent identity between them, ignoring any positions with gaps. These scores are used to build an approximate phylogenetic tree between the sequences using the Neighbour–Joining method ***(14)***. These trees are referred to here as dendrograms (structures that indicate similarity in a hierarchical manner between a set of objects but do not necessarily indicate phylogenetic relatedness). Finally, the multiple alignment is built up gradually by

```
     ......p--- USER>1FX1
     ....p-b-p- USER>FLAV_DESDE
     ....|...b- USER>FLAV_DESVH
     ..p-b---p- USER>FLAV_DESGI
     ..|.....b- USER>FLAV_DESSA
     ..b-p----- USER>4FXN
     ....b----- USER>FLAV_MEGEL
389
     ..p------- USER>2FCR
     ..|.....p- USER>FLAV_ANASP
     ..|...p-b- USER>FLAV_AZOVI
     p-b-p-b--- USER>FLAV_ECOLI
     |...b----- USER>FLAV_ENTAG
     b--------- USER>FLAV_CLOAB


block 0 = 13 seqs
*USER>1FX1 :    2.00 flavodoxin - Desulfovibrio vulgaris
*USER>FLAV_DESDE :      FLAVODOXIN. - DESULFOVIBRIO DESULFURICANS.
*USER>FLAV_DESVH :      FLAVODOXIN. - DESULFOVIBRIO VULGARIS (STRAIN HILDENBOROUGH).
*USER>FLAV_DESGI :      FLAVODOXIN. - DESULFOVIBRIO GIGAS.
*USER>FLAV_DESSA :      FLAVODOXIN. - DESULFOVIBRIO SALEXIGENS.
*USER>4FXN :    1.80 flavodoxin (semiquinone form) - Clostridium sp.
*USER>FLAV_MEGEL :      FLAVODOXIN. - MEGASPHAERA ELSDENII.
*USER>2FCR :    1.80 flavodoxin - red alga (Chondrus crispus)
*USER>FLAV_ANASP :      FLAVODOXIN. - ANABAENA SP. (STRAIN PCC 7120), AND ANABAENA SP.
*USER>FLAV_AZOVI :      FLAVODOXIN. - AZOTOBACTER VINELANDII.
*USER>FLAV_ECOLI :      FLAVODOXIN. - ESCHERICHIA COLI.
*USER>FLAV_ENTAG :      FLAVODOXIN. - ENTEROBACTER AGGLOMERANS.
*USER>FLAV_CLOAB :      FLAVODOXIN. - CLOSTRIDIUM ACETOBUTYLICUM.

 PKALIVYGSTTGNTEYTaETIARQLANAGYEVDSRDAASVEAGGLFEGFDLVLLgCSTWGDDSIELQD        DFI
MSKVLIVFGSSTGNTESIaQKLEELIAAGGHEVTLLNAADASAENLADGYDAVLFgCSAWGMEDLEMQD        DFL
MPKALIVYGSTTGNTEYTaETIARELADAGYEVDSRDAASVEAGGLFEGFDLVLLgCSTWGDDSIELQD        DFI
MPKALIVYGSTTGNTEGVaEAIAKTLNSEGMETTVVNVADVTAPGLAEGYDVVLLgCSTWGDDEIELQE        DFV
MSKSLIVYGSTTGNTETAaEYVAEAFENKEIDVELKNVTDVSVADLGNGYDIVLFgCSTWGEEEIELQD        DFI
   MKIVYWSGTGNTEKMaELIAKGIIESGKDVNTINVSDVNIDELLNE DILILgCSAMGDEVLE  E        SEF
  MVEIVYWSGTGNTEAMaNEIEAAVKAAGADVESVRFEDTNVDDVASK DVILLgCPAMGSEELE  D        SVV
  KIGIFFSTSTGNTTEVaDFIGK  TLGAKADAPIDVDDVTDPQALKDYDLLFLgAPTWNTGADTERSGT    SWDEF
SKKIGLFYGTQTGKTESVaEIIRD  EFGNDVVT LHDVSQAEVTDLNDYQYLIIgCPTWNIGELQS       DWEGL
 AKIGLFFGSNTGKTRKVaKSIKK  RFDDETMSDALNVNRVSAEDFAQYQFLILgTPTLGEGELPGLSSDCENESWEEF
 AITGIFFGSDTGNTENIaKMIQK  QLGKDV ADVHDIAKSSKEDLEAYDILLLgIPTWYYGEAQC        DWDDF
MATIGIFFGSDTGQTRKVaKLIHQ  KLDGIADA PLDVRRATREQFLSYPVLLLgTPTLGDGELPGVEAGSQYDSWQEF
 MKISILYSSKTGKTERVaKLIEEGVKRSGNIEVKTMNLDAVDKKFLQESEGIIFgTPTYYANI           SWEMK

PLFDSLEETGAQGRKVACfGCGDSSY  EYFCGAVDAIEEKLKNLgAEIVQD                       GLRID
SLFEEFNRFGLAGRKVAAfASGDQEY  EHFCGAVPAIEERAKELgATIIAE                       GLKME
PLFDSLEETGAQGRKVACfGCGDSSY  EYFCGAVDAIEEKLKNLgAEIVQD                       GLRID
PLYEDLDRAGLKDKKVGVfGCGDSSY  TYFCGAVDVIEKKAEELgATLVAS                       SLKID
PLYDSLENADLKGKKVSVfGCGDSDY  TYFCGAVDAIEEKLEKMgAVVIGD                       SLKID
EPFIEEISTKISGKKVALfGSYGWGD  GKW   MRDFEERMNGYgCVVVET                       PLIVQ
EPFFTDLAPKLKGKKVGLfGSYGWGS  GEW   MDAWKQRTEDTgATVIGT                       A IVN
L YDKLPEVDMKDLPVAIfGLGDAEGYPDNFCDAIEEIHDCFAKQgAKPVGFSNPDDDYDYEESKSVRDGK FLGLPLDMV
  YSELDDVDFNGKLVAYfGTGDQIGYADNFQDAIGILEEKISQRgGKTVGYWSTDGYDFNDSKALRNGK FVGLALDED
  LPKIEGLDFSGKTVALfGLGDQVGYPENYLDALGELYSFFKDRgAKIVGSWSTDGYEFESSEAVVDGK FVGLALDLD
  FPTLEEIDFNGKLVALfGCGDQEDYAEYFCDALGTIRDIIEPRgATIVGHWPTAGYHFEASKGLADDDHFVGLAIDED
  TNTLSEADLTGKTVALfGLGDQLNYSKNFVSAMRILYDLVIARgACVVGNWPREGYKFSFSAALLENNEFVGLPLDQE
KWIDESSEFNLEGKLGAAfSTANSIAGGSDI  ALLTILNHLMVKgMLVYS    GGVAFGKPKTHLGYVHINEIQENED

GDPRAARDDIVGwAHDVRGAI
GDASNDPEAVASfAEDVLKQL
GDPRAARDDIVGwAHDVRGAI
GEP  DSAEVLDwAREVLARV
GDP  ERDEIVSwGSGIADKI
NEPDEAEQDCIEfGKKIANI
EMPDNA PECKE1GEAAAKA
NDQIPMEKRVAGwVEAVVSETGV
NQSDLTDDRIKSwVAQLKSEFGL
NQSGKTDERVAAwLAQIAPEFGLSL
RQPELTAERVEKwVKQISEELHLDEILNA
NQYDLTEERIDSwLEKLKPAVL
ENARIfGERiANkVKQIF
```

Fig. 2. Final text output from MULTAL aligning 13 flavodoxin sequences.

aligning together larger and larger groups of sequences, following the branching order in the dendrogram, with the most similar sequences being aligned first.

3.1. Basic Multiple Alignment

The sequences to be aligned must be collected together in one file. These can be in any of seven different file formats, all of which are recognized and read automatically by the program. These formats are NBRF/PIR, EMBL/SWISSPROT, Pearson (Fasta), CLUSTAL (*.aln), GCG/MSF (Pileup), GCG9/RSF, and GDE flat file. All nonalphabetic characters (spaces, digits, punctuation marks) are ignored except ”-” which is used to indicate a GAP (”.” in GCG/MSF). A complete alignment may be input to the program for further analysis such as the calculation of a phylogenetic tree. Sequence input is carried out by requesting the appropriate item from the menus and the user will enter the name of the file to be read. The file will be checked for sequence type (amino acid or nucleic acid) and number and lengths of the sequences.

If there is no error on input, the sequences will be kept in memory awaiting alignment. In CLUSTAL X, the sequences are displayed on the screen as they were read in and the user may then scroll though them. Multiple alignment is carried out by going to the Multiple alignment menu where the first option is Do complete multiple alignment now. Selecting this option will trigger requests file names for the complete alignment (the original file name with the characters .aln appended or as a replacement for an existing file extension name) and for the dendrogram file (the same file name but ending in .dnd instead of .aln). The complete alignment process is then carried out automatically and the intermediate results are displayed on the screen to help monitor progress. The scores (percent identity) of each initial pairwise alignment are displayed as they are calculated, and then the scores of each intermediate alignment in the final alignment are displayed along with the numbers of sequences being aligned at each stage. If any sequences are particularly distant from the remaining set of sequences, the alignment of these may be delayed until all of the more easily aligned sequences are dealt with and a message is posted on the screen.

With CLUSTAL W, the complete alignment is displayed on the screen, one page at a time, using three different symbols to indicate conservation in each column of the alignment: “*” for complete conservation (identity), “:” for a strongly conserved column (conserved amino acid type) and “.” for a weakly conserved position. A user-modifiable coloring scheme is used with CLUSTAL X to indicate conservation in each column. Furthermore, CLUSTAL X can detect and display alignment positions and sections of sequence that appear to be badly aligned (relative to the rest of the sequences). This is particularly useful in detecting scrambled sections of proteins, perhaps due to DNA

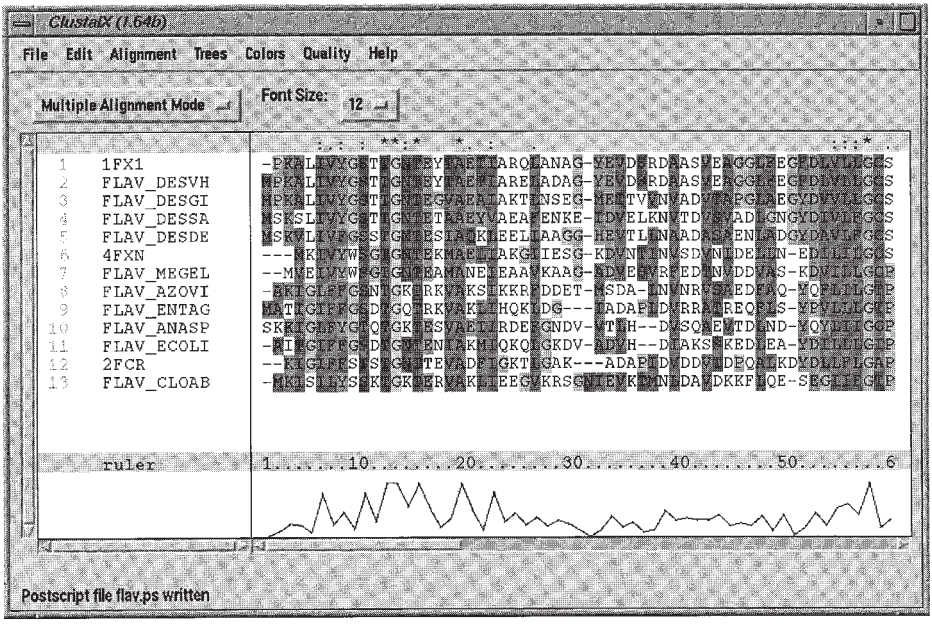


Fig. 3. CLUSTAL X display of aligned flavodoxin sequences.

sequencing errors that cause frameshifts in the translated amino acid sequence (*see* **Fig. 3**).

If the user has many sequences to align, the initial all against all comparisons may become very time consuming. This can be helped by adjusting the parameters (*see* **Subheading 3.2.**) or the user may use an old dendrogram file (file names ending in .dnd), providing it applies to exactly the same sequences (the same sequence names and the same number of sequences). Similarly, users can request the dendrogram file only. Dendrograms are written in the New Hampshire/nested parentheses format and can be viewed using tree display software and can, in principle, be modified in order to change the order in which sequences are aligned. The latter is a complex task, however, without appropriate tools for modifying trees unless the tree is very simple.

3.2. Changing Alignment Parameters

There are many parameters that can be used to control the alignments. There are two sets, one for the initial pairwise alignments and a second for the final multiple alignments. Under normal circumstances, it will make little difference to change the pairwise parameters. The only measurable effect will be to change the branching order in the dendrogram, and hence the order of sequence alignment in the final stages. There is one useful parameter, however, that can

be used to carry out the initial alignments using full dynamic programming or using a much faster but less accurate method. This parameter can be set using a clearly marked item in the multiple alignment menu. For small numbers (e.g., less than 30 or so) of small sequences (e.g., 200 or so residues each), this parameter will have little effect, but for large numbers of long sequences there can be a huge saving in time by using the faster method.

The multiple alignment parameters may be changed in a submenu of the multiple alignment menu. The main parameters are the two gap penalties (the gap-opening penalty, which gives the cost of opening a new gap, and the gap-extension penalty, which gives the cost of extending a gap) and the amino acid weight matrix. Terminal gaps are not penalized. Default values are given for these, but the user is free to select alternatives. The situation is complicated because the initial values selected from the menu will be modified depending on the weight matrix chosen, the similarity of the sequences, and the sequence lengths. The gap penalties are also varied along each sequence or prealigned set of sequences, depending on the local occurrence of existing gaps or certain residue types. Nonetheless, the overall occurrence of gaps may be easily controlled by setting the two gap parameters. The user can choose to have fewer gaps (increase the gap opening penalty) or shorter gaps (increase the gap extension penalty) overall.

The scores given to various aligned pairs of amino acids are controlled by the use of amino acid weight matrices. In principle, these give a score for each possible pair of residues and are balanced by the gap penalties. In practice, it is more complicated, as there are now several sets of matrices available and each usually consists of a series of matrices suitable for sequences of different degrees of divergence. Some matrices are ideal for very similar sequences where most weight is given to identical pairs of residues. Other matrices are better suited for distantly related sequences where much weight is given to residues with similar biochemical properties (e.g., hydrophobic residues, aromatic residues, positively charged residues). By default, the software uses the BLOSUM series of tables from Jorja and Steven Henikoff ***(15)*** and uses four different ones, depending on the divergence of the sequences to be aligned. These matrices are changed automatically by the software as the alignment progresses. Two alternative series of matrices are offered and the user can enter their own if they have a matrix in the format used by the BLAST program. In MULTAL, weight matrices are also adjusted for sequence divergence but in a different way (*see* **Section 2**).

Gaps do not occur with equal frequency in all parts of protein alignments. They are rare in the main secondary structure elements of alpha helices and beta strands and more frequent in loops and non-core regions. CLUSTAL attempts to mirror this by making gaps more or less likely along alignments.

This is controlled by a series of protein gap parameters, which are set from the multiple alignment parameters menu. First, the user can use a series of weights that are associated with each of the 20 amino acids, which make gaps more or less likely adjacent to certain columns of residues. These weights are empirically derived from the observed frequencies of gaps in structurally based protein alignments. Columns with conserved glycines are more likley to have gaps beside them than columns in alignments that are rich in valine, for example. Second, the user can choose to make gaps more likely beside short runs of hydrophilic residues. These runs are usually in exposed loop regions. The length of these runs and the residues that are considered to be hydrophilic may both be set from the menu.

Of the remaining parameters, the most important is that marked as "use negative matrix" in the menu. By default, all amino acid weight matrix values are set to being positive, regardless of whether or not they contain negative values. This has the effect of making all alignment regions, even completely misaligned ones, score positively. Occasionally, fragments or sequences with large N-terminal or C-terminal overhangs, will be misaligned because of this. It is worth checking the ends of alignments for serious mismatches and changing this parameter. It is difficult to make settings for alignments that will automatically work well with both full-length sequences and mixtures of full-length sequences and fragments.

3.3. Phylogenetic Trees

The dendrograms that are used to decide the branching order of the alignments may be viewed using appropriate tree-viewing software (e.g., NJPLOT or TREEVIEW). These are not normally used as real phylogenetic trees, although they may give a reasonable approximation. The dendrograms are approximate because the pairwise distances are derived from separately aligned sequences rather than a complete multiple alignment, which is expected to be more accurate and because the distance measure that is used is simple percent distance rather than an evolutionary distance. The Neighbour–Joining method ***(14)*** is used because it is fast and gives accurate trees in a wide variety of situations. There are more sophisticated and/or more accurate methods available, many of which are available in alternative packages and users are encouraged to explore these. The Neighbour–Joining method works by taking all pairwise distances between the sequences and attempting to fit these to a tree topology using an iterative least-squares procedure. It produces unrooted trees with branch lengths for each branch in the tree.

Before trees are calculated, the sequences must already be aligned. If not, the tree topology will be roughly star like with all sequences very distant from each other. The alignment can be read in from an existing alignment that the

user has carried out previously and that is stored in a file. Alignments can be read in a variety of formats, including CLUSTAL ".aln" files. Alternatively, if the user has just carried out a multiple alignment, the alignment will still be in memory and trees can be calculated. If an appropriate alignment is in memory, a phylogenetic tree can be requested from the phylogenetic tree menu. Here there is a menu item that will produce a tree in one step after the user is prompted for the name of the file to contain the tree (by default, these files end in .ph). There are no facilities in CLUSTAL for displaying the trees graphically. User must take the tree file (*.ph) and use a tree-drawing program such as TREEVIEW or NJPLOT to view them.

There are two parameters that users can set from the menus and that are used to help control the production of the trees. First, users may request that all gap positions be removed from the alignment. This means that any positions in the multiple alignment that contain gaps in any sequence will be ignored. This is wasteful of data in that sites are removed, even if just one sequence is not represented at any position. This is not appropriate when fragments of sequences are used for this reason. It does have the benefit, however, of removing the most difficult alignment areas (the sections of alignment that are most ambiguous) automatically as these tend to cluster around gap positions. It also means that all calculations are carried out on exactly the same positions in all sequences.

The second parameter allows users to use a correction for multiple hits. The pairwise distances are initially calculated as mean numbers of observed differences per position. These distances are roughly percent differences divided by 100. With closely related sequences, these distances will approximate the number of substitutions per site that have occurred between each pair. For more distantly related sequences, however, these distances will greatly underestimate the actual numbers of substitutions and the user can then use this option to try and correct for this. The correction is based on the model of protein evolution by Margaret Dayhoff and coworkers ***(5)***. This model is the same one that was also used to produce the famous PAM series of amino acid weight matrices. It has the effect of taking distances and stretching them, especially with large distances that can be stretched several fold.

Finally, users may request bootstrap confidence measures for each grouping in the tree. This involves making a series of trees from randomized alignments and comparing the original tree with this set of bootstrap pseudoreplicate trees. The measures are expressed as percentages and can crudely be used as measures of confidence. The precise interpretation of these figures in a statistical sense is the subject of ongoing debate, but they do give very useful indications of stability and reliability in the trees. Informally, any groupings that occur in more than 90% of the pseudoreplicates is often considered strongly supported

by the data, given the method used to make the tree. It does not prove biological significance. Strong bootstrap support for incorrect groupings may be obtained with highly biased data and poor or inappropriate methods.

References

1. Taylor, W. R. (1990) Hierarchical method to align large numbers of biological sequences, in *Methods in Enzymology*, vol. 183, *Molecular Evolution: Computer Analysis of Protein and Nucleaic Acid Sequences* (Doolittle, R. F., ed.), Academic, San Diego, CA, pp. 456–474.
2. Feng, D. F. and Doolittle, R. F. (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.* **25(4),** 351–360.
3. Taylor, W. R. (1988) A flexible method to align large numbers of biological sequences. *J. Mol. Evol.* **28,** 161–169.
4. Jones, D. T., Taylor, W. R., and Thornton, J. M. (1992) The rapid generation of mutation data matrices from protein sequences. *CABIOS* **8,** 275–282.
5. Dayhoff, M. O., Schwartz, R. M., and Orcutt, B. C. (1978) A model of evolutionary change in proteins, in *Atlas of Protein Sequence and Structure*, vol. 5, suppl. 3, National Biomedical Research Foundation, Washington DC, pp. 345–352.
6. Taylor, W. R. (1994) Motif-based protein sequence alignment. *J. Comp. Biol.* **1,** 297–311.
7. Taylor, W. R. (1995) An investigation of conservation-biased gap-penalties for multiple protein sequence alignment. *Gene* **165,** GC27–GC35. Internet journal Gene Combis: http://www.elsevier.nl/locate/genecombis
8. Taylor, W. R. (1996) A non-local gap-penalty for profile alignment. *Bull. Math. Biol.* **58,** 1–18.
9. Taylor, W. R. (1996) Multiple protein sequence alignemnt: algorithms for gap insertion, in *Methods in Enzymology*, vol. 266, *Computer Methods for Macromolecular Sequence Analysis* (Doolittle, R. F., ed.), Academic, San Diego, FL, pp. 343–367.
10. Higgins, D. G. and Sharp, P. M. (1988) Clustal: a package for performing multiple sequence alignment of a microcomputer. *Gene* **73,** 237–244.
11. Higgins, D. G., Bleasby, A. J., and Fuchs, R. (1992) Clustal V: improved software for multiple sequence alignment. *CABIOS* **8,** 189–191.
12. 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.
13. Thompson, J. D., Gibson, T. J., Plewniak, F., Jeanmougin, F., and Higgins, D. G. (1997) Clustal-X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Res.* **25,** 4876–4882.
14. Saitou, N. and Nei, M. (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Eval.* **4,** 406–425.
15. Henikoff, S. and Henikoff, J. G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA* **89,** 10,915–10,919.

2

Protein Structure Comparison Using SAP

William R. Taylor

1. Introduction

In contrast to DNA, proteins exhibit an apparently unlimited variety of structure. This is a necessary requirement of the vast array of differing functions that they perform in the maintainance of life, again, in contrast to the relatively static archival function of DNA. Not only do we observe a bewildering variety of form but even within a common structure, there is variation in the lengths and orientation substructures. Such variation is both a reflection on the very long time periods over which some structures have diverged and also a consequence of the fact that proteins cannot be completely rigid bodies but must have flexibility to accommodate the structural changes that are almost always necessary for them to perform their functions. These aspects make comparing structure and finding structural similarity over long divergence times very difficult. Indeed, computationally, the problem of recognizing similarity is one of three-dimensional pattern recognition, which is a notoriously difficult problem for computers to perform. In this chapter, guidance is provided on the use of a flexible structure comparison method that overcomes many of the problems of comparing protein structures that may exhibit only weak similarity.

1.1. Structural Hierarchy

The aspect of protein structure that makes the comparison problem inherently tractable, is that protein structure is organized in a hierarchy of structural levels, beginning with the basic unit of an amino acid, short stretches of these can adopt one of two semiregular local structures referred to as α and β, being, respectively, helical and extended in nature. The simplicity of having only two secondary-structures (as they are jointly known) is that there are only three (pairwise) combinations of them that can be used to construct proteins, thus

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

giving the three major structural classes: (1) α with α, (2) α with β, and (3) β with β. Various attempts have been made to order and classify the proteins within these groups. One early attempt called a Structural Classification of Proteins (SCOP), is based mainly on visual assessment (`http://scop.mrc-lmb.cam.ac.uk/scop/`), whereas a later classification, called CATH, is based on a more automatic classification, using an earlier version of the program to be described in this chapter. CATH, which stands for the four major levels in this hierarchy — Class, Architecture, Topology (fold family), and Homologous superfamily — also contains a considerable degree of expert added information (`http://www.biochem.ucl.ac.uk/bsm/cath/`). The third main classification is Dali, which is more oriented toward searching for structural similarity using a fast, but rough, similarity method (`http://www2.ebi.ac.uk/dali/`). The resulting similarities are ordered by a variety of measures but it is sometimes difficult to draw the line between true and chance

1.1.1. All-α Proteins

The all-α protein class is dominated by small folds, many of which form a simple bundle with helices running up and down. The interactions between helices are not discrete (in the way that hydrogen bonds in a β-sheet are either there or not), which makes their classification more difficult. Set against this, however, the size of the α-helix (which is generally larger than a β-strand) gives more interatomic contacts with its neighbors (relative to the a β-strand), allowing interactions to be more clearly defined.

1.1.2. All-β Proteins

The all-β proteins are often classified by the number of β-sheets in the structure and the number and direction of β-strands in the sheet. This leads to a fairly rigid classification scheme that can be sensitive to the exact definition of hydrogen-bonds and β-strands. Because they are less rigid than an α-helix, the β-sheets in two proteins can be relatively distorted — often with differing degrees of twist of fragmented or extra strands on the edges of the sheet — making comparisons difficult.

1.1.3. α–β Proteins

The α–β protein class can be subdivided roughly into proteins that exhibit a mainly alternating arrangement of α-helix and β-strands along the sequence and those that have more segregated secondary-structures. The former class includes some large and very regular arrangements of structure (in which a central β-sheet formed of parallel β-strands is covered on both sides by α-helices. Often it is not clear whether this dominance is an evolutionary relic

or simply a stable (and so favored) arrangement of secondary-structures. If the latter, then any evolutionary implications based on finding similar substructures must be weak.

1.2. Comparison Methods

The simplest approach to compare two proteins is to move the coordinate set of one structure (as a rigid body) over the other and look for equivalent atoms. This can only be done easily for relatively similar structures and any large scale movement of equivalent substructure can quickly obscure similarities.To avoid this problem, one structure can be broken into fragments; however, this can lead to a series of local comparisons in which the overall global "picture" might be missed.

Both global and local aspects are important and were combined in a number of approaches that used local environments (or views) of the structure to produce an overall equivalence ***(1,2)***. These methods determine an alignment of one protein sequence on the other (but based on structure not generic sequence similarity) that may then be used as a set of equivalences to produce a three-dimensional superposition of the structural coordinate sets. Both methods embody the constraint that the structures maintain a linear equivalence, and although this is usually a firm basis for evolutionary relationship, other methods can identify similarity without this constraint. The constraint of the linear ordering of structure is sometimes neglected simply for computational convenience but sometimes through a specific wish to find non topological relationships in structures ***(3)***. Although these might elucidate structural principles — such as the mode of packing of an α-helix on a β-sheet (regardless of the β-strand ordering in the sheet) — their application to problems of evolutionary relationships would not be recommended. A major use for such methods, however, is in the identification of local arrangements of groups that constitute an active site or binding pocket, which might well have arisen independently. One of these algorithms based on a geometric hashing algorithm ***(1)*** is shown in **Fig. 1**.

1.3. Statistical Significance

The statistical significance of structure comparison results is not easily assessed. This is largely because there is no simple model of a random protein (in the same way that random sequences can be simply generated). The approach often taken (e.g., in Dali), is to generate a "random" background distribution from miss-hits on other proteins in the protein structure databank. This suffers from the problem that some of this background might contain unrecognized nonrandom similarities. However, it is a reasonable assumption to assume that these are relatively few.

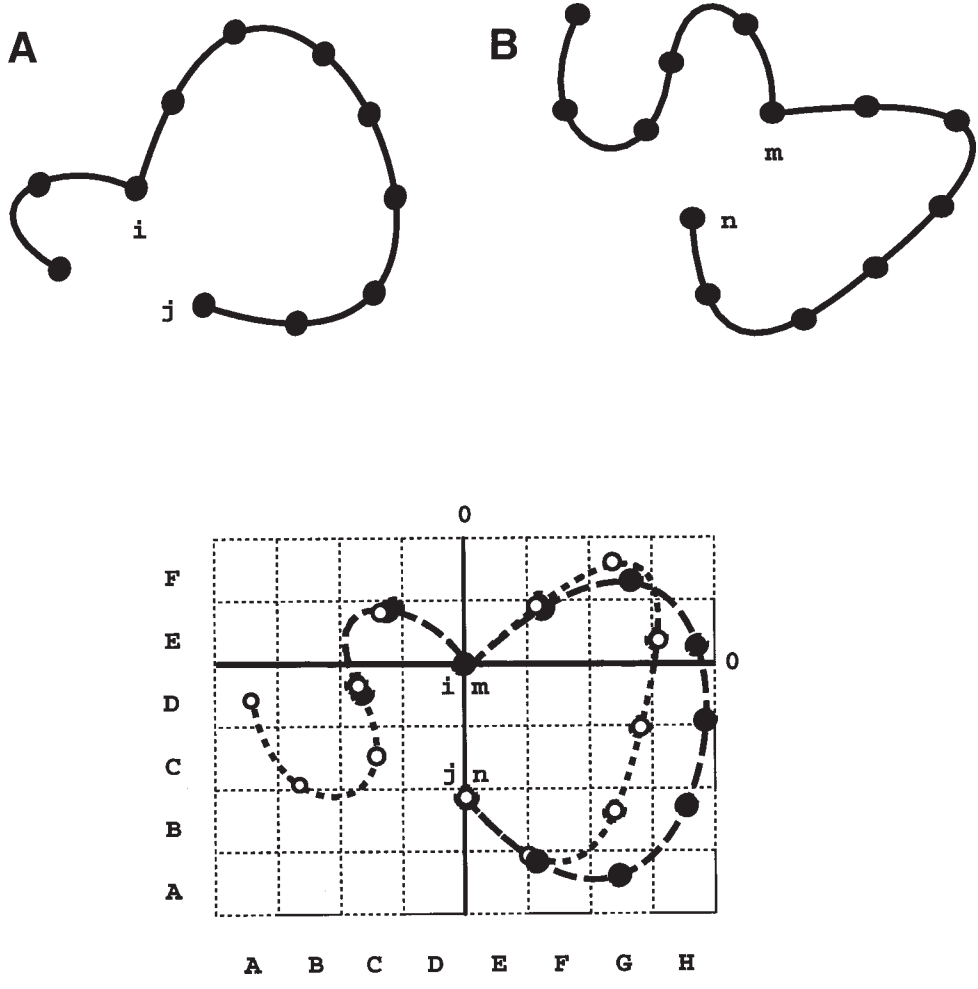


Fig. 1. Geometric hashing algorithm. Two protein structures **(A)** and **(B)** are shown schematically. Two pairs of positions (*i*, *j*) in **(A)** and *m*, *n* in **(B)** are selected. Both structures are centered on the origin of a grid **(C)** at *i* and *m* and orientated by placing a second atom in each structure (*j* and *n*) on the vertical axis which is (coincidentally) the terminal atom of each structure. (In three dimensions, three atoms are required to define a unique orientation.) Atoms in both structures (open and filled circles) are assigned an identifier that is unique to the cell in which they lie (the *hash* key). For simplicity, this is shown as the concatenation of two letters associatedwith the ordinate with the abscissa (XY). For example, atoms in structure **(B)** are assigned identifiers AD, BC, CC, CD, etc. The number of common identifiers between the structures provides a score of similarity. In this example, these are CD, CE, FE, GF, HE, and FA (not counting *i*, *j* and *m*, *n*) giving a score of 6. The process is repeated for all pairs of pairs, or in three dimensions, all triples of triples and the results pooled.

When one is dealing only with unconnected secondary-structure segments, better theoretical distributions can be deduced, allowing very fast filtering of potentially significant similarities ***(5)***. This is the basis underlying the vector alignment search tool (VAST) structure comparison and search method (`http://www.ncbi.nlm.nih.gov/Structure/VAST.vast.html`).

A hybrid approach adopted in the program SAP (described in **Subheading 2.**) in which the protein structure is reversed to form a random model (as this program only uses α-carbons, the secondary-structure remains virtually unaltered under reversal). Further variation is generated by random reconnections of secondary-structure and randomization in the selection phase of the comparison algorithm ***(6)***.

2. SAP

The program described here is called SAP (for Structure Alignment Program) and was derived from a related program SSAP, which forms the basis of the CATH classification and was one of the earlier methods based on the use of a local structural view to make an alignment ***(1,7)***. The current version is largely a simplification of its predecessor but is also based on a refined iterative algorithm.

2.1. Structure Alignment Algorithm

The core comparison algorithm underlying both SAP (as well as SSAP, and also some sequence/structure comparison methods ***[8,9]***) is based on the same algorithm as is used to compare protein sequences ***(10)***. As such, insertions and deletions can be easily incorporated, allowing the full range of variation that would be expected between distantly related proteins. When comparing just sequences, one amino acid is (from the point of view of the algorithm) just like any other amino acid of the same type, and as such can be assigned a generic score when matched up (aligned) with another residue. This is not the situation in structure comparison where an amino acid in the core of the protein is fundamentally different from an amino acid on the surface of the protein — even if they are the same amino acid type. This difference in situation can be embodied in a measure of the local structural environment of each residue that can then form the basis of a similarity measure between positions and so allow an alignment algorithm to be applied.

2.1.1. Double Dynamic Programming

The simplest comparison approach would be to have a measure based only on the secondary-structure state and degree of burial of the two residues in the two proteins being compared. Such a simplistic measure, however, could not distinguish two adjacent β-strands, both of which were buried in the core of

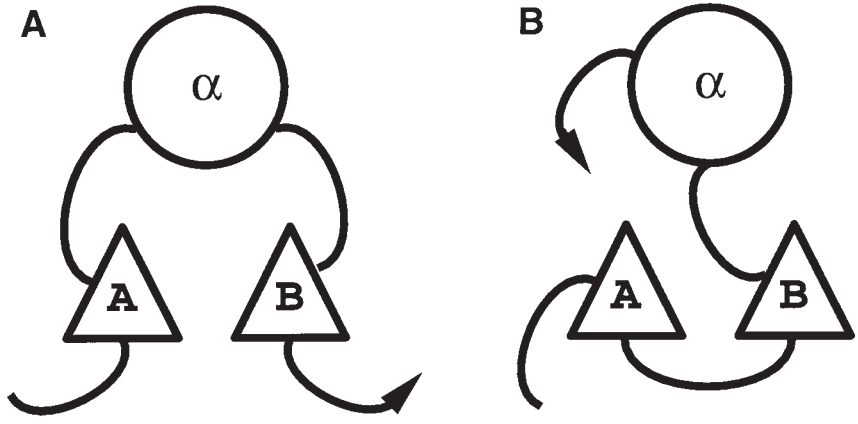


Fig. 2. Two β-strands, A and B, are shown schematically as triangles packing against an α-helix (circle) in two distinct structural fragments, **(A)** (βαβ) and **(B)** (ββα). The packing in the two fragments could be identical but a comparison method that takes account of the topology (or connectivity) of the units would not detect any great similarity.

both proteins. For this, a description of environment is required that can capture the true three-dimensional relationship between residues (referred to as their topological relationship). This is a difficult computational problem and might best be appreciated by the following simple example. Consider two β-strands — A and B — found in both proteins being compared and lying in that order both in the sequence of the two proteins and also in their respective β-sheets. If both pack against an α-helix then, in both proteins, a point on A would be buried by a β-strand to the right and an α-helix above, and would be considered to be in similar environments. If, however, in one protein, the α-helix lay between strand A and B, while in the other protein it lay after strand B, then the two arrangements would not be topologically equivalent (**Fig. 2**).

To discount the contribution of the α-helix in the foregoing example, one must know before assessing the environments of the β-strands that the two helices are not equivalent. Were this known beforehand (for all such elements), then the comparison problem would be solved before the first step was taken. To break this circularity, the following computational device was used: given the assumption (retaining the foregoing example) that strand A in both proteins are equivalent, then how similar can their environments be made to appear while still retaining topological equivalence? If, in the foregoing example, only the B strands could be equivalenced and, consequently, the assumption that the two A strands are equivalent would not be supported strongly. If, on the other hand, the two helices were also equivalent (say both proteins had a βαβ struc-

ture), then the equivalence of the A strands would be scored more highly. These scores themselves can be calculated between all pairs of residues and taken to form the basis of a score matrix, from which the "best-of-the-best" set of equivalences can be extracted while still retaining topological equivalence.

The basic alignment (or Dynamic Programming) algorithm is applied at two distinct levels: a low level to find the best score given that residue *i* is equivalent to *j*, and at a high level to select which of all possible pairs form the best alignment. This double level (combined with the basic algorithm) gave rise to the name *"Double Dynamic Programming."* Although previously discussed in terms of secondary-structures, the algorithm operates at the level of individual residues and the environments that are compared consist of interatomic vector sets. To convey some impression of these data, a simplified (two-dimensional) example is shown in **Fig. 3**, in which the construction of the low-level matrix is demonstrated.

2.1.2. Selection and Iteration

The Double Dynamic Programming algorithm described earlier, requires a computation time proportional to the fourth power of the sequence length (for two proteins of equal length) as it performs an alignment for all residue pairs. To circumvent this severe requirement, some simple heuristics were devised based on the principle that comparing the environment of all residue pairs is not necessary. Based on local structure and environment, many residue (indeed most) pairs can be neglected. This selection is based on secondary-structure state (one would not normally want to compare an α-helix with a β-strand) and burial (those with a similar degree of burial are most similar) but a component based on the amino acid identity can also be used, giving any sequence similarity a chance to contribute.

The basic algorithm was implemented, as previously ***(11)***, in an iterative form using the heuristics on the first cycle to make a selection of potentially similar residue pairs. On subsequent cycles, the results of the comparison based on this selection are used to refine the next selection. Previously, a large number of potentially equivalent residue pairs were selected for the initial comparison, and after this only 20 were taken. In the reformulated algorithm, this trend is reversed and an initially small selection (typically 20–30) pairs are selected and gradually increased with each iteration. This initial sparse sampling can, however (just by chance), be unrepresentative of the truly equivalent pairs. To avoid this problem, continuity through the early sparse cycles was maintained in the current algorithm by using the initial rough similarity score matrix (referred to as the *bias* matrix) as a base for incremental revision. As the cycles progress, the selection of pairs becomes increasingly determined by the dominant alignment, approaching (or attaining), by the final cycle, a self-consistent

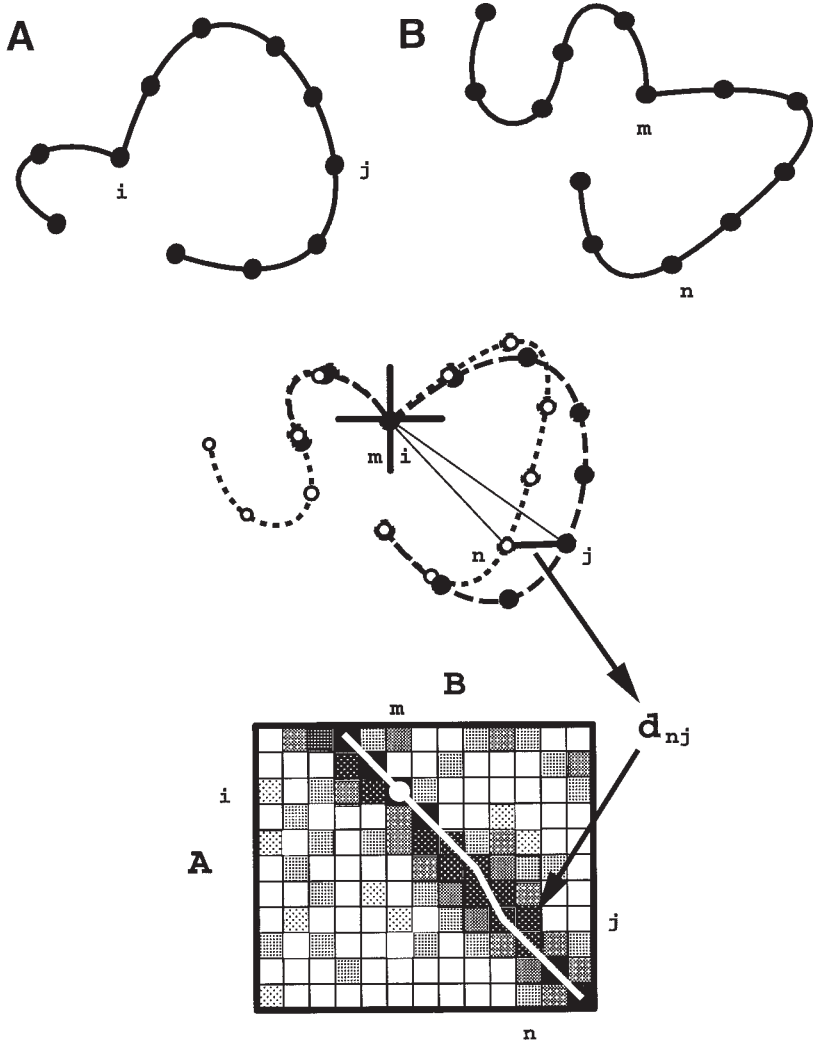


Fig. 3. Two protein structures **(A)** and **(B)** are shown schematically. A pair of positions (*i* in **(A)** and *m* in B) is selected. Both structures are centered on *i* and *m* and orientated by a local measure, such as the α-carbons geometry (indicated by the large cross). In this superposition the relationship between all pairs of atoms (e.g., *n* and *j*) is quantified, either as a simple distance (d_{nj}) or by some more complex function. All pair values are stored in a matrix and an alignment (white trace) found by the dynamic programming algorithm. The arbitrary choice of equating *i* and *m* is circumvented by repeating the process for all possible *i,m* super positions and pooling the results. In the SSAP algorithm, a final alignment is extracted from the pooled results by a second dynamic programming step.

state in which the alignment has been calculated predominantly (or completely) from pairs of residues that lie on the alignment

2.2. Multiple Structure Comparison

The problem of multiple structure comparison, which is often problematic when dealing with structure superposition falls naturally into the structure alignment algorithm described earlier. The approach follows that described for the multiple alignment of protein sequences by MULTAL (*see* Chapter 1). The only difference is that the idea of comparing a point in one subalignment with a point in another requires a measure that is more complex than simply the average pairwise amino acid similarity used in MULTAL.

In SAP the internal description of the structural environments is captured as interatomic vector sets. In a multiple version, in which the positions in two proteins have been equivalenced (say, *i* and *j*), these vector sets are combined to produce an averaged set in which the multiple vector is an average of the two (or more) contributions. Importantly, the coherence of the vector is recorded. If the combined vectors form a tight bunch, then the position is conserved and given a high weight (one for identical vectors), whereas if the vectors point in opposite directions their weight is zero ***(12)***.

The multiple version of SAP is currently being developed for use on a paralyzed Web server called PHASE (funded by the European Union Esprit program) and the state of availability can be checked at the following Web site: `http://mathbio.nimr.mrc.ac.uk/`.

2.3. Treatment of Domains

In most comparison problems, the problem of domains is avoided by dividing the proteins into different domains before any comparison is made. This is also done in the various classification databases (CATH, SCOP, etc.) and also in specialized domain databases such as DDBASE (`http://www-cryst.bioc.cam.ac.uk/~ddbase/`) or 3Dee (`http://circinus.ebi.ac.uk:8080/3Dee/help/help\_intro.html`).

The approach in SAP is to iteratively define domains as the comparison of the structures progresses. This approach is experimental and is not yet generally implemented in the publicly available program (*see* **Subheading 3.**) except for a limited facility that looks for internal domain duplication within a structure. If the same structure is presented twice to SAP, rather than return the obvious, the trivial solution (on the diagonal of the comparison matrix) is masked out in the program, and the ensuing selection of pairs with similar local structure directs the search toward off-diagonal solutions that correspond to internal duplications.

3. Installation and Operation

3.1. Installation

SAP can be downloaded by `ftp` from `http://mathbio.nimr.mrc.ac.uk/`. It is currently specific to Silicon Graphics computers.

1. In the Internet location `http://mathbio.nimr.mrc.ac.uk/`, click on the SAP–FTP name to go to the `sap` directory. Here, two files will be found: `README.txt` and `sap.tar.gz`.
2. Click on `sap.tar.gz` and provide a local directory name into which it canbe copied.
3. Unpack the file in the local directory by typing `gunzip -c sap.tar.gz | tar xvof -`. This will create a director called `sap` containing the program and a subdirectory `data` containing an amino acid similarity matrix.
4. `SAP` can be run by typing the line `sap file1.pdb file2.pdb`. The program can read the full PDB (Protein DataBank) files but needs only the α-carbons

3.2. Operation

A good example on which to test SAP is the two small β/α proteins flavodoxin and the chemotaxis-Y (PDB codes: `4fxn` and `3chy`, respectively). These two proteins have the same fold but no specific sequence similarity. After 10 cycles, SAP should find a solution in which 102 common α-carbons are equivalenced, at which point 84.21% of the selected residue pairs lay on the alignment — in other words, convergence was not complete. This is reported in the output as "`Percents sel on aln`." Of the 102 residues in the alignment, 62.75% of them had been selected as pairs for comparison (reported as "`Percent aln in sel`"). These percentages are a guide to the quality of the comparison but should not be expected to reach 100%. However, if either (or both) fall far below 50%, then caution should be exercised in the interpretation of the results.

The alignment is presented in vertical format with the numbered sequences on either side. Inserted or deleted segments are not printed; these are only apparent from breaks in the residue numbering.* Between the sequences is a numeric value that reflects the degree of similarity between the two local environments (big is more similar). Thus the similar portions are immediately apparent (having values over 100, whereas the dissimilar regions will have values below 10). These numbers are normalized and applied as weights to produce a weighted rigid body superposition of the two structures ***(13)***, for

*The numbering is the sequential numbering in the files as presented and not the attached (PDB) residue number.

```
 file1 = /pdb/brk/3chy.brk
 prot1->Compound = CHE*Y
 file2 = /pdb/brk/4fxn.brk
 prot2->Compound = FLAVODOXIN (SEMIQUINONE FORM)
Mutation Data Matrix (120 PAMs)
ARNDCQEGHILKMFPSTWYVBZX
matrix constant = 8
Cycle 1, 16 residues selected
Cycle 2, 23 residues selected
Cycle 3, 29 residues selected
Cycle 4, 36 residues selected
Cycle 5, 43 residues selected
Cycle 6, 49 residues selected
Cycle 7, 56 residues selected
Cycle 8, 63 residues selected
Cycle 9, 69 residues selected
Cycle 10, 76 residues selected

 score = 2555.236572

Percent sel on aln =   84.21
Percent aln in sel =   62.75

**M    1 105.4   7 F*
**K    2  93.4   8 L**
**I    3 120.6   9 V**
 *V    4 102.9  10 V*
 *Y    5  87.7  11 D**
 *W    6  49.6  12 D*
**T   12   3.9  13 F
 *E   13  27.7  14 S
 *K   14  20.8  15 T
**M   15  22.9  16 M
**A   16  36.4  17 R**
  E   17  23.8  18 R
 *L   18  17.6  19 I
**I   19  27.8  20 V*
 *A   20  34.8  21 R*
  K   21  17.2  22 N
  :                :
 *I  116  16.9 107 V*
 *V  117   2.2 108 K**
  Q  126   0.1 109 P
 *D  127   4.9 110 F*
 *C  128   1.5 111 T*
**I  129   6.2 112 A*
  E  130   6.2 113 A
 *F  131   7.3 114 T
**G  132  15.4 115 L*
 *K  133  14.5 116 E*
 *K  134   8.7 117 E
 *I  135   4.9 118 K*
**A  136   3.4 119 L
  N  137   5.2 120 N
  I  138   0.1 121 K
Weighted RMSd =  2.498 (over 102 atoms)
Un-weighted RMSd =  2.328 over best 43 atoms
Un-weighted RMSd =  4.068 over all matched atoms (102)
```

Fig. 4. Text output from SAP. Two small proteins were compared (`4fxn` and `3chy`). In each alignment, the sequences run vertically and the intervening numeric value is a measure of the strength of each equivalence in the alignment. Solvent exposure is also indicated as "**" = very buried and "*" = partly buried.

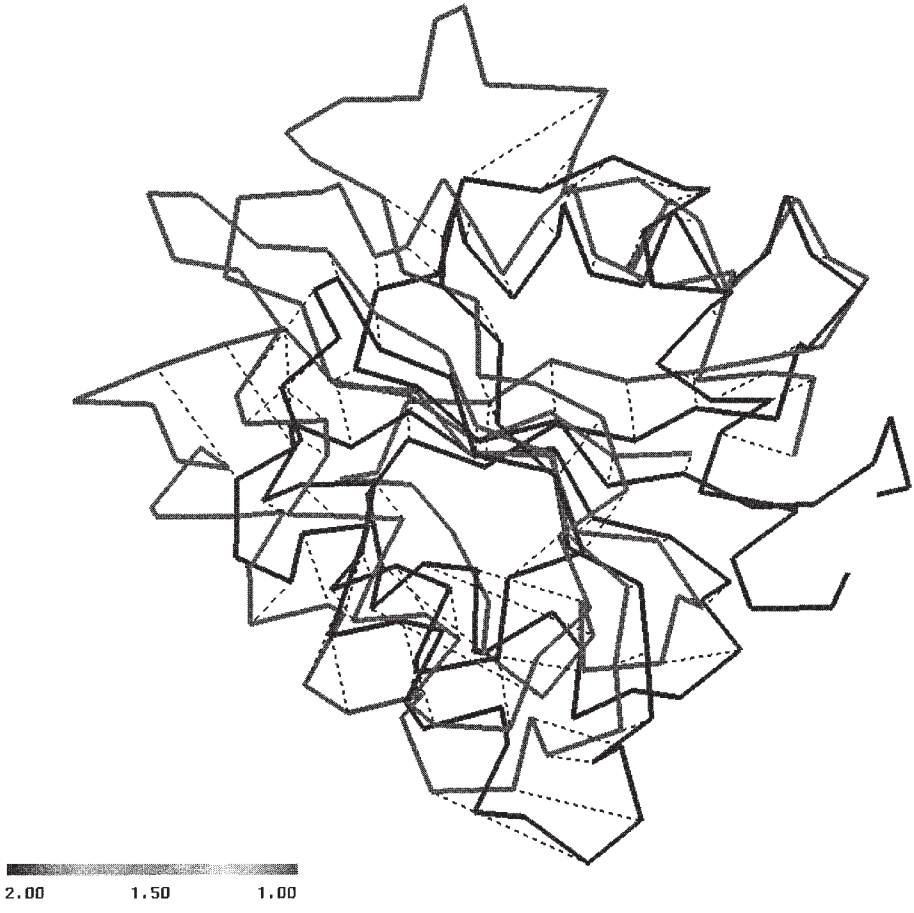


Fig. 5. PROTDRAW display of `4fxn` (gray) on `3chy` (black).The dashed lines connect residues (α-carbons positions) that have been aligned by the SAP program as described in the text.

which the root-mean-square deviation (RMSD) is quoted as "`Weighted RMSD = 2.498 (over 102 atoms)`." Two further values are quoted, which are the unweighted RMSD based on the highest scoring (most locally similar) residue pairs and an unweighted RMSD over all the matched atoms (*see* **Fig. 4**, previous page).

3.3. Visualization

SAP uses the alignment of the two structures and the local similarity values to perform a weighted rotation of one coordinate set onto the other. The result

of this transformation is saved in the file `super.pdb` (which is written with each run of the program). Only the α-carbons are saved, but the format is in standard PDB form with each structure separated by a "TER" record and the local similarity score written to the B-value field. Any visualization program (such as RASMOL) can be used to view the results, however, a simple viewing program called PROTDRAW (András Aszódi, unpublished software) is provided in the FTP-file (and should automatically appear in the local directory) (*see* **Fig. 5**). The `README.txt` file should be consulted for a full description of this program.

PROTDRAW has various options (which can be reached by pressing the right mouse button), the most useful of which is to color the structures by B-value and so illuminate their most similar regions. These appear as red with gradations through yellow and green to blue for the least similar parts of the structure. The darkest blue is reserved for unaligned portions of the structures. A second useful feature to visualize the equivalence between the structures is to connect the equivalent residues. SAP does this by writing "fake" hydrogen-bond records to the PDB file and when these are turned-on in PROTDRAW, a white dashed line links equivalent atoms

References

1. Taylor, W. R. and Orengo C. A. (1989) Protein structure alignment. *J. Mol. Biol.* **208,** 1–22.
2. Sali, A. and Blundell T. L. (1990) Definition of general topological equivalence in protein structures: a procedure involving comparison of properties and relationship through simulated annealing and dynamic programming. *J. Mol. Biol.* **212,** 403–428.
3. Holm, L. and Sander, C. (1993) Protein-structure comparison by alignment of distance matrices. *J. Mol. Biol.* **233,** 123–138.
4. Nussinov, R. and Wolfson, H. J. (1991) Efficient detection of 3-dimensional structure motis in biological macromolecules by computer vision techniques. *Proc. Natl. Acad. Sci. USA* **88,** 10,495–10,499.
5. Gibrat, J. F., Madej, T., Spouge, J. L., and Bryant S. H. (1997) The VAST protein structure comparison method. *Biophys. J.* **72,** MP298.
6. Taylor, W. R. (1997) Random models for double dynamic score normalization. *J. Mol. Evol.* **44,** S174-S180. (Special issue in memory of Kimura.)
7. Taylor, W. F. and Orengo, C. A. (1989) A holistic approach to protein structure comparison. *Prot. Eng.* **2,** 505–519.
8. Jones, D. T., Taylor, W. R., and Thornton J. M. (1992) A new approach to protein fold recognition. *Nature* **358,** 86–89.
9. Taylor, W. R. (1997) Multiple sequence threading: an analysis of alignment quality and stability. *J. Mol. Biol.* **269,** 902–943
10. Neeleman, S. B. and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48,** 443–453.

11. Orengo, C. A. and Taylor, W. R. (1990) A rapid method for protein structure alignment. *J. Theor. Biol.* **147,** 517–551.
12. Taylor, W. R., Flores, T. P., and Orengo, C. A. (1994) Multiple protein structure alignment. *Protein Sci.* **3,** 1858–1870.
13. Rippmann, F. and Taylor, W. R. (1991) Visualization of structural similarity in proteins. *J. Mol. Graph.* **9,** 3–16

3

Discovering Patterns Conserved in Sets of Unaligned Protein Sequences

Inge Jonassen

1. Introduction

A protein family is a set of proteins that are homologous (have a common ancestor) and have similar function and structure. If one discovers that a pattern of residues is common to the sequences in a family, it is possible that the presence of these residues is crucial to the structure and/or function of the proteins. For example, the sequences in the zinc finger c2h2 family of DNA binding proteins all match the pattern `C-x(2,4)-C-x(3)-[LIVFYWC] -H-x(3,5)-H`. The pattern describes residues critical to the formation of the substructure (finger) that interacts with the DNA molecule. The substructure contains a zinc ion coordinated by two cysteines and two histidines. In addition to characterizing the sequences of the proteins in the family, the pattern can be used for classification, i.e., for identifying new proteins belonging to the zinc finger c2h2 family.

The problem that we are addressing in this chapter is how to discover the most interesting patterns conserved in a protein family or in any set of protein sequences believed to be related. We focus on two important aspects of this problem. The first is how to decide which are the most interesting patterns, i.e., the problem of evaluating the patterns. The second aspect is how to proceed to discover the most interesting patterns.

One approach to pattern discovery is to first obtain a global or local alignment of the sequences. Depending on the quality of the alignment, interesting conserved patterns can be found from the alignment. Another approach is to use a method for discovering conserved patterns directly from the unaligned sequences. Both approaches have their strengths and weaknesses. For example, high-quality multiple alignments are difficult to obtain when the sequences contain repeated elements, and in these cases methods for discovering conserved patterns directly

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

from the unaligned sequences can provide better results. And, conversely, multiple global alignment methods may be better at picking up subtle similarities between sequences that are globally similar.

Other chapters in this volume cover methods for the automatic alignment of sequences. Here we describe methods for the automatic discovery of patterns conserved in unaligned protein sequences. A number of methods have been developed, and we look in some detail at the methods implemented in the PRATT programs ***(1,2)***. We give a detailed procedure for how to use the program. First we provide some background on the use of patterns, and on alternative ways to discover and evaluate patterns. *See* **ref. *3*** for a more in-depth discussion of pattern discovery approaches.

1.1. Defining Patterns

We adopt the pattern notation used in the PROSITE database of protein sites and families ***(4)***. A pattern is a list of pattern positions, the positions being separated by hyphens "- ". A pattern position can be:

1. An *identity position* contains one letter, e.g., C, and matches one identical letter in the sequence.
2. An *ambiguous position* matches any one of a specified set of alternative letters. The set is specified in one of two ways:
   a. The allowed letters are given within brackets, e.g., `[ADE]`.
   b. The forbidden letters are given within braces, e.g., `{KEH}`.
   A sequence letter matches the ambiguous pattern position if it is one of the allowed letters (given within brackets), or if it is not one of the forbidden letters (within braces).
3. A *wildcard* `x` which matches any one letter in the sequence.

Pattern positions can be repeated a fixed or variable number of times by writing *(i)* or *(i,j)* after the position where *i* and *j* are non-negative integers. The repeated position is matched by *i* (or between *i* and *j*) consecutive letters in the sequence that each matches the pattern position. For example, `[DE](4,6)` matches between four and six consecutive letters in the sequence, each letter being either D or E, and `x(2,4)` matches between two and four consecutive arbitrary letters.

When one is matching a sequence and a pattern, consecutive pattern positions are to match consecutive elements of the sequence. A sequence matches the pattern if it contains consecutive elements matching all of the positions of the pattern. For example, the pattern `A-x(2,3)-[DE]` is matched by any sequence containing an A followed by two or three arbitrary letters followed by a D or an E. So the sequence `SEALVDS` matches this pattern, but the sequence `SALVIKESLA` does not. Additionally, PROSITE patterns can be restricted to

match from the beginning of the sequence by writing "<" in front of the pattern or to match until the end of a sequence by appending ">" to the pattern. For example, the sequence `ALVDS` matches `<A-x(2,3)-[DE]`, but `SEALVDS` does not. Most patterns used for protein sequences can be described using the PROSITE pattern notation.

We will write patterns in a simplified form that allows for writing most of the patterns given in the PROSITE database and all patterns that can be discovered using the PRATT algorithm (*see* **Subheading 2.2.2.3.**).

$$P = A_1 - x(i_1,j_1) - A_2 - x(i_2,j_2) - A_{3-} \ldots - x(i_{p-1},j_{p-1}) - Ap \quad (1)$$

where each A_k is a nonempty set of amino acid symbols, and i_k and j_k are nonnegative integers so that $i_k \leq j_k$. The set A_k represents a nonwildcard pattern position (from now on simply called pattern position). A_k is an identity position if it contains one letter and an ambiguous position if it contains more than one letter. We write ambiguous pattern positions using brackets. A wildcard $x(i_k,j_k)$ is said to be *flexible* if $j_k > i_k$, otherwise it is *fixed*.

Most pattern discovery methods use exact matching between patterns and sequences, i.e., they report only patterns that have exact matches in all the input sequences, or in some proportion of them. However, some methods do allow for approximate matching. A sequence S matches a pattern P approximately if there is another sequence T that matches P so that T can be obtained from S by, at most, some maximum number of basic operations (substitutions, insertions, deletions). An alternative to patterns is profiles or hidden Markov models (*see* **Note 1**).

1.2. Use of Patterns

Patterns can be used to describe residues that are conserved in a set of sequences. Discovering patterns conserved in a protein family can help in the understanding of relationships between sequence, structure, and function of the proteins under study. When a conserved pattern has been discovered, one should analyze how likely it is that pattern has been conserved by chance. The less likely this is, the more likely the pattern is to describe functionally or structurally important residues (*see* **Subheading 2.1.**).

If one finds a pattern that not only is conserved in the family, but also is unique to the family, i.e., no (or few) sequences outside the family matches the pattern, then the pattern can be used to identify new members of the family. The PROSITE database of protein sites and families illustrates this. Release 13 (November 1995) of PROSITE contains more than 1000 families. For most of the families a pattern is given that is matched by (most of) the sequences in the family and a few other sequences. The patterns often describe structurally or functionally important residues, but the primary purpose of the patterns is that they should be useful for classification purposes. The sequences in the family

that do not match the pattern are called *false negatives*, and the sequences outside the family that do match the pattern are called *false positives*. Ideally, the pattern for a family has no false positives or negatives.

Using a pattern to identify new family members is more efficient than comparing a new sequence to every sequence in the family. Also, a pattern can sometimes provide a more sensitive test of family membership. While similarities and dissimilarities are rewarded and penalized equally along the complete sequences in pairwise sequence comparison, one can define a sequence pattern that describes only the elements needed for the sequence to be a member of the family.

Recently, PROSITE has started using profiles (*see* **Note 1**) as an alternative to patterns. The profiles have higher expressive power and are able to describe conserved regions that cannot be described by patterns, e.g., when there are too few completely or highly conserved positions. However, when it is possible to describe the conserved elements using a pattern, this gives a very compact and easily interpretable representation. Also, the direct discovery of profiles from unaligned sequences is more difficult than discovery of conserved patterns. Profiles contain many parameters and, to get good estimates of their values, one needs a large number of examples (sequences in the family). Otherwise the resulting profile might overfit the examples, i.e., it might fail to generalize to other family members.

Pattern discovery methods can be used to "fish" for new relationships in sets of sequences, e.g., to find new protein families. Finding that a set of sequences contains a conserved pattern, depending on the "strength" of the pattern, one might find it unlikely that the pattern has evolved independently in these sequences and therefore hypothesize that the sequences are evolutionarily related. This type of analysis is similar to that used in sequence similarity searches like FastA ***(5)*** and BLAST ***(6)***. However, using pattern discovery, one is not limited to analyzing two sequences at the time. Patterns that are not unexpected to be shared by two sequences can be highly unexpected when found to be common to many sequences. Thus, multiple sequence comparison in general, and pattern discovery in particular, is more sensitive to subtle similarities between sequences than is pairwise comparison.

Methods for the discovery of conserved pattern can also be used as an aid in solving other problems. For example, in order to find a good profile for a family, a first step can be to discover patterns conserved in subsets of the sequences in the family. The local alignment of the segments matching a pattern can be used as a starting point for making a multiple global sequence alignment. This ideas was used in early alignment methods that first found sets of identical (or very similar) segments, one from each sequence, and based their alignment on using these as "anchors," *see*, e.g., **ref. *7***. Also, structure prediction methods

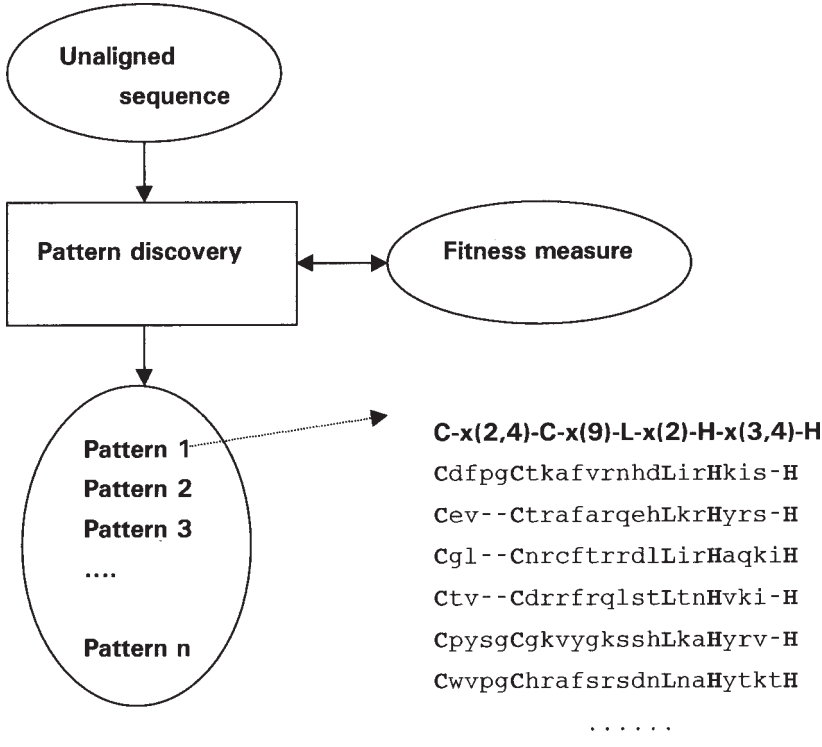


Fig. 1. Schematic figure of pattern discovery process. The unaligned sequences are input to a pattern discovery program that finds conserved patterns. Each pattern also defines a local alignment of the segments matching the pattern. For example, the figure shows a pattern output from the Pratt program when analyzing a subset of the sequences in the zinc finger c2h2 family from PROSITE (accession number PS00028).

can be based on patterns. We will not discuss this use of patterns further, but refer the interested reader to a survey *(8)*.

2. Pattern Discovery

We have seen that pattern discovery can be used for different purposes, i.e., classification, characterization, and discovery of new families. In this section we describe approaches to the automatic discovery of patterns. The aim is to help the investigator in making informed decisions about how to proceed in his/her own analysis, e.g., what sequences to include in the analysis, what method to use, and how to interpret the results. **Figure 1** illustrates pattern discovery.

If the aim is to find a pattern characterizing a family of proteins, one should collect a set of sequences belonging to the family so that the set represents as much variation within the family as possible. For most methods, one should

also try to avoid biases in the selected set of sequences in order to get a fair evaluation of the discovered patterns (*see* **Subheading 2.1.**). Collecting sequences, one may, by mistake, include some sequences that do not belong to the family, or the sequences may contain errors ***(9)***. Also, patterns conserved in subsets of the family may bring valuable information especially in cases where there is no nontrivial pattern conserved in the complete family. Therefore, one should not only look for patterns matching all the sequences, but also for patterns matching subsets of the sequences. Hence, the problem of discovering patterns in a family is very closely related to the problem of discovering families in sets of possibly related sequences. In both cases one searches for interesting patterns shared by a subset of the input sequences.

A pattern discovery method normally takes as input one or several sets of sequences and tries to find the patterns that are "best" for the input sequences. Before going in detail on the methods used to do this, we will discuss different ways of evaluating the "goodness" of the patterns.

2.1. Evaluation of Patterns

Having found that a number of patterns all are shared by the input sequences *S,* we want to rank higher the patterns that brings us the most information about the sequences in *S* or that are the least likely (probable) to be conserved in *S* by chance. Both information content and statistical measures have been used to rank discovered patterns.

2.1.1. Information Content and Minimum Description Length

In **ref.** ***1*** we defined the information content of a pattern. The information content of the pattern as defined in **Subheading 1.1.**, is

$$I(P) = \sum_{k=1}^{p} I_1(A_k) - \sum_{k=1}^{p-1} c \cdot (j_k - i_k) \tag{2}$$

where c is a constant (normally set to 0.5), and

$$I_1(A_k) = -\sum_{a \in A} (p(a) \cdot \log p(a)) + \sum_{a \in A} p(a)/p(A_k) \cdot \log p(a)/p(A_k) \tag{3}$$

A is the set of all 20 one-letter amino acid symbols, $p(a)$ is the a priori probability of amino acid a (approximated by the frequency of a in a sequence database), and $p(A) = \Sigma_{b \in A}\, p(b)$.

The information content is the sum of the information content of the pattern positions A_1 to A_p minus a penalty for the flexibility of the wildcards. The information content of the position A_k is the reduction in uncertainty when you are told that an amino acid belongs to the set A_k, and is defined as in ***(10)***. The function $I(P)$ gives a reasonable ranking of patterns when all the patterns

match the same number of sequences. A generalization of the measure above taking into account the number of sequences matching each pattern was developed by Brazma et al. ***(11)***. They used the minimum description length (MDL) principle from machine learning ***(12)***, which states that the best explanation for a set of examples/observations is the theory that minimizes the length (in bits) of the coding of the theory (pattern) itself, and the examples (sequences) encoded using the theory. In our case, a strong pattern will shorten the description when coding each matching sequence using the pattern, but on the other hand describing the pattern itself will also require some bits and might not give increased compression if it is only matched by a few sequences. Using the MDL principle helps to penalize patterns overfitting the example sequences.

2.1.2. Statistical Significance

Another way of defining the fitness measure is based on the statistical significance of the patterns defined as follows ***(3,13)***. Assume that we have found the patterns $p_1,\ldots, p_n$, each p_i matching a set of sequences $S_i \subseteq S$. Then, for the pattern p_i, the pattern probability is the probability that p_i matches at least $|S_i|$ out of $|S|$ random sequences (of the same length and composition as the sequences in S) purely by chance. In this analysis, the sequences and the sequence positions are assumed to be independent. The pattern significance can be defined as the reverse of the pattern probability, thus patterns having lower probability should be ranked higher. The statistical significance of the pattern will increase either if the information content of the pattern increases or if the pattern matches more sequences.

Both the statistical significance measure and the MDL-based fitness measure is sensitive to biases in the sequence set. For example, if a set of very similar sequences has been included, a pattern matching one of these will probably match them all and in this way get a too high score. This effect can be avoided by not including very similar sequences in the set of sequences to be analyzed, or by using a measure explicitly taking this effect into account. One such measure has been proposed by Jonassen et al. ***(14)***, and is included in the PRATT program (*see* **Subheading 2.3.2.**).

2.1.3. Evaluating Patterns to be Used for Classification

If the aim of the discovery is to find a pattern that can be used for classification, then the best pattern is one that matches all of the sequences in the family S (no false negatives) and none of the sequences outside S (no false positives) and at the same time avoids overfitting the examples. In choosing a pattern to be used for classification, there will often be a trade-off between the specificity (number of false positives) and the sensitivity (number of false negatives): weaker patterns may pick up all family members, but also a number of false positives, whereas stronger patterns may match no sequences outside the family, but they may also fail to match some family members. In order to calculate the number of

false positives, one needs to check the patterns against the set (or some subset) of sequences outside the family under analysis. Some methods for pattern discovery therefore take as input not only positive examples (sequences in the family) but also a set of negative examples (sequences outside the family); *see*, e.g., **ref. *15***. As an alternative, one can use the pattern probability or information content to estimate the number of false positives. An analysis performed by Sternberg demonstrates that there is a strong correlation between the pattern probability and the number of false positives for the patterns in the PROSITE database ***(16)***.

2.1.4. Biological Significance

If we have found a significant pattern, it is likely that the matching sequences are evolutionarily related. We might also hypothesize that it corresponds to residues critical to the function of the proteins in the family. The hypothesis can be tested using experimental techniques, e.g., site-directed mutagenesis. Alternatively, if the structure of one or several of the proteins in the family, is known, one can analyze the structural location of the residues in the pattern, e.g., using the tool PDBMOTIF ***(17)*** in combination with RASMOL ***(18)***. Moreover, if the structures of several proteins in the family are known, one can assess whether the residues matching the pattern are structurally equivalent in the proteins. McClure et al. ***(19)*** carried out a systematic study of multiple sequence alignment programs to see whether they correctly aligned structurally conserved motifs.

2.2. Discovery of Patterns

There exist a large number of different methods for the discovery of patterns. Here we only give a few examples of methods reported in the literature. For a more thorough survey, *see* **ref. *3***. Important characteriztics of the individual methods are:

1. What class of patterns are they able to discover? Most methods are able to discover subsets of the patterns that can be written using PROSITE notation. All methods can discover patterns with identity positions, some allow for ambiguous and wildcard positions, and some allow for flexibility — e.g., elements of the type `x(2,4)`.
2. How are discovered patterns evaluated? Which fitness function is used?
3. Is the algorithm guaranteed to find the best pattern (with respect to the fitness function)?
4. How efficient is the algorithm? The running time may depend linearly on the number and length of the sequences, quadratically in the number of sequences, etc.

2.2.1. Pattern Discovery Algorithms Based on Pairwise Sequence Comparisons

In **ref. *3*** these algorithms are referred to as sequence driven approaches. Methods for comparing pairs of sequences are often based on dynamic programming, which finds an optimal (with respect to a predefined scoring

scheme) alignment using time proportional to the product of the lengths of the two sequences. Theoretically this could be extended to aligning k sequences for any $k > 1$, but the computation time would grow proportionally to l^k where l is the average sequence length. Thus, the approach becomes infeasible even for quite small k values, e.g., $k = 10$. There are strong reasons to believe that there does not exist efficient (polynomial time) algorithms guaranteed to find optimal multiple alignments when the number of sequences is not bounded by a constant ***(20)***.

A natural approach to multiple sequence comparison is to combine the results of several pairwise comparisons. This approach is used in many methods for multiple global sequence alignment. For example, the program CLUSTAL W uses what has been called the progressive alignment method (*see* **ref.** ***21*** and Chapter 1 in this volume). Smith and Smith proposed to use a closely related method for pattern discovery ***(22)***. They progressively build a pattern that is common to all the sequences, first finding the best pattern common to the two most similar sequences, and later on finding the best pattern common to a pair of sequences, a pair of patterns, or a pair of one pattern and one sequence. Each pairwise comparison is done using dynamic programming, and its result is a pattern with maximum score (instead of an alignment as in CLUSTAL W). A pattern here is a string of amino acid characters, special gap characters, and characters from an Amino Acid Class Covering (AACC) hierarchy. Although the result of each pairwise comparison is guaranteed to be optimal, the optimality of the end result (pattern common to all the sequences) with respect to the input sequences, cannot be guaranteed. However, the method seems to work well in many cases.

A number of other methods also use pairwise sequence comparisons to build up a pattern shared by all of or many of the example sequences. Examples are MacAW ***(23)*** and methods proposed by Roytberg ***(24)*** and Vingron and Argos ***(25)***.

2.2.2. Pattern Discovery Algorithms Based on Enumerating or Searching a Solution Space

Some times it is more efficient to start from the other side, that is to start from the patterns instead of from the sequences. This is called the pattern–driven approach in **ref.** ***3***. One first defines a *solution space,* i.e., a set of patterns that the algorithm should be able to discover. As a very simple example, this could be the set of all patterns of length 4 with only identity positions. Then one searches the solution space to find the patterns having the highest fitness with respect to the input sequences. The best patterns found are presented to the user. If the number of patterns to be considered is not too large, one can simply enumerate and analyze all the patterns. For each pattern one needs to find the set of matching sequences. This can be done in time linear in the total length of the sequences. On the other hand, the number of patterns to

be considered typically increases exponentially with the maximum pattern length. Thus, normally the time usage of these algorithms depends more heavily on the definition of the pattern set than on the number and lengths of the sequences to be analyzed.

2.2.2.1. MOTIF

This method was proposed by Smith et al. ***(26)***. They define their solution space to be the set of patterns on the form $A_1 - x(d_1) - A_2 - x(d_2) - A_3$ where A_1, A_2, and A_3 are single amino acid symbols, and d_1 and d_2 are non-negative integers within some range (for instance, less than or equal to 10). For each pattern in this set, they count the number of matching sequences. For the most promising patterns, they make an alignment of the matching segments that is then improved using a heuristic method. Sequences not having the pattern are added to the alignment. This method is used in making the BLOCKS database ***(27)***.

2.2.2.2. ASSET

Neuwald and Green ***(13)*** took the idea of Smith et al. ***(27)*** further, allowing for a larger number of identity positions and fixed-length wildcards. As the number of patterns in the solution space grows exponentially with the maximum number of pattern positions, exhaustive enumeration cannot be used in this case. Instead, Neuwald and Green use a depth-first search algorithm to find the most significant patterns (using a measure of statistical significance). The search is heuristically pruned by cutting parts of the solution space that will probably not contain significant patterns. The discovered patterns are combined, if possible, and the matching segments are used as a starting point for a profile that is then refined using an iterative algorithm.

2.2.2.3. PRATT

We built on the approach of Neuwald and Green when developing the PRATT program for automatic pattern discovery ***(1,2)***. PRATT is described here in some detail. In PRATT we extended the solution space to include patterns with flexible wildcards and ambiguous positions. On the other hand, we restricted the search to only consider patterns that were matched by some minimum proportion of the sequences (we call those pattern "conserved"). The patterns are written in the form defined in **Subheading 1.1**. The user defines restrictions on the patterns to be considered by specifying the maximum length of wildcards, the maximum degree of flexibility, etc., effectively defining a class (set) of patterns to be considered.

PRATT uses a depth-first search to find the conserved patterns with the highest information content. The search is done in two phases, during the first of which patterns with only identity positions and (possibly flexible) wildcards are considered. During the second phase, the best patterns found during the first phase, are refined, i.e., they are analyzed to see if ambiguous symbols can be added.

The refinement phase can be done either using an exhaustive search or a heuristic algorithm. During the refinement phase, ambiguous symbols can be added inside a pattern (substituting wildcard positions), or added to the right (*see* **Note 2**) of a pattern. As an option, patterns with ambiguous symbols can also be considered during the first phase, but in practice this slows the program down considerably.

In the first version of PRATT ***(*ref. *1)***, an exhaustive search for the best pattern in the defined class is performed during the first phase. For the second phase, one can choose between a heuristic and an exhaustive algorithm. This version of PRATT can be guaranteed to find the conserved pattern with the highest information content. In many cases it works very efficiently. However, when the sequences share strong patterns (with high information content), the program takes very long time to run. One reason is that the program contains no mechanism to avoid considering *all* conserved patterns in the defined class, and not only the ones with the highest information content. Motivated by this inefficiency, we designed a new version of the program that aims at finding only the highest scoring conserved patterns ***(2)***. This uses branch-and-bound and heuristics to reduce the search time. A pruning mechanism is introduced that avoids exploring unnecessarily generalized patterns. This can be guaranteed to find the best pattern when no flexible spacers are allowed, but it is heuristic when flexible spacers *are* allowed (the optimality of the result cannot be guaranteed). Experimental results indicate that even when the heuristics are used, PRATT still often finds the highest scoring conserved patterns.

The resulting program is reasonably efficient. In an experiment, more than 900 of the families in the PROSITE database were analyzed in less than 10 s each when using default parameters and a UNIX desktop workstation (for more details, *see* **ref. *2***). However, the program can use very much time when searching for patterns matching a relatively small proportion of the sequences (e.g., minimum 10%), especially when big sets of sequences are analyzed. One reason for this is that there is a large number of sequence subsets of size >10%, and the program will find many patterns that, by chance, match one such subset. More details on how to use PRATT is included in **Subheading 2.3.** below.

2.2.2.4. Sagot and Co-workers

Sagot et al. have explored using depth-first search and breadth-first search to find conserved patterns without wildcards but allowing for ambiguous positions ***(28,29)***. They found that breadth-first search is more efficient, but that it is very memory-intensive and can realistically be applied only for finding very short patterns.

In a later paper, Sagot and Viari *(30)* presents an algorithm using a depth-first search strategy that is able to discover patterns with both ambiguous positions and fixed-length wildcards. As in PRATT, they let the user set a minimum percentage of the sequences that match a pattern for the pattern to be considered. One advantage of this method over PRATT is that, in principle, it is not necessary to specify beforehand which groups of letters are to be used in ambiguous pattern positions. However, in practice this is necessary when analyzing protein sequences. One also needs to define a limit on how many times each possible pattern symbol can be used in an individual pattern. Improvements over the straightforward depth-first search algorithms are presented that are in many ways similar to the ones we have used in PRATT. If two patterns match the same segments, and if one of them is a generalization of the other, only the most specific one is analyzed further in the search (this corresponds to the pruning of too general patterns in the second version of PRATT). Also, they do an initial search (for sketching the solution space) analogous to the two-phase search strategy used in PRATT. Programs implementing the algorithms of Sagot and co-workers have not yet been made publicly available

2.3. Using PRATT to Discover Patterns

The source code for PRATT can be downloaded from `ftp://ftp.ii.uib.no/pub/bio/PRATT`. The latest version is 2.2. The program is written in ANSI C, and has been compiled and run successfully on a number of different UNIX operating systems, on Linux, and on OS/2 systems. There are world wide web servers allowing you to run PRATT on remote servers at EBI (http://www2.ebi.ac.uk/Pratt) and in Bergen (http://www.ii.uib.no/~inge/Pratt.html). From this page you will also find more information on the PRATT program (*see also* **refs. *1*** and ***2***). Also, *see* **Note 4** for instructions on how to download and install the program locally.

2.3.1. Input Format and Command Line

PRATT accepts as input sequences in one ASCII text file using either SWISS-PROT *(31)* or FASTA format (without annotation — *see* **Note 4**). Depending on the format your sequences are given in, PRATT is started using the following command: `PRATT fasta filename` or `PRATT swissprot filename`. As an alternative to using the menu described in **Subheading 2.3.2.** you can give the parameters on command line after the filename.

2.3.2. Choosing Parameters

Running PRATT, you get a menu allowing you to choose values for a large number of parameters. The values chosen determine what type of patterns

Table 1
Pratt Version 2.2: Analyzing 286 Sequences from File Sequences

Pattern conservation		Search parameters	
CM: min Nr of Seqs	286	G: pattern graph from seq	
C%: min Percentage	100.0		
		E: search greediness	3
Pattern restrictions		R: pattern refinement	on
PP: pos in seq	off	RG: generalize	off
PL: max length	50		
PN: max Nr of symbols	50	*Output*	
PX: max Nr of x's	5	OF: output filename	seqs.286.pat
FN: max Nr of flexibility	2	OP: pat notation	on
FL: max flexibility	2	ON: max nr patterns	50
BI: symbol file	off	OA: max nr alignments	50
		M: match summary	on
BN: initial search	20	MR: ratio	10
		MV: vertical summary	off
Pattern Scoring			
S: scoring	info		
X: eXecute program	Q: Quit	H: Help	

Table 1. PRATT's menu when analyzing a set of 286 sequences contained in the file seqs.

PRATT will look for, how they are evaluated, and in what way the results are presented to you. PRATT's menu is shown in **Table 1**. The parameters of PRATT fall into groups. The first is used to set the minimum number of sequences a pattern should match (CM and C% parameters). The other groups contain parameters defining the set of patterns that can be discovered, how the patterns are to be evaluated, etc.

2.3.2.1. Defining a Set of Patterns

Choosing liberal values for pattern restrictions parameters (e.g., allowing for long, flexible wildcards, ambiguous pattern positions in the initial search — *see* **Note 5**) enables PRATT to find more patterns. On the other hand, if the parameters are too liberal, the pattern search might take a long time and require a lot of memory (*see* **Note 6**).

One strategy is to first use quite strict values for the parameters (e.g., using default parameter values). The analysis using these parameters is normally quite fast. If no interesting patterns are found, one can

1. reduce the minimum number of sequences that a pattern should match (using the CM or C% options, and/or
2. use more liberal restrictions on the patterns. For example, one might increase the maximum length of a wildcard (using the "PX" option), or the maximum flexibility

(referring to the pattern defined in **Subheading 1.1.**, this is the maximum value of $j_k - i_k$) using the FL option. One might want to allow for more flexible wildcards (using the FN option), or for ambiguous symbols during the initial search (using the BN option – *see* **Note 5**).

This will help to reveal patterns conserved in different subsets of the input sequences. Another set of parameters in PRATT offers a choice of different evaluation functions.

2.3.2.2. Evaluation of Patterns

By default, the fitness of a pattern is its information content as defined in **Subheading 2.1.1**. Alternative fitness functions can be chosen using the S option:

1. *MDL* — the pattern's fitness will also depend on how many sequences it matches. Using this measure, one can also choose values for a set of parameters having to do with the coding scheme used. The values should be related to the alphabets used for the sequences and the patterns.
2. *tree* — the fitness of a pattern will depend on how different the matching sequences are. This is to correct for biases in the set of input sequences. To use this one needs to input a file containing an estimate of the phylogenetic tree showing the evolutionary relationships between the sequences, e.g., the guide tree produced by CLUSTAL W ***(21)***. For more details, *see* **ref. *14***.
3. *ppv* — the conserved patterns having the highest information content are scanned against the SWISS-PROT database to find the number of matches outside the family. In the analysis it is assumed that all input sequences are from the SWISS-PROT database. The positive predictive value (ppv) for a pattern is the proportion of the sequences it matches, that actually belongs to the family. The patterns are ranked by their ppv. In order to use this option, you need to have a local copy of the SWISS-PROT database in flat file format.

2.3.2.3. Search Parameters

As mentioned in **Subheading 2.2.2.**, PRATT uses heuristics to speed the search for the best patterns. The user can control the degree of greediness by adjusting the value of the E parameter. The default value is 3, which gives reasonable performance on protein sequences (if DNA sequences are to be analyzed, one should use a lower value). The higher value one chooses for the E parameter, the more greedy and faster the search will be, but on the other hand, the more likely one is to miss good patterns. To be guaranteed to find the best pattern, set E to 0 if you allow for flexible wildcards, or set E to 1 if you do not allow flexible wildcards (*see* **ref. *2*** for details).

2.3.2.4. Refinement Parameters

The patterns found in the initial search can be output directly (set the R option to "off") or they can be subjected to the refinement phase. Normally, during pattern refinement, PRATT will include as few amino acids in the

ambiguous positions as possible to make the pattern match the required minimum number of sequences. The amino acids included will be a subset of one of the allowed amino acid groups (*see* **Note 5**). By switching on the RG option, PRATT will include the complete smallest allowed group instead. For example, assume that `ILVF` is an allowed group and that the pattern `D-x(2)-E` can be refined to `D-x-[IV]-E` and still be conserved. If RG is set to "off," this is the pattern PRATT will find, but if RG is turned on, PRATT will produce the pattern `D-x-[ILVF]-E`. An argument for including the whole group can be that if I and V have been seen in this position, then it is likely that other proteins in the family can have L or F in this position.

2.3.2.5. Setting Output Parameters

Normally the patterns found by PRATT are output to a file with the filename of the input appended with `.k.pat` where *k* is the value chosen for the CM parameter (minimum number of sequences to match a pattern). You can instruct PRATT to use another filename by using the OF option. Also, you can choose to output the patterns in a simplified format instead of PROSITE format by toggling the OP parameter. ON gives the (maximum) number of patterns to be output, and for the OA best, PRATT will also print the matching segments. If the M option is on, it will also print a summary of where in the sequences the different patterns have their matches.

2.3.3. Using Alignments and Query Sequences

From version 2, PRATT uses a *pattern graph* to define the set of patterns to be analyzed. All patterns considered in the search are derived from paths in the graph (for details, *see* **ref. *2***). Normally the graph is constructed so that it is possible to find any conserved pattern in the user-defined class. However, using the "G" option, the user can instruct PRATT to make the graph from an alignment (set "G" to "al") or from a special query sequence (set "G" to "q") in order to restrict PRATT's search to patterns consistent with the alignment or matching the special query sequence.

A pattern is said to be consistent with an alignment if each sequence in the alignment has a match to the pattern so that for each (nonwildcard) pattern position the matching sequence symbols are on top of each other in the alignment. This option is intended to be used when one has obtained a reliable alignment for a subset of the sequences. This can be the case, e.g., when the structure of some of the proteins are known. Using an alignment to guide PRATT can also speed up the program, especially when the alignment is between sequences that are not too similar. One can also use this option if one has an alignment of the whole set of sequences. PRATT will then find the patterns from the alignment that score the highest with respect to any of the evaluation functions supported by PRATT.

Constructing the pattern graph from a special query sequence can be useful if one wants to use PRATT together with a sequence similarity search ***(5,6)***. As-

sume that you have found a set *S* of sequences that are similar to a query sequence *q*. Now you can input *S* (the complete sequences or the segments similar to segments in *q*) to PRATT and instruct it to search for patterns matching *q* (which will be guaranteed when the pattern graph is constructed from *q*) and a minimum number of the sequences in *S*. This might help to lift similarities out of the twilight zone. Tatusov et al., among others, has has performed experiments with a similar approach ***(32)***, using a local alignment method where we propose to use PRATT. One advantage of using PRATT is that it allows for insertions/deletions between the matching segments (i.e., flexible wildcards).

2.4. Examples

Here we give some examples of results obtained using PRATT version 2.2 running on a Sun Ultra 1. The sequence families analyzed were taken from PROSITE release 13.0 (November 1995) ***(4)***, and the full sequences were retrieved from release 34 (October 1996) of the SWISS-PROT protein sequence database ***(31)***.

2.4.1. Zinc Finger c2h2 Family

The 286 sequences in the zinc finger c2h2 family in PROSITE (accession number PS00028), were input to PRATT. Using default parameters (requiring patterns to match all 286 sequences), the best conserved patterns we got were `H-x(3,5)-H` and `C-x(2,4)-C` (each having an information content of 7.3). We ran PRATT again, this time setting the PX parameter to 15 to allow for longer wildcards. Now, the best patterns we got were `C-x(2,4)-C-x(12)-H-x(3,5)-H` and `C-x(12)-H-x(3,5)-H` (having an information content, respectively, of 14.7 and 11.5). Finally, we set CM to 285 to find patterns matching all but one sequence (and PX to 15). This time we got the patterns `C-x(2,4)-C-x(3)-[CFILMVY]-x(8)-H-x(3,5)-H` (which is the result of refining `C-x(2,4)-C-x(12)-H-x(3,5)-H)` and `C-x(3)-[CFILMVY]-x(8)-H-x(3,5)-H` (refinement of `C-x(12)-H-x(3,5)-H`). These patterns have an information content of 16.2 and 13.0. We see that the longest pattern we got is very similar to the one given in PROSITE (*see* **Subheading 1.**). Each of these PRATT runs took 15–20 s.

2.4.2. Somatomedin Family

From SWISS-PROT we retrieved the six (unaligned) sequences contained in the somatomedin family in PROSITE (accession number PS00524). Running PRATT using default parameters takes 14 s and produces, as the best conserved pattern, `I-x(0,2)-L-x(1,3)-L-x-[ALPV]-x(3)-L-A-x-[ANQ]-[EP]-[DS]-x-[KR]-x(3)-[GPT]-x-[GP]-x(3)-[DENS]-[DEKR]-x(3)-[ACS]-x(3)-[ACN]-x-[FY]-x-[AGQ]-x-[CGT]-x[ACGT]` having an information content of 59.9. Considering that there were only six sequences, the pattern contains ambiguous positions with too many

alternative amino acids. The amino acid groups that PRATT by default uses during refinement are intended for use with bigger sets of sequences. Switching off refinement (setting "R" to "off"), PRATT takes 12 s and finds as the best pattern `C-x-C-x(3)-C-x(5)-C-C-x-D-x(1,3)-E-x(0,2)-C` (with an information content of 31.4). Disallowing flexible wildcards (setting FN to 0) we get in less than 1 s, the pattern `C-x-C-x(3)-C-x(5)-C-C-x-D-x(4)-C` (with an information content of 29.2). This example illustrates that one should be careful when choosing values for the parameters; one set of values does not give the best result in all cases.

2.4.3. Snake Toxin Family

We retrieved the 166 sequences from the snake toxin family in PROSITE (accession number PS00272). Using default parameters PRATT uses less than 1 s, but finds no conserved patterns (matching all 166 sequences). Allowing for longer wildcards (setting PX to 10), PRATT still finds no patterns. Setting CM to 160 (searching for patterns matching at least 160 sequences), PRATT uses 6 s and finds the pattern `G-C-x(1,3)-C` (having an information content of 11.5). Reducing CM further to 155, PRATT uses 13 s to find (among others) the pattern `G-C-x(1,3)-C-P-x(8,10)-C-C-x(2)-[DENP]` (having an information content of 25.2). We first found this pattern using the first version of PRATT ***(1)***. The pattern was later included in the PROSITE database. It had 11 false negatives (166 – 155 = 11) and no false positives, i.e., no sequences outside the family matched the pattern.

We wanted to use PRATT's option for evaluating patterns by their discriminatory power (*see* **Subheading 2.3.2.**), and ran PRATT again setting option S to ppv and ON to 4 (as PRATT first stores in memory all sequences matching each pattern, we can only subject a few patterns to this analysis). Apart from this we used the same parameters as in the last paragraph, and we got the pattern shown there. PRATT finds this to match 161 sequences in SWISS-PROT out of which 155 are members of the snake toxin family as given in release 13.0 of PROSITE. This gives a positive predictive value of 0.96. As the information in the SWISS-PROT and PROSITE databases had not been synchronized since November 1995, it is likely that the six new matches (which are sequences added to SWISS-PROT after November 1995) actually belongs to the snake toxin family.

3. Notes

1. As an alternative to patterns, the best conserved part of a set of related sequences can be described using a profile ***(33)***. Although a pattern gives a single amino acid (or a set of alternative amino acids) for each position in the pattern, a profile specifies a score for each of the 20 amino acids in each position. Additionally, a profile specifies position-specific insertion/deletion penalties for each position. The match between a profile and a sequence is given a score, and a sequence that receives a high score when compared to a profile for a specific family is believed to belong to that family.

This differs from patterns where a sequence either matches or does not match the pattern. Another alternative to patterns is hidden Markov models – *see* e.g., **ref. *34***.

2. The reason for this is related to the way matches to patterns are found. A block data structure (as described in **ref. *13***) is constructed. Conceptually this is done by first fixing a constant w. Then the set B of all w-segments (substrings of length w) in the sequences is generated. In this process $w - 1$ non-amino acid symbols are appended to each sequence so that there are $l - 1$ segments for a sequence of length l. Next, for each pair (i,a), i being an integer between 1 and w and a being an amino acid symbol, the set $b_{i,a}$ of segments having a in position i is computed. Now, finding the segments matching a pattern can be done using set-intersection operations on the sets $b_{i,a}$. For example, the set of segments matching `A-x(2)-D` is $b_{1,A} \cap b_{4,D}$. The first pattern symbol is always matched against the first symbol in each segment, and the segments matching an extended pattern is found using set intersections between the set of segments matching the original pattern and the sets in the block data structure corresponding to the extension. Therefore, it is not straightforward to find matches to a pattern extended to the left.
3. First use ftp to connect to the anonymous ftp server in Bergen using the command `ftp`. Log in with user name "anonymous" and use your e-mail address as the password. Change directory to /pub/bio/PRATT (`cd /pub/bio/PRATT`) and download the file Pratt2.2.tar (`get Pratt2.2.tar`). Exit ftp (`bye`), and unpack the tar file (`tar xvf Pratt2.2.tar`). Compile the program by using the make command (`make`). If the system complains, you may have to edit the makefile to use compilers etc. available on your local system. If things go well, make should produce an executable file "pratt." If you have problems, contact your local system support administrator or send an e-mail to <inge@ii.uib.no>.
4. SWISS-PROT format is the format used for distributing the flat file version of the SWISS-PROT sequence database ***(31)***. Fasta format is very simple. For each sequence you simply type the sequence name on a separate line beginning with the ">" symbol. After this line you simply type the sequence. The end of the sequence is marked with either end-of-file or with a new line with ">" followed by the name of the next sequence.
5. PRATT uses an ordered list of identity and ambiguous symbols that can be included in patterns. By default, this list first contains the 20 single-letter amino acid symbols and then it contains a number of groups of amino acids from ***(22,35)***. By default, only the first 20 are allowed during the intial pattern search, and the rest are allowed during the pattern refinement phase. The user can define his/her own list of pattern symbols. This is done by making an ascii/text file with one line per symbol. The line should contain the one-letter amino acid symbols to be contained in the pattern symbol. Thus, if one wishes to have a pattern symbol [ILV], e.g., one line in the file should simply contain ILV. The identity symbols should always come at the top of the file. Using option BI, one can instruct PRATT to read its pattern symbols from an external file (specified using option BF). Using option BN, one can specify how many of these pattern symbols should be allowed during the initial pattern search.
6. PRATT can require a lot of memory. The memory usage depends on the total length of the input sequences. Additionally, the memory usage increases with increasing values for the maximum pattern length (option PL), the maximum flexibility (F-options), the number of symbols to be used during initial search (option BN).

References

1. Jonassen, I., Collins, J. F., and Higgins, D. G. (1995) Finding flexible patterns in unaligned protein sequences. *Protein Sci.* **4,** 1587–1595.
2. Jonassen, I. (1997) Efficient discovery of conserved patterns using a pattern graph. *Comput. Appl.* **13,** 509–522.
3. Brazma, A., Jonassen, I., Eidhammer, I., and Gilbert, D. (1997) Approaches to the automatic discovery of patterns in biosequences. *J. Comp. Biol.,* in press.
4. Bairoch, A., Bucher, P., and Hofmann, K (1996) The PROSITE database, its status in 1995. *Nucleic Acids Res.* **24,** 189–196.
5. Lipman, D. J. and Pearson, W. R. (1985) Rapid and sensitive protein similarity searches. *Science* **227,** 1435–1441.
6. Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lippman, D. J. (1990) A basic local alignment search tool. *J. Mol. Biol.* **215,** 403–410.
7. Sobel, F. and Martinez, H. M. (1986) A multiple sequence alignment program. *Nucleic Acids Res.* **14,** 363–374.
8. Cohen, B. I., Presnell, S. R., and Cohen, F. E. (1991) Pattern-based approaches to structure prediction. *Methods Enzymol.* **202,** 252–268.
9. Kristensen, T., Lopez, R. S., and Prydz, H. (1992) An estimate of the sequencing error frequency in the DNA sequence databases. *DNA Seq.* **2,** 343–346.
10. Shannon, C. E. (1948) A mathematical theory of communication. *Bell Sys. Tech.* **27,** 379–423, 623–656.
11. Brazma, A., Jonassen, I., Ukkonen E., and Vilo, J. (1996) Discovering patterns and subfamilies in biosequences. In *Proceedings of the Fourth International Conference on International Systems for Molecular Biology* (States, D. J., et al., eds.), AAAI Press, pp. 34–43.
12. Rissanen, J. (19978) Modeling by the shortest data description. *Automatica-J/. IFAC* **14,** 465–474.
13. Neuwald, A. F. and Green, P. (1994) Detecteing patterns in protein sequences. *J. Mol. Biol.* **239,** 689–712.
14. Jonassen, I., Helgesen, C., and Higgens, D. G. (1996) Scoring function for pattern discovery programs taking into account sequence diversity. *Report in Informatics* **116,** University of Bergen, Norway.
15. Ogiwara, A., Uchiyama, I., Seto, Y., and Kanehisa, M. (1992) Construction of a dictionary of sequence motifs that characterize groups of related proteins. *Prot. Eng.* **5,** 479–488.
16. Sternberg, M. J. E. (1991) Library of common protein motifs. *Nature* **349,** 111.
17. Saqi, M. A. S. and Sayle, R. (1994) PDBMotif—a tool for the automatic identification and display of motifs in protein structures. *Comput. Appl. Biosci.* **10,** 545–546.
18. Sayle, R. and Milnar White, E. J. (1995) RASMOL: Biomolecular graphics for all. *Trends Biochem. Sci.* **20,** 374–376.
19. McClur, M. A., Vasi, T. K., and Fitch, W. M. (1994) Comparative analysis of multiple protein sequence alignment methods. *Mol. Biol. Evol.* **11,** 571–592.
20. Wang, L. and Jiang, T. (1994) On the complexity of multiple sequence alignment. *J. Comp. Biol.* **1,** 337–348.
21. Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994) Clusatal W: improving the sensitivity of progressive multiple sequence alignment through sequence align-

ment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* **22,** 4673–4680.
22. Smith, R. F. and Smith T. F. (1990) Automatic generation of primary sequence patterns from sets of related protein sequences. *Proc. Natl. Acad. Sci. USA* **87,** 118–122.
23. Schuler, G. D., Altschul, S. F., and Lipman, D. J. (1991) A workbench for multiple alignment construction and analysis. *Proteins: Struct. Funct, Genet.* **9,** 180–190.
24. Roytberg, M. A. (1992) A search for common patterns in many sequences. *Comput. Appl. Biosci.* **8,** 57–64.
25. Vingron, M. and Argos, P. (1991) Motif recognition and alignment for many sequence by comparison of dot-matrices. *J. Mol. Biol.* **213,** 33–34.
26. Smith, H. O., Annau, T. M., and Chandrasegaran (1990) Finding sequence motifs in groups of functionally related proteins. *Proc. Natl. Acad. Sci. USA* **87,** 826–830.
27. Henikoff, S. and Henikoff, J. G. (1991) Automatic assembly of protein blocks for database searching. *Nucleic Acids Res.* **19,** 6565–6572.
28. Sagot, M.-F., Viarai, A., and Soldano, H. (1995) A distance-based block searching algorithm. In *Proceedings of the Third International Conference on International Systems for Molecular Biology* (Rawlings, C., et al., eds.) AAAI Press, pp. 322–331.
29. Sagot, M.-F., Viarai, A., and Soldano, H. (1995) Multiple sequence comparison: a peptide matching approach, in *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching* (Galil, Z. and Ukkonen, E., eds.) Springer-Verlag, New York, pp. 366–385.
30. Sagot, M.-F. and Viari, A. (1996) A double combinatorial approach to discovering patterns in biological sequences, in *Proceedings of the 7th Annual Symposium on Combinatorial Pattern Matching* (Hirschberg, D. and Myers, G., eds.), Springer-Verlag, New York, pp. 186–208.
31. Bairoch, A. and Apweiler, R. (1996) The SWISS-PROT protein sequence data bank and its new supplement TREMBL. *Nucleic Acids Res.* **24,** 189–196.
32. Tatusov, R. L., Altschul, S. F., and Koonin, E. V. (1994) Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Comput. Appl. Biosci.* **8,** 57–64.
33. Gribskov, M., McLachlan, M., and Eisenberg, D. (1987) Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. USA* **89,** 100,915–100,919.
34. Krogh, A., Brown, M., Mian, I. S., Sjoelander, K., and Haussler, D. (1994) Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.* **235,** 1501–1531.
35. Taylor, W. R. (1986) The classification of amino-acid conservation. *J. Theoret. Biol.* **119,** 205–218.

4

Identification of Domains from Protein Sequences

Chris P. Ponting and Ewan Birney

1. Introduction

The fundamental unit of protein structure is the domain, defined as a region or regions of a polypeptide that fold independently and possesses a hydrophobic core (*see* **Note 1**). Domains, particularly those with enzymatic activities, may possess functions independently of whether they are present in isolation or else part of a larger multidomain protein. Other domains confer regulatory and specificity properties to multidomain proteins usually via the provision of binding sites. Because the majority of eukaryotic proteins, and a large number of eubacterial and archaeal proteins, are multidomain in character, the determination of the structures and functions of these proteins requires detailed consideration of their domain architectures.

Experience gathered from decades of structural and molecular biology demonstrates that domain pairs that show similarities in sequence (>35% identity) also possess a common fold and usually possess similarities in function ***(1)***. These domains are thought to be "homologous," i.e., they are derived from a common ancestral gene following its duplication. The amino acid sequences of these domains, once identical at the time of gene duplication, have diverged increasingly during their evolution, yet have retained those amino acid properties that are essential for their proper domain folding, structure, and function. Similarities between homologous domains sequences may have eroded sufficiently that their common evolutionary heritage is not apparent simply from their pairwise comparison. In these cases, the existence of a homology relationship may be inferred either from knowledge of their functions and tertiary structures, and/or application of sequence analysis methods that use multiple alignments. Many methods are now available that are used to predict homology relationships. In this chapter we survey their use and provide indicators on

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

how to realize the predictive potential of sequence analysis by database searching.

2. Materials

All that is needed to perform sequence analysis is a computer with access to the Internet. In **Table 1**, we list a variety of applications that can be accessed via the World Wide Web (WWW). In general, the user cuts and pastes a query protein sequence into a form, sets a variety of parameters, and then initiates the application. Users with nucleotide sequences should first use gene-prediction algorithms to generate open-reading frame (ORF) predictions (a list of gene prediction tools is given at `http://www.bork.embl-heidelberg.de/genepredict.html`).

Detailed analysis of sequences is best performed using tools compiled and running locally. Sequence databases, alignment programs, and database searching tools (*see* **Table 1**) are all available free via anonymous file transfer protocol (FTP) (log-in as username "anonymous" and fill in your e-mail address as the password). Instructions on downloading and compiling files are usually provided at the `ftp` site. In general, programs are provided as "platform-specific" versions that can be installed on different computer systems.

The majority of the applications described here compare protein and not nucleotide sequences. Consequently, and also if computer disk space is limited, it is recommended to download protein sequence databases before nucleotide databases (*see* **Note 2**). For historical reasons, different alignment programs or editors use different file formats (such as CLUSTALW (ALN), multiple sequence format (MSF) and Pearson (FASTA) formats). Format conversion is provided by, among others, the CLUSTALW, CLUSTALX, SEAVIEW, and READSEQ programs.

3. Methods

Later in this section we suggest a recipe for the analysis of a single protein sequence with respect to its domain architecture. This recipe uses various programs that provide different statistical indicators of the significance of predictions. Many of these indicators are derived from nontrivial analyses of score distributions. Anyone using such statistics to derive homology arguments is urged to understand their derivation and the limitations of their use.

3.1. Database Searching: Single or Multiple Sequence Queries

There are many methods that find domains in sequence databases (reviewed in **ref.** *2*). Some of these use distinctive motifs containing relatively few amino acids (I. Jonassen, this volume) and others use the degrees of amino acid conservation present throughout a domain structure as represented in a multiple

Table 1
Useful Links to Sequence Analysis Sites on the WWW

Single Sequence Analysis via WWW		
BLAST	(Sequence similarity)	http://www.ncbi.nlm.nih.gov/blast/ blast.cgi
COILS	(Coiled-coils)	http://www.ch.embnet.org/software/COILS_form.html
FASTA3	(Sequence similarity)	http://www2.ebi.ac.uk/fasta3/
NETOGLYC	(O-glycosylation	http://www.cbs.dtu.dk/services/NetOGlyc/
PSI-BLAST	(Sequence similarity)	http://www.ncbi.nlm.nih.gov/blast/psiblast.cgi
PSORT	(Protein localization)	http://psort.nibb.ac.jp/form.html
SIGNALP	(Signal peptides)	http://www.cbs.dtu.dk/services/SignalP/
TMAP	(Transmembrane regions)	http://www.embl-heidelberg.de/tmap/tmap_sin.html
TMPRED	(Transmembrane regions	http://www.ch.embnet.org/software TMPRED_form.html
Multiple sequence alignments via WWW		
CLUSTALW		http://www2.ebi.ac.uk/clustalw/
MSA		http://www.ibc.wustl.edu/ibc/msa.html
MULTALIN		http://www.toulouse.inra.fr/multalin.html
SAM		http://www.cse.ucsc.edu/research/compbio/short_form.html
SIM		http://www.expasy.ch/tools/sim-prot.html
Sequence databases via ftp		
Protein:		
GENPEPT		ftp://ncbi.nmlm.nih.gov/genbank/
NRDB (NCBI)		ftp://ncbi.nlm.nih.gov/blast/db/
NRDB90		ftp://ftp.ebi.ac.uk/pub/databases/nrdb90/
OWL		ftp://ftp.seqnet.dl.ac.uk/pub/database/owl/

SWISSPROT	ftp://ftp/ebi/ac.uk/pub/databases/swissprot/
TREMBL	ftp://ftp.ebi.ac.uk/pub/databases/trembl/
Nucleotide:	
EMBL	ftp://ftp.ebi.ac.uk/pub/databases/embl/release/
GENBANK	ftp://ncbi.nlm.nih.gov/genbank/
Multiple sequence databases via WWW	
BLOCKS	http://www.blocks.fhcrc.org/
PFAM	http://www.sanger.ac.uk/Pfam/
PRINTS	http://www.biochem.ucl.ac.uk/bsm/dbbrowser/PRINTS/PRINTS.html
PRODOM	http://protein.toulouse.inra.fr/prodom.html
PROFILESCAN	http://www.isrec.isb-sib.ch/software/PFSCAN_form.html
PROSITE	http://www.expasy.ch/sprot/prosite.html
SMART	http://smart.embl-heidelberg.de/
Database searching programs via ftp	
BLAST	ftp://ncbi.nlm.nih.gov/blast/executables/
HMMER	ftp://ftp.genetics.wustl.edu/pub/eddy/hmmer/
MOST	ftp://ncbi.nlm.nih.gov/pub/koonin/most/
PFSEARCH	http://www/isrec.isb-sib.ch/ftp-server/pftools/pft2.2/
PROBE	ftp://ncbi.nlm.nih.gov/pub/neuwald/probe1.0/
PSI-BLAST	ftp://ncbi.nlm.nih.gov/toolbox/ncbi_tools/
SAM	http://www.cse.ucse.edu/research/compbio/sam.html

SSEARCH/FASTA		ftp://ftp.virginia.edu/pub/fasta/
WISETOOLS		ftp://ftp.sanger.ac.uk/pub/birney/wise2/

Other programs available via ftp

BELVU	(Alignment editing)	ftp://ftp.cgr.ki.se/pub/esr/belvu/
BOXSHADE	(Alignment shading)	ftp://www.isrec.isb-sib.ch/pub/sib-Isrec/boxshade/
CLUSTALW	(Multiple alignments)	ftp://ftp.ebi.ac.uk/pub/software/unix/clustalw/
CLUSTALX	(Multiple alignments)	ftp://ftp.ebi.ac.uk/pub/software/unix/clustalw/clustalx/
DOTTER	(Dot-matix program)	ftp://ftp.cgr.ki.se/pub/esr/dotter/
GDE	(Alignment editing)	ftp://ftp.ebi.ac.uk/pub/software/unix/
MACAW	(Multiple alignments)	ftp://ncbi.nlm.nih.gov/pub/macaw/
READSEQ	(Format conversion)	ftp://ftp.bio.indiana.edu/molbio/readseq/
SEAVIEW	(Alignment editing)	ftp://biom3.univ-lyon1.fr/pub/mol_phylogeny/seaview/

alignment. Many of the latter methods are based on similar algorithms. Differences among the methods are threefold and are due to (1) assumptions made in the method, (2) the manner by which a domain is represented numerically in the method, and (3) parameters indicating the statistical significance of matches reported by the method.

An important consideration is whether a method uses a single sequence or multiple sequences as input: in general, multiple sequence methods, especially when multiple alignments are constructed well, significantly outperform single-sequence methods. However at the onset of an analysis a researcher usually is aware of only a handful of homologues. As a result, single-sequence searches are essential to find sufficient examples of a domain to allow the use of more potent multiple-sequence methods.

3.2. Assumptions Made by the Methods

All methods assume that conservation patterns of different positions in a sequence or alignment are not correlated (principally, this is for algorithmical reasons, as the consideration of such correlations would otherwise result in unreasonably long computation times). Although some methods allow gaps in matched regions of the domain alignment, others do not. Disallowing gaps simplifies the computation and also allows classical analytical statistics to be used to evaluate results ***(3)***. Gapped methods almost invariably rely on dynamic programming methods ***(4)*** or else approximations to them. As might be expected, for gapped methods computation times increase significantly. In addition, the analytical statistical theory does not yet extend to gapped alignments (*see* **Note 3**). In these cases, statistical significance has to be estimated using curve-fitting to assumed distributions or by using a Bayesian framework to provide inference of the outcome of the search (*see* **Note 4**).

3.3. Derivation of the Parameters for the Method

Single-sequence search methods need to estimate the rates by which sequences have mutated during their evolution. These are usually provided by an amino acid substitution matrix (*see* **Note 5**), and also when using gapped alignments, a penalty for gaps (*see* **Note 6**). Multiple-sequence methods need to represent numerically the patterns of conservation in a multiple alignment. Representations of amino acid distributions for each position in the alignment are often called "profiles." A probabilistic interpretation of these profiles in the Hidden Markov Model (HMM), is called a profile-HMM. This does not change the algorithm used but does change the statistical interpretation of the algorithm (*see* **Note 7**). As with single-sequence searches, some multiple sequence methods allow gaps in the alignment of the domain model with each sequence (*see* **Note 8**).

3.4. Statistical Results

Database searching methods report scores for the comparison of a target sequence with a domain model. Usually, methods provide two scores: one that is not an indicator of statistical significance and one that is. The statistical interpretation is an estimate of the likelihood that a sequence with that score or higher is not related to the query sequence or model, and has matched by chance. Note that such statistics are algorithm-specific, and are not highly-related to the real biological significance of the hit: many algorithms detect only a small percentage of true homologues (true positives) with statistical significance. The scores of undetected homologues (false negatives) lie within the distribution of scores for unrelated proteins (true negatives) and so are indistinguishable from the noise. There is always a finite probability that scores for some unrelated proteins (false positives) may be greater than expected. Consequently, sequence analysts should always consider the biological contexts of possible false positives or false negatives, and also consider the results of complementary methods.

Statistical results derived from classical (frequentist) approaches provide some estimate of the probability that a sequence picked at random could have produced a score equal to, or greater than, the score of a real database sequence, i.e., P (score > X). This probability estimate is of little use, as it takes no account of the size of the database searched. Statistics that do, and are commonly reported in database searches, are:

1. Expect-values (or E-values), which represent the number of sequences with scores equal to X, or greater, expected absolutely by chance, which is simply P (score > X) × N, where N is the number of sequences in the database; and,
2. P-values, which represent probabilities, given a database of a particular size, that random sequences score higher than X. This is not the same as P (score > X)!

An estimated E-value of less than 1 indicates possible significance of the hit. In our hands an E-value of less than 0.01 is likely to represent an homologous relationship. This does not mean that all hits with E-values equal to 0.01 are predicted to be real homologues, only that the algorithm estimates that there is a 1% chance that you would have seen a random sequence of this score or higher in this particular database search. Do not, however, have 100% confidence in your algorithm. Be aware that some algorithms (including PSI-BLAST) appear to underestimate E-values by at least one or two orders of magnitude.

3.5. A Recipe for In-Depth Analysis of a Sequence

We suggest a four-step protocol for sequence analysis as follows. Because the requirements for one search often are different from those for another, this

represents one of several possible approaches. An example of the analysis of a protein sequence via a combination of methods has been placed on the Web (`http://www.ocms.ox.ac.uk/~ponting/methmb/example.html`).

3.5.1. Step 1: Motifs, Patterns, and Profile-Scans

Table 1 contains the addresses of several Web sites that allow the prediction of a variety of molecular features from protein sequence information. These include coiled coils ***(5)***, transmembrane helices, protein localization, signal peptides, and glycosylation sites. Care should be taken in interpreting results from these methods: e.g., predicted glycosylation sites in intracellular proteins, and kinase-mediated phosphorylation sites in extracellular proteins, are both extremely unlikely to be of biological relevance. In addition, these methods do not predict homology relationships.

Other Web sites provide easy comparisons of a user's sequence with large numbers of domain or motif (*see* **Note 1**) alignments. If your sequence contains one or more domains that are represented among collections of mutiple alignments, there is a fair chance that they shall be detected. The Profilescan site "`http://www.isrec.isb-sib.ch/software/PFSCAN_form.html`" probably is of most use, as it derives its predictions from three sources, i.e., Prosite, Pfam, and independent collections of alignments. A similar server (SMART; `http://smart.embl-heidelberg.de/`) should be used for domain and motif detection.

Using these servers in combination is the most rapid way to arrive at functional prediction arguments. Successful domain, coiled coil, or transmembrane helix prediction using these methods also serves to reduce the "searchspace": it is valid to concentrate subsequent searches only on sequence regions that are not assigned with significant statistics by these methods.

3.5.2. Step 2: Pairwise Methods

A powerful and popular way to assign domains within a query sequence is via BLAST searches. Early versions of BLAST ***(6)*** provided significance estimates for ungapped pairwise alignments. More recent versions have provided two improvements: pairwise alignments that are gapped, and iterative searches derived from multiple alignments ***(7)***. BLAST searches may be initiated via Web servers (**Fig. 1**) or locally, using code contained in the NCBI toolbox (`ftp://ncbi.nlm.nih.gov/toolbox/ncbi_tools/`). The BLAST family of programs includes: BLASTP, which compares a protein sequence with a protein database; BLASTN, which compares a nucleotide sequence with a nucleotide sequence; BLASTX, which compares a nucleotide sequence (in all reading frames) with a protein database; TBLASTN, which compares a protein

sequence with a nucleotide sequence (in all reading frames); and TBLASTX, which compares a nucleotide sequence (in all reading frames) with a nucleotide database (in all reading frames). Current versions of BLAST provide significance estimates in terms of *E*-values (*see* **Note 3**). An alternative to BLAST is SSEARCH ***(8)*** which also ascribes *E*-values to candidate homologues.

Several considerations must be borne in mind during such analyses. The first is that several nonoverlapping hits to different portions of your query sequence may indicate that it contains multiple domains. If this appears to be the case, then scan your BLAST output for hits from sequences contained in the protein databank (PDB) (a database of known protein structures) or else initiate a BLAST search against the set of PDB sequences (try using the SCOP resource: `http://scop.mrc-lmb.cam.ac.uk/scop/`). A significant hit against a domain of known tertiary structure immediately provides an accurate prediction of domain boundaries for its homologue contained in your query sequence (*see* **Note 9**). Similarly, a significant hit against the whole of a polypeptide with bone fide N- and C-termini also provides accurate domain limits. As previously, successful prediction of a domain allows reduction of the searchspace in subsequent searches.

Second, the existence of compositionally biased regions in (predominantly eukaryotic) proteins, such as occurs in collagens, might distort the reported significance estimates of your findings. Ensure that such "low-complexity regions" are masked in your query sequence by preprocessing with the program SEG ***(9)***, which is the default option for the NCBI-BLAST servers. Preprocessing your sequence with transmembrane helix and coiled coil prediction algorithms (*see* **Subheading 3.5.1.**) also is highly recommended (*see* **Note 10**). However, such algorithms provide imperfect predictions and regions that are not assigned as coiled coil may yield low *E*-values against known coiled coil regions in database proteins such as myosins and kinesins; such alignments are unlikely to be biologically relevant.

Third, be vigilant against errors. These can be present in all quarters. Your query sequence or a database sequence may contain one or more frameshift errors, or they may be artificially truncated or extended beyond the proteins normal termini. Hypothetical protein sequences deduced from eukaryotic genome projects frequently are inaccurate as a result of errors in intron–exon boundary assignments. Such errors often are apparent on construction of a multiple alignment of homologues: the error-ridden sequence usually demonstrates substantial nonconservation in regions that are relatively well-conserved among its homologues. A strategy to combat frameshifts and unknown intron–exon boundaries is to compare a profile derived from either a similar sequence or else a multiple alignment of homologues, with the relevant DNA sequence using PAIRWISE, as described elsewhere ***(10)***. Finally, errors are not restricted

to sequence. Functions of proteins assigned by some are found frequently to be at odds with those found by others. Unfortunately, this results not only in errors in the database annotation for the protein in question, but also propagation of the error to all homologues whose functions are predicted on the basis of the original misannotation.

3.5.3. Step 3: Constructing a Multiple Alignment

At some stage in any sequence analysis project, a multiple alignment must be constructed. Constructing an optimal multiple alignment consists of two stages: calculation of a preliminary alignment using programs such as CLUSTALW, and its subsequent refinement using manual alignment editors such as CLUSTALX, GDE, and/or SEAVIEW (*see* **Fig. 1**). Manual intervention in otherwise automated procedures is an unfortunate consequence of the "alignment problem": algorithms are currently unable to generate alignments with high accuracies, in comparison with alignments generated from superimposed tertiary structures of homologues. This is due to the exponential increase in memory and computing power required to consider mathematically-optimal alignments (*see* **Note 11**). However, manual editing can greatly improve alignments if the following guidelines (cf. **ref.** ***2***) are followed:

1. Minimize the number of gaps in an alignment. Consider that insertions and deletions occur predominantly on the exterior surface of proteins and in loops *between* secondary-structures. Loops are usually highly variable in structure and in sequence and so need not be well-aligned. Ensure that those residues in a loop region between two alignment blocks that are not alignable are shunted (without gaps) to the alignment block that is either N-terminal or C-terminal to it. Single-residue insertions can occur within β-strands as "β-bulges." α-Helices are relatively intolerant of insertions or deletions (although over- or underwinding of helices can occur). In a small minority of cases an insertion/deletion position in a loop may accommodate a whole domain structure as a "domain insertion" ***(11)***.
2. Maximize conservation of "core" hydrophobic residues. Hydrophobic residues comprise the majority of proteins interiors and therefore are subject to relatively strong evolutionary pressures. Ensure that hydrophobic residues in homologues are aligned within all predicted secondary-structures. Note that the periodicity of hydrophobic residues in β-strands differs from that in α-helices, which provides information essential for secondary-structure prediction. Domains that are rich in cysteine residues, such as several small extracellular domains and nuclear zinc fingers, are exceptions in that the bulk of their hydrophobic core is provided by disulphide bridges and metal ions, respectively. These domains are often characterized by poor amino acid conservation and few secondary-structures relative to their size.

3. Maximize conservation of residues known through experiment to be important for function. However, consider that homologues can possess non-identical functions: there are many instances, e.g., of enzyme homologues that are enzymatically inactive.
4. Minimize the presence of Pro and Gly in the middle of all secondary-structures except for edge β-strands, and maximize the presence of strings of charged residues to insertion/deletion positions.
5. Ensure that all subfamilies in an alignment are represented equally by including only one of each pair of sequences that are greater than, e.g., 80% identical (the program BELVU provides such an option).
6. Take care in the choice of domain boundaries. The best evidence for these boundaries are experimentally determined N- and C-terminal residues of the protein, or else detection of an homologue with known tertiary structure. Other indicators to be considered are limits of domains contiguous to the sequence of interest, degrees of sequence conservation (or lack of conservation) between closely related homologues, and the presence of low-complexity regions (normally present within interdomain, rather than intradomain, regions). Tandem repeats, as indicated using dot-plot algorithms such as Dotter, may also help in determining boundaries.

3.5.4. Step 4: Multiple Sequence Analysis

Once a multiple alignment has been created to your satisfaction, then its representation (either a profile or a profile-HMM (*see* **Note 7**) can be compared with protein sequence databases. There are several tools that have been written to search databases (**Table 1**), with which two (WISETOOLS ***[10]*** and HMMER ***[12]***) the authors are most familiar. A profile, constructed using PAIRWISE of the WISETOOLS suite ***(10)***, may be either "negative" or "positive": i.e., it may be intended to search databases for similarities with either a portion of the alignment (a "local" similarity) or the entire alignment (a "global" similarity). A profile-based search may be initiated directly using SWISE, or else indirectly using the menu-driven SEARCHWISE. Both SWISE and HMMER report *E*-values from Extreme Value distribution curve fitting (S. Eddy, personal communcation) (*see* **Note 3**). A profile-HMM (*see* **Note 12**) may be compared with databases in searches for local similarities or else global similarities; similarity scores are represented as bits scores (*see* **Note 4**). HMMER is recommended when searching for domains that occur multiply within the same polypeptide chain. As a result of these searches the sequences of proposed domain homologues should be realigned and searches should recommence in an iterative manner until no further candidate homologues are revealed.

In our experience, no single database searching method detects all bona fide homologues that are detected by the sum of searching methods. Consequently,

candidate homologues suggested using a single method such as SWISE or HMMER should be compared with those indicated by other methods. In particular, the Position-Specific Iterative BLAST (PSI-BLAST) program ***(7)*** allows a database search with a query sequence, with which all candidate homologues (with low *E*-values; recommended threshold $E < 0.001$) are aligned, and provides iterative comparisons with profiles derived from this and subsequent alignments, until convergence. Due to the alignment problem discussed previously, derived alignments are suboptimal. However, PSI-BLAST has proven to be highly-sensitive in revealing subtle homologies and therefore is a method of choice in detecting domain homologues. As discussed previously for other versions of BLAST, the query sequence should be chosen to exclude transmembrane, low-complexity, and coiled coil regions, as well as domain sequences that are irrelevant to the search.

Single ungapped motifs (defined in **Note 1**), such as those encompassing active or binding sites, may also be compared with databases using the Motif Searching Tool, MOST ***(13)***. This also is an iterative method, and allows choice of candidate homologues on the basis of *E*-values (option: e; recommendation, e 0.05), and closely similar sequences to be discarded (option: i; recommendation, i 80%). PROBE ***(14)*** is a tool that combines an initial BLAST search with iterations of multiple motif searches on the basis of *E*-values until convergence is achieved.

It is imperative that a hypothesis that two sequences represent domain homologues is justified statistically. This may be achieved using an *E*-value threshold and programs such as MOST and PSI-BLAST. However the user should be aware that the inclusion of a false-positive scoring lower than the supplied *E*-value threshold following one iteration negates the identification of putative homologues detected in subsequent iterations. It is preferable that all candidate homologues are related by multiple instances of low *E*-values from BLAST (or SSEARCH) queries, and essential that their sequences display similarities in their patterns of conservation across their multiple alignment.

Non-statistically based evidence may provide information consistent with a homology hypothesis. Experimental and contextual information may be used to predict homology for sequences that score at levels similar to the top true negative in searches. For example, a predicted domain may possess limits exactly coincident with the boundaries of a region intervening between two properly annotated domains; or, the predicted domain may be known experimentally to possess a molecular function equivalent with the functions of the proteins used to derive the query profile. In contrast, non-statistically based evidence may provide evidence that two sequences are *not* homologues. A sequence may score highly against a query profile, yet the pattern of conservation and/or domain limits apparent from the alignment of its close homologues

may differ substantially from the conservation pattern of the query profile's alignment.

So far, we have described the comparison of protein sequences with protein databases. Additional information that is absent from protein databases may be gleaned from their nucleotide counterparts. It is advised that protein sequences (using TBLASTN) or profiles (using SWISE) be compared directly to GENBANK and to the databases of expressed sequence tags (ESTs), sequence tagged sites (STS), high-throughput genomic sequences (HTGS) and genome survey sequences (GSS). A set of sequences that are predicted to be homologues on the basis of statistical and experimental or contextual information may then be used as the basis for experimentally testable hypotheses concerning function (*see* **Note 13**).

4. Notes

1. Here we employ a terminology discussed elsewhere ***(2)***. In short: motifs are short, conserved regions that are short stretches of domain sequences (e.g., "binding-site motifs"); patterns are assemblies of one or more motifs; alignment blocks are ungapped alignments usually representing a single secondary-structure; and, domains are conserved structural entities with distinctive secondary-structures and an hydrophobic core. Thus, these terms are not mutually exclusive and, in particular instances, are interchangeable.
2. Databases are best maintained locally in a simple (FASTA) format of concatenated sequences separated by a single header line containing a ">" symbol, accession codes, and relevant species, gene, and molecular information. Sequence databases are often redundant: they contain several copies of identical sequences. Databases that are less redundant than others are SWISSPROT, OWL, and NCBI's NRDB. The SWISSPROT database is recommended for its extensive annotation (*see* `http://www.expasy.ch/sprot/sp-docu.html`), whereas GenPept is recommended for its daily updates (*see* `ftp://ncbi.nlm.nih.gov/genbank/daily-nc/`). All database entries can be accessed via the highly informative Entrez system (*see* `http://www.ncbi.nlm.nih.gov/Entrez/index.html`), which provides links between sequence, literature, structure, and taxonomy information.
3. Ungapped BLAST algorithms ***(6)*** and the motif-searching tool MOST ***(13)*** use analytically derived statistics from the theory of Karlin and Altschul ***(3)***. This predicts that the distribution of scores of ungapped alignments should follow an Extreme Value Distribution (EVD) with parameters that can be used to provide P (score $> X$), and hence E and P values. Extending this theory to gapped searches is problematic; it has been suggested that scores from gapped alignments also vary according to an EVD ***(7,15)***. In this approximation, significance statistics may be estimated from the fit of parameters to presumed random sequences. These are either precalculated using a comparison to artificial random databases (as with gapped BLAST methods), or generated "on the fly" from the distribution

of scores from presumed non-homologues in the same database search (the "noise") (as with FASTA and SSEARCH).

4. Inference on the probability of a match being a random sequence or not can be derived using Bayesian methods for profile-HMMs ***(12,16)***. The profile-HMM provides a likelihood that the sequence was an example of the domain. This likelihood is compared to the likelihood that the sequence was generated by a "random" model. Profile-HMMs use a simple random model based on the distribution of amino acids drawn at random from a set of real sequences. The score reported by the program is a log likelihood ratio (the base of the logarithm is 2, hence the name "bits score"). This implies that bit scores between different profile-HMMs are comparable, in contrast to other statistics. For the likelihood ratio to provide a probability that the sequence came from either the domain HMM or the random model, estimates of the probability of these outcomes *before* examining the sequence have to be made ***(12,16)***. In general, these probabilities are not explicitly defined, but rather are converted to an ad hoc threshold for the bits score, over which the search is considered significant. This threshold is suggested to be around 25 bits ***(12)***, although there are cases where very different thresholds are applicable, in particular, small or all α-helical domains require higher threshold (35 bits).
5. The amino acid substitution matrix represents what is known of the results of protein evolution: in trusted multiple alignments, pairs of amino acids that are often aligned against each are given higher scores relative to pairs of amino acids that are seldom paired in alignments. These scores are in qualitative agreement with known physical and chemical properties of amino acids. Numbers in matrices can be interpreted as log-odd ratios of the observed frequency of a pairing, relative to the frequency expected by chance. Commonly used score matrices are the BLOSUM ***(17)*** and Gonnet ***(18)*** series.
6. Gap penalties are a necessity, as mathematically optimal alignments would otherwise be dominated by excessively gapped regions. There are two main forms of a gap penalty. Linear gap penalties are used when individual gaps are penalized separately. Affine gap penalties, which are more common, are used when each gapped region is penalized both for the initiation of the gap and also for its extension when additional gap positions are needed. Unfortunately, there is no analytical theory that reliably calculates "optimal" gap penalties. Thus many programs allow the user complete freedom in setting penalties. However, a series of studies using both random databases and databases of known structure have provided some emperically derived optima ***(15,19)***. The optimal gap penalty is linked to the comparison matrix used. For the common BLOSUM62 matrix, gap penalties of 12 for initiation of a gap and 2 for its extension are considered reasonable.
7. A trivial representation of amino acid conservation in a multiple alignment is the set of amino acid frequencies at each position. However, this is of little practical use because (1) multiple alignments are often strongly biased toward the detection of a particular subfamily of homologues, and (2) this provides insufficient information concerning amino acid substitution to identify more distant homo-

logues. The problem of sequence bias is addressed by calculating weights for each sequence in the multiple alignment using a derived phylogenetic tree ***(20)***: similar sequences are weighted lower than dissimilar ones. A model of evolution is used to supplement the observed amino acid frequencies in the multiple alignment. In standard profile-based methods ***(20,21)*** the same amino acid substitution matrices that are used in single sequence searches are used. In HMM profiles the model of evolution is represented as an undersampling problem. This provides a consistent mathematical framework that combines frequencies observed in the alignment and an evolutionary model. Two models have been found to be useful. The first is the equivalent to the standard profile techniques recast to fit the Bayesian framework. The second is a model based around Dirichlet mixtures of expected amino acid distributions in multiple alignments ***(22)***.

8. Gaps generated in aligning a domain model with a target sequence are penalized to prevent an unrealistic overgapped alignment. In profile methods, gap penalties are set arbitrarily, in a similar manner to single-sequence searches. The penalties are multipled by an ad hoc position specific ratio indicating the tolerance of insertions or deletions at this position. HMM-profiles have a stronger theoretical basis for setting gap penalties, being the best fit to an assumed distribution from the observed gap length at each position. As with the raw frequency of amino acids, the observed gap-length distribution is considered to be an undersampling of real allowed gaps at each position, and the observed frequencies are modified by a model of gap evolution.
9. In rare cases, the order of secondary-structures in pairs of homologous domains is known to be circularly permuted ***(23)***. Identifying such homologues and determination of their domain limits is not a trivial task using conventional methods.
10. In a few cases (e.g., **ref. *24***), conserved domains are known to contain coiled coils.
11. Methods used for multiple alignment algorithms are either (1) progressive alignment procedures where the multiple alignment proceeds up an evolutionary tree, fixing the alignment at each node (as in CLUSTALW, PILEUP), or (2) iterative training procedures that derive a model of the resulting multiple alignment (as in HMMER, MOST).
12. The default parameters for HMMER (version 1) rarely produce good HMMs. In our hands both the weighting of sequences (option: -w), and an ad hoc PAM prior from BLOSUM62 (option: -P blosum62.bla) are required. HMMs generated by version 2 of HMMER using default parameters are reliable.
13. Homologues from different organisms that have sequences more similar to each other than they are to those of other homologues (i.e., "orthologues" ***[25,26]***) are likely to perform essentially identical molecular and cellular functions, whereas homologues from the same organism (i.e., "paralogues" ***[25,26]***) are more likely to perform dissimilar cellular functions, even if their molecular functions are comparable. More distant homologues are more likely to perform dissimilar — albeit related — molecular or cellular functions than are closely related ones: e.g., a divergent *Escherichia coli* homologue of mammalian phospholipases D

(that are phosphodiesterases) is known to be an endonuclease (also a phosphodiesterase) ***(27,28)***.

Addendum

Since the original draft of this chapter, three major advances have in 3 identifying protein sequences.

1. PSI-BLAST and HMMER (version 2) have been as pre-eminent methods for database searching.
2. The WISE2 suite (http://www.sanger.ac.uk/software/wise21) is now available for the cross-comparison of protein and DNA sequences using HMM-based methods.
3. Domain databases, such as PFAM and SMART have considerably reduced the arduous task of delineating domains in complex multidomain architectures.

Acknowledgments

Chris B. Ponting is a Wellcome Trust Fellow and a member of the Oxford Centre for Molecular Sciences. The authors thank Alex Bateman and Andy May for critical reading of the manuscript.

References

1. Doolittle, R. F. (1995) The multiplicity of domains in proteins. *Annu. Rev. Biochem.* **64,** 287–314.
2. Bork, P. and Gibson, T. J. (1996) Applying motif and profile searches. *Methods Enzymol.* **266,** 162–184.
3. Karlin, S. and Altschul, S. F. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA* **87,** 2264–2268.
4. Pearson, W. R. and Miller, W. (1992) Dynamic programming algorithms for biological sequence comparison. *Methods Enzymol.* **210,** 575–601.
5. Lupas, A. (1996) Coiled coils: new structures and new functions. *Trends Biochem. Sci.* **21,** 375–382.
6. Altschul, S. F., Boguski, M. S., Gish, W., and Wootton, J. C. (1994) Issues in searching molecular sequence databases. *Nat. Genet.* **6,** 119–129.
7. Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25,** 3389–3402.
8. Pearson, W. R. (1991) Searching protein sequence libraries: comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics* **11,** 635–650.
9. Wootton, J. C. and Federhen, S. (1996) Analysis of compositionally biased regions in sequence databases. *Methods Enzymol.* **266,** 554–571.
10. Birney, E., Thompson, J. D., and Gibson, T. J. (1996) PairWise and SearchWise: finding the optimal alignment in a simultaneous comparison of a protein profile against all DNA translation frames. *Nucleic Acids Res.* **24,** 2730–2739.

11. Russell, R. B. (1994) Domain insertion. *Protein Eng.* **7,** 1407–1410.
12. Eddy, S. R. (1996) Hidden Markov models. *Curr. Opin. Struct. Biol.* **6,** 361–365.
13. Tatusov, R. L., Altschul, S. F., and Koonin, E. V. (1994) Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proc. Natl. Acad. Sci. USA* **91,** 12,091–12,095
14. Neuwald, A. F., Liu, J. S., Lipman, D. J., and Lawrence, C. E. (1997) Extracting protein alignment models from the sequence database. *Nucleic Acids Res.* **25,** 1665–1677.
15. Altschul, S. F. and Gish, W. (1996) Local alignment statistics. *Methods Enzymol.* **266,** 460–480.
16. Krogh, A., Brown, M., Mian, I. S., Sjolander, K., Haussler, D. (1994) Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.* **235,** 1501–1531.
17. Henikoff, S. and Henikoff, J. G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA* **89,** 10,915–10,919.
18. Benner, S. A., Cohen, M. A., and Gonnet, G. H. (1994) Amino acid substitution during functionally constrained divergent evolution of protein sequences. *Protein Eng.* **7,** 1323–1332.
19. Brenner, S. E., Chothia, C., and Hubbard, T. J. P. (1998) Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci.* **95,** 6073–6078.
20. 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.
21. Gribskov, M. and Veretnik, S. (1996) Identification of sequence pattern with profile analysis. *Methods Enzymol.* **266,** 198–212.
22. Karplus, K. (1995) Evaluating regularizers of estimating distributions of amino acids. *Ismb* **3,** 188–196.
23. Lindqvist, Y. and Schneider, G. (1997) Circular permutations of natural protein sequences: structural evidence. *Curr. Opin. Struct. Biol.* **7,** 422–427.
24. Weimbs, T., Low, S. H., Chapin, S. J., Mostov, K. E., Bucher, P., and Hofmann, K. (1997) A conserved domain is present in different families of vesicular fusion proteins: a new superfamily. *Proc. Natl. Acad. Sci. USA* **94,** 3046–3051.
25. Fitch, W. M. (1970) Distinguishing homologues from analogous proteins. *Syst. Zool.* **19,** 99–113.
26. Fitch, W. M. (1995) Uses for evolutionary trees. *Philos. Trans. R. Soc. Lond. B Biol. Sci.* **349,** 93–102.
27. Ponting, C. P. and Kerr, I. D. (1996) A novel family of phospholipase D homologues that includes phospholipid synthases and putative endonucleases: identification of duplicated repeats and potential active site residues. *Protein Sci.* **5,** 914–922.
28. Koonin, E. V. (1996) A duplicated catalytic motif in a new superfamily of phosphohydrolases and phospholipid synthases that includes poxvirus envelope proteins. *Trends Biochem. Sci.* **21,** 242–243.

5

Third Generation Prediction of Secondary Structures

Burkhard Rost and Chris Sander

1. Introduction

The sequence–structure gap is rapidly increasing. Currently, databases for protein sequences (e.g., SWISS-PROT ***[1]***) are expanding rapidly, largely due to large-scale genome sequencing projects: at the beginning of 1998, we know already all sequences for a dozen of entire genomes ***(2)***. This implies that despite significant improvements of structure determination techniques, the gap between the number of protein structures in public databases (PDB ***[3]***), and the number of known protein sequences is increasing. The most successful theoretical approach to bridging this gap is homology modeling. It effectively raises the number of "known" 3D structures from 7000 to over 50,000 ***(4,5)***.

No general prediction of structure from sequence, yet. John Moult (Center for Advance Research in Biotechnology [CARB], Washington) has initiated an important experiment: those who determine protein structures submitted the sequences of proteins for which they were about to solve the structure to a "to-be-predicted" database; for each entry in that database predictors could send in their predictions before a given deadline (the public release of the structure); finally, the results were compared, and discussed during a workshop (in Asilomar, CA). The results of the first two critical assessment of protein structure prediction (CASP) experiments ***(6,7)*** demonstrated clearly that we still cannot predict structure from sequence.

Simplifying the structure prediction problem. The rapidly growing sequence–structure gap has enticed theoreticians to solve simplified prediction problems ***(4)***. An extreme simplification is the prediction of protein structure in one dimension (1D), as represented by strings of, e.g., secondary-structure or residue solvent accessibility. Theoreticians are lucky in that a simplified

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

predictions in 1D (e.g., secondary-structure or solvent accessibility ***[4,8,9]***) even when only partially correct — are often useful, e.g., for predicting protein function, or functional sites.

In this review we focus on recent secondary-structure prediction methods (for reviews on older methods ***(10–17)***, for reviews on other prediction methods in 1D ***[4,5,18]***). We present some of the new, successful concepts and a few "hints for the user" based on the currently most widely used secondary-structure prediction method: PHD.

2. Materials

Assignment of secondary-structure. Secondary-structure is most often assigned automatically based on the hydrogen bonding pattern between the backbone carbonyl and NH groups (e.g., by Dictionary of Secondary Structure assignment of Proteins [DSSP] ***[19]***). DSSP distinguishes eight secondary-structure classes which are often grouped into three classes: H = helix, E = strand, and L = non-regular structure. Typically the grouping is as follows: H (α-helix) -> H, G (3_{10}-helix) -> H, I (π-helix) -> H, E (extended strand) -> E, and B (residue in isolated b-bridge) -> E, T (turn) -> L, S (bend) -> L, (blank = other) -> L, with the "corrections": B -> EE, but B_B -> LLL. Note that developers often use different projections of the eight DSSP classes onto three predicted classes; most of these yield seemingly higher levels of prediction accuracy. For example, short helices are more difficult to predict ***(20)*** (*see also* **Fig. 5**); thus, converting GGG -> LLL results, on average, in higher levels of prediction accuracy.

Per-residue prediction accuracy. The simplest and most widely used score is the three-state-per-residue accuracy, giving the percentage of correctly predicted residues predicted correctly in either of the three states: helix, strand, other:

$$Q_3 = 100 \cdot \sum_{i=1}^{3} c_i / N \qquad (1)$$

where c_i is the number of residues predicted correctly in state i (H, E, L), and N is the number of residues in the protein (or in a given data set). Because typical data sets contain about 32% H, 21% E, and 47% L, correct prediction of the nonregular class tends to dominate the three-state accuracy. More fine-grained methods that avoid this shortcoming are defined in detail elsewhere ***(21,22)***.

Per-segment prediction accuracy. Measures for single-residue accuracy do not completely reflect the quality of a prediction ***(14,22–26)***. There are three simple measures for assessing the quality of predicted secondary-structure segments: (1) the number of segments in the protein, (2) the average segment length, and (3) the distribution of the number of segments with length ***(27)***. These measures are related. They are useful in characterizing prediction meth-

ods, in particular, methods with fairly high per-residue accuracy, yet an unrealistic distribution of segments. However, there is a more elaborated score base on the overlap between predicted and observed segments ***(22)***.

Conditions for evaluating sustained performance. A systematic testing of performance is a precondition for any prediction to become reliably useful. For example, the history of secondary-structure prediction has partly been a hunt for highest accuracy scores, with over-optimistic claims by predictors seeding the skepticism of potential users. Given a separation of a data set into a training set (used to derive the method) and a test set (or crossvalidation set, used to evaluate performance), a proper evaluation (or crossvalidation) of prediction methods needs to meet four requirements: (1) no significant pairwise sequence identity between proteins used for training and test set, i.e., < 25% (length-dependent cutoff ***[28]***); (2) all available unique proteins should be used for testing, as proteins vary considerably in structural complexity; certain features are easy to predict, others harder; (3) no matter which data sets are used for a particular evaluation, a standard set should be used for which results are also always reported; (4) methods should never be optimized with respect to the data set chosen for final evaluation. In other words, the test set should never be used before the method is set up.

Number of crossvalidation experiments of NO meaning. Most methods are evaluated in *n*-fold crossvalidation experiments (splitting the data set into *n* different training and test sets). How many separations should be used, i.e., which number of *n* yields the best evaluation? A misunderstanding is often spread in the literature: the more separations (the larger *n*) the better. However, the exact number of *n* is not important provided the test set is representative, and comprehensive and the crossvalidation results are not misused to again change parameters. In other words, the choice of *n* has no meaning for the user.

3. Methods

3.1. The Dinosaurs of Secondary Structure Prediction Are Still Alive

First generation: single-residue statistics. The first experimentally determined 3D structures of hemoglobin and myoglobin were published in 1960 ***(29,30)***. Almost a decade before, Pauling and Corey suggested an explanation for the formation of certain local conformational patterns such as α-helices and β-strands ***(31,32)***. Shortly later (and still prior to the first published structure), the first attempt was made to (positively) correlate the content of certain amino acids (e.g., proline) with the content of an α-helix ***(33)***. The idea was expanded to correlate the content for all amino acids with that of the α-helix and the β-strand structure ***(34,35)***. The field of predicting secondary-structures had been opened. Most methods of the first generation based on single-residue

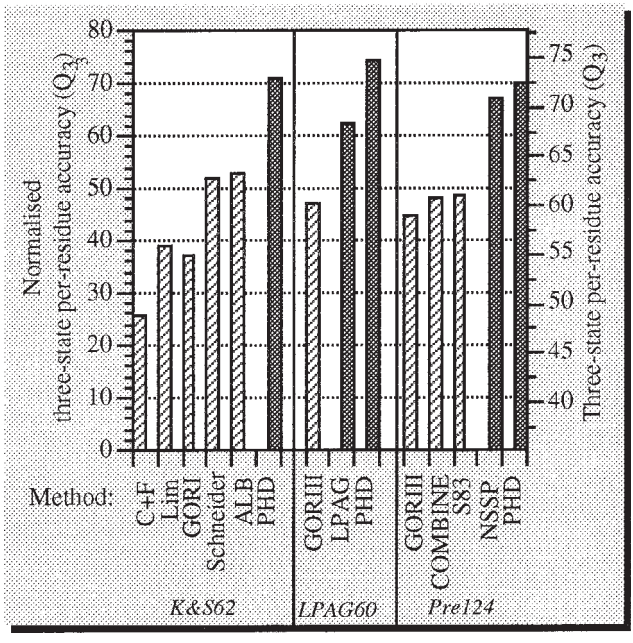


Fig. 1. Three-state-per-residue accuracy of various prediction methods. Shaded bars: methods of first and second generation; filled bars: methods of third generation. The left axis showed the normalized three-state-per-residue accuracy, for which a random prediction would rate 0%, and an optimal prediction by homology modeling would rate as 100% (unnormalized values according to **Eq. 1**, shown on the right axis):

$$\text{norm } Q3 = 100 \cdot Q_3^{\text{method}} - Q_3^{\text{RAN}} / Q_3^{\text{HM}} - Q_3^{\text{RAN}} \quad (1)$$

$$\text{with } Q_3^{\text{HM}} = 88.4\%, \text{ and } Q_3^{\text{RAN}} = 35.2\%$$

Only methods were included for which the accuracy had been compiled based on comparable data sets, the sets in particular are *K&S62*, 62 proteins taken from **ref. *45***; *LPAG60*, 60 proteins taken from **ref. *128***; *Pre124*, 124 unique proteins taken from **ref. *48***. The methods were: *C + F* Chou & Fasman (first generation) ***(42,148)***; *Lim* (first) ***(43)***; *GORI* (first) ***(53)***; *Schneider* (second) ***(87)***; *ALB* (second) ***(62)***; *GORIII* (second) ***(54)***; *LPAG* (third) ***(128)***; *COMBINE* (second) ***(17)***; *S83* (second) ***(86)***; *NSSP* (third) ***(84)***; *PHD* (third) ***(48)***. Most values were recompiled — only those for NSSP and LPAG were taken from the original publications. The scores for PHD on the three different data sets illustrated that data sets can be tuned to give more optimistic (LPAG62), or more realistic estimates for prediction accuracy. The first two structure prediction contests have indicated that the most conservative estimates of this graph (Pre124) tend to be slightly too optimistic, still PHD rates at an average accuracy of about 72% (as originally estimated ***[18,48]***).

statistics, i.e., from the limited databases evidence was extracted for the preference of particular residues for particular secondary-structure states ***(36–44)***. By 1983, it became clear that the performance accuracy had been overstated ***(45)*** (*see* **Fig. 1**).

Second generation: segment statistics. The principal improvement of the second generation of prediction tools was a combination of a larger database of protein structure and the usage of statistics based on segments: typically 11–21 adjacent residues were taken from a protein and statistics were compiled to evaluate how likely the residue central in that segment was in a particular secondary-structure state. Similar segments of adjacent residues were also used to base predictions on more elaborated algorithms, some of which were spun off from artificial intelligence ***(46)***. Almost any algorithm has meanwhile been applied to the problem of predicting secondary-structures; all were limited to accuracy levels slightly higher than 60% (*see* **Fig. 1**; reports of higher levels of accuracy were usually based on too small, or non-representative data sets ***[21,25,47,48]***). The most-used algorithms were based on (1) statistical information ***(49–61)***, (2) physicochemical properties ***(62)***, (3) sequence patterns ***(63–65)***, (4) multilayered (or neural) networks ***(66–73)***, (5) graph theory ***(74,75)***, (6) multivariate statistics ***(76,77)***, (7) expert rules ***(75,78–82)***, and (8) nearest-neighbor algorithms ***(83–85)***.

Problems with first- and second-generation methods. All methods from the first and second generation shared, at least, two of the following problems (most all three): (1) three-state per-residue accuracy was below 70%, (2) β-strands were predicted at levels of 28–48%, i.e., only slightly better than random, and (3) predicted helices and strands were too short.

The first problem (<100% accuracy) has two sources: (1) secondary-structure assignments differ even between different crystals of the same protein, and (2) secondary-structure formation is partially determined by long-range interactions, i.e., by contacts between residues that are not visible by any method based on segments of 11–21 adjacent residues. The second problem (β-strands <50% accuracy) has been explained by the fact that b-sheet formation is determined by more nonlocal contacts than is α-helix formation. The third problem was basically overlooked by most developers (for exceptions, *see* **refs. *86*** and ***87***). This problem makes predictions very difficult to use in practice (*see* **Fig. 2**). As we show in the next paragraph, some of the prediction methods of the third generation address all three problems simultaneously, and are clearly superior to the old methods (*see* **Fig. 1**). Nevertheless, many of the secondary-structure prediction methods available today (e.g., in University of Wisconsin Genetics Computer Group (GCG) ***[88]***, or from Internet services ***[89]***) are unfortunately still using the dinosaurs of secondary-structure prediction.

```
SEQ   KELVLALYDYQEKSPREVTMKKGDILTLLNSTNKDWWKVEVNDRQGFVPAAYVKKLD
OBS     EEEE          E--E    EEEEEE      EEEEEE  EEEEEEEHHHEEEE
TYP     EEHHH         EE    EEEE   EE     HHHEE       EEEHH
```

Fig. 2. Example for typical secondary structure prediction of the second generation. The protein sequence (*SEQ*) given was the SH3 structure ***(131)***. The observed secondary structure (*OBS*) was assigned by DSSP ***(19)*** (H = helix; E = strand; blank = nonregular structure; the dashes indicate the continuation of the second strand that was missed by DSSP). The typical prediction of too short segments (*TYP*) poses the following problems in practice: (1) Are the residues predicted to be strand in segments 1, 5, and 6 errors, or should the helices be elongated? (2) Should the second and third strand be joined, or should one of them be ignored, or does the prediction indicate two strands here? Note: the three-state-per-residue accuracy is 60% for the prediction given.

3.2. Breakthrough By Using Evolutionary Information

3.2.1. Is Evolutionary Odyssey Informative?

Variation in sequence space. The exchange of a few residues can already destabilize a protein ***(90)***. This implies that the majority of the 20^N possible sequences of length N form different structures. Has evolution really created such an immense variety? Random errors in the DNA sequence lead to a different translation of protein sequences. These "errors" are the basis for evolution. Mutations resulting in a structural change are not likely to survive, as the protein can no longer function appropriately. Furthermore, the universe of stable structures is not continuous: minor changes on the level of the 3D structure may destabilize the structure (due to high complexity). Thus, residue exchanges conserving structure are statistically unlikely. However, the evolutionary pressure to conserve function has led to a record of this unlikely event: structure is more conserved than sequence ***(91–93)***. Indeed, all naturally evolved protein pairs that have 35 of 100 pairwise identical residues have similar structures ***(28,94)***. However, the attractors of protein structures are even larger: the majority of protein pairs of similar structures has levels of below 15% pairwise sequence identity ***(95,96)***.

Long-range information in multiple sequence alignments. The residue substitution patterns observed between proteins of a particular family, i.e., changes that conserved structure, are highly specific for the structure of that family. Furthermore, multiple alignments of sequence families implicitly also contain information about long-range interactions: suppose residues i and i + 100 are close in 3D, then the types of amino acids that can be exchanged (without changing structure) at position i are constrained by that their physicochemical characteriztics have to fit the amino acid types at position i + 100 ***(97,98)***.

3.2.2. Can Evolutionary Information Be Used?

Expert predictions: visual use of alignment information. The first method that used information from family alignments was proposed in the 1970s already ***(99)***. Furthermore, experts have based single-case predictions successfully on multiple alignments ***(99–116)***.

Automatic use of alignment information. The simplest way to use alignment information automatically was first proposed by Maxfield and Scheraga and by Zvelebil et al. ***(117,118)***: predictions were compiled for each protein in an alignment, and then averaged over all proteins. A slightly more elaborated way of automatically using evolutionary information is to directly base prediction on a profile compiled from the multiple sequence alignment ***(18,21,48)***. The following steps are applied in particular for the PHD method ***(18,119)*** (*see* **Fig. 3**): (1) A sequence of unknown structure (*U*) is quickly aligned against the database of known sequences (typically by BLAST ***[120]***) (i.e., no information of structure required); (2) proteins with sufficient sequence identity to *U* to assure structural similarity are extracted and realigned by a multiple alignment algorithm *MaxHom* ***(121)***; (3) for each position, the profile of residue exchanges in the final multiple alignment is compiled, and used as input to a neural network.

3.2.3. Third Generation: Evolution to Better Predictions

Example chosen: PHD. We illustrated the principal concepts of third generation methods based on the particular neural network-based method PHD because it is currently the most accurate method ***(7)***, and because most of these concepts were introduced by this method ***(21,48)***. Meanwhile, several other methods have reported and/or achieved similar levels of performance ***(16,18,21,48,84,114,122–129)***.

Multiple levels of computations. PHD processes the input information on multiple levels (*see* the neural network in **Fig. 3**). The first level is a feed-forward neural network with three layers of units (input, hidden, and output). Input to this first-level sequence-to-structure network consists of two contributions: one from the local sequence, i.e., taken from a window of 13 adjacent residues, and another from the global sequence. Output of the first-level network is the 1D structural state of the residue at the center of the input window. The second level is a structure-to-structure network. The next level consists of an arithmetic average over independently trained networks (jury decision). The final level is a simple filter.

Balanced predictions by balanced training. The distribution of the training examples (known structures) is rather uneven: about 32% of the residues are observed in helix, 21% in strand, and 47% in loop. Choosing the training

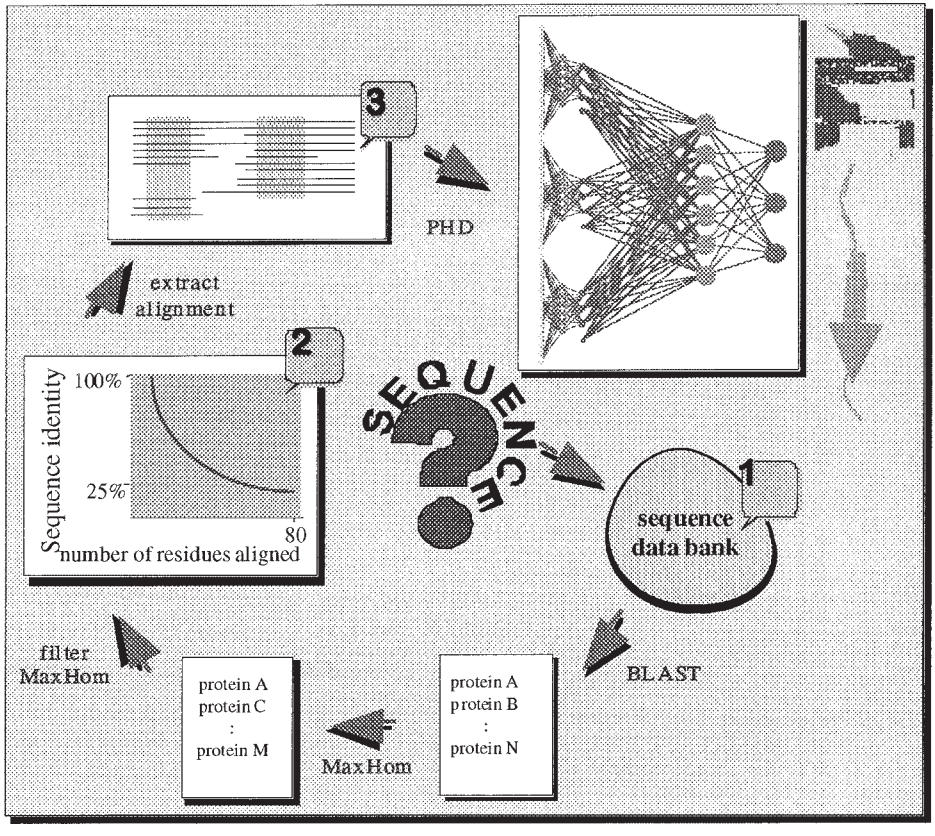


Fig. 3. Using evolutionary information to predict secondary structure. Starting from a sequence of unknown structure (SEQUENCE) the following steps are required to finally feed evolutionary information into the PHD neural networks (upper right): (1) a database search for homologues (method BLAST ***[120]***), (2) a refined profile-based dynamic-programming alignment of the most likely homologues (method *MaxHom* ***[121]***), (3) a decision for which proteins will be considered as homologues (length-dependent cutoff for pairwise sequence identity ***[28,92]***), and (4) a final refinement, and extraction of the resulting multiple alignment. Numbers 1–3 indicate the points where users of the *PredictProtein* service ***(18)*** can interfere to improve prediction accuracy without changes made to the final prediction method PHD.

examples proportional to the occurrence in the data set (unbalanced training) results in a prediction accuracy that mirrors this distribution, e.g., strands are predicted inferior to helix or loop ***(20,21,48)***. A simple way around the database bias is a balanced training: at each time step one example is chosen from each class, i.e., one window with the central residue in a helix, one with the

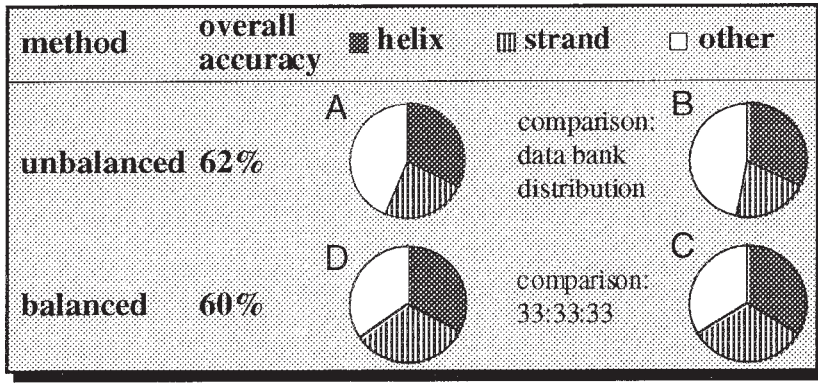


Fig. 4. Prediction balanced between three secondary structure states. The pies were valid for a simple neural network prediction not using evolutionary information (second generation). The entire pies represented 100% of **(A + D)** all correctly predicted residues, **(B)** all residues in a representative subset of PDB, and **(C)** all residues presented during balanced training. The basic message is that the prediction of strand is not inferior to the one for helix for second-generation methods **(A)** because strand formation is more dominated by long-range interactions (as previously argued) but because the database distributions differ between the three states **(B)**. Simply skewing the distribution **(C)** resulted in an equally accurate prediction for all three states **(D)**.

central residue in a strand and one representing the loop class. This training results in a prediction accuracy well balanced between the output states (*see* **Fig. 4**).

Better segment prediction by structure-to-structure networks. The first level sequence-to-structure network uses as input the following information from 13 adjacent residues: (1) the profile of amino acid substitutions for all 13 residues, (2) the conservation weights compiled for each column of the multiple alignment, (3) the number of insertions, and the number of deletions in each column, (4) the position of the current segment of 13 residues with respect to the N- and C-term, (5) the amino acid composition, and (6) the length of the protein. Output consists of three units coding for helix, strand, and nonregular structure. The output coding for the second level network is identical to the one for the first. The dominant input contribution to the second level structure-to-structure network is the output of the first-level sequence-to-structure network. The reason for introducing a second level is the following. Networks are trained by changing the connections between the units such that the error is reduced for each of the examples successively presented to the network during training. The examples are chosen at random. Therefore, the examples taken at time step t and at time step $t + 1$ are usually not adjacent in sequence. This implies

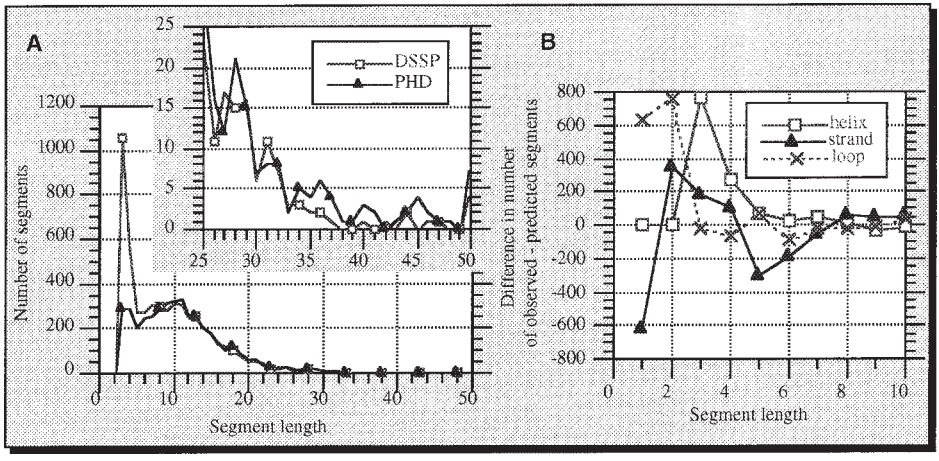


Fig. 5. Distribution of segment length **(A)** The number of helical segments observed (open squares; according to DSSP ***[19]***) and predicted (filled triangles; by PHD ***[18]***) is plotted against their length. Obviously, most short helices are missed by the prediction. The inlet zooms on longer helices, revealing that PHD predicts slightly too long helices. Figures for strands and nonregular structures are not given, as the observed and predicted distributions agree relatively well, for longer segments at least. However, there are important differences for shorter segments: **(B)** plots the differences between the numbers of observed–predicted segments at given lengths (helices: open squares, strands: filled triangles, nonregular structure: dashed line with crosses). In particular, strands of a single residue are overpredicted; short loop regions and three helices ***(10)*** (three residues) are underpredicted.

that the network cannot learn, e.g., that helices contain at least three residues. The second-level structure-to-structure network introduces a correlation between adjacent residues with the effect that predicted secondary-structure segments have length distributions similar to the ones observed ***(27)***. Problems arise, in particular, for short segments (*see* **Fig. 5**).

3.3. State-of-the-Art Secondary Structure Prediction

3.3.1. Estimates of Prediction Accuracy

Difference between 60% and 70% accuracy may matter a lot. Some of the third-generation methods for secondary-structure prediction are clearly superior to previous methods: β-strands are predicted more accurately; predicted segments look like those observed; and the overall accuracy is about 10 percentage points higher. The advantage in practice is illustrated in **Fig. 6**. Not only does the third-generation method (here PHD) gets most segments right,



Fig. 6. Example for secondary structure prediction of first–third generation. The protein sequence (*SEQ*) given was the SH3 structure ***(131)***. The observed secondary structure (*OBS*) was assigned by DSSP ***(19)*** (H = helix; E = strand; blank = nonregular structure; the dashes indicated the continuation of the second strand that was missed by DSSP). The methods are first generation: *C + F* ***(42)***; second generation: *GOR* ***(17)*** (= GORIII), and third generation: PHD ***(18)***. The levels of three-state accuracy were: C + F = 59%; GOR = 65%; and PHD = 72%. Whereas the first- and second-generation methods performed above their average accuracy (**Fig. 1**) for this protein, the PHD prediction was average (*see* **Figs. 1** and **7**). The strength of the PHD prediction was reflected in the one-digit reliability index (*Rel*, 0 = low, 9 = high), correlated with prediction accuracy. All residues predicted at values of Rel > 4 (marked by *) were predicted correctly.

but it also enables one to focus on more reliably predicted residues. The reliability index (*Rel* in **Fig. 6**) is compiled as the difference between the output unit with highest value (winner unit) and the output unit with the next highest value (normalized to a scale from 0 [low] to 9 [high]). All strongly predicted residues (* in **Fig. 6**) are predicted correctly.

Values for expected prediction accuracy are distributions. Statements such as "secondary-structure is about 90% conserved within sequence families" ***(22)*** refer to averages over distributions. The same holds for the expected prediction accuracy (*see* **Fig. 7**). Such distributions explain why some developers have overestimated the performance of their tools using data sets of only tens of proteins (or even fewer). In general, single sequences yield accuracy values about 10 percentage points lower than multiple alignments ***(21,25,48)***. Note that for most proteins some helix and strand residues are confused (refer to **Fig. 7**).

Reliability of prediction correlates with accuracy. For the user interested in a particular protein *U*, the fact that prediction accuracy varies with the protein (*see* **Fig. 7**) implies a rather unfortunate message: the accuracy for *U* could be lower than 40%, or it could be higher than 90% (*see* **Fig. 7**). Is there any way to provide an estimate at which end of the distribution the accuracy for *U* is likely to be? Indeed, the reliability index correlates with accuracy. In other words, residues with a higher reliability index are predicted with higher accuracy

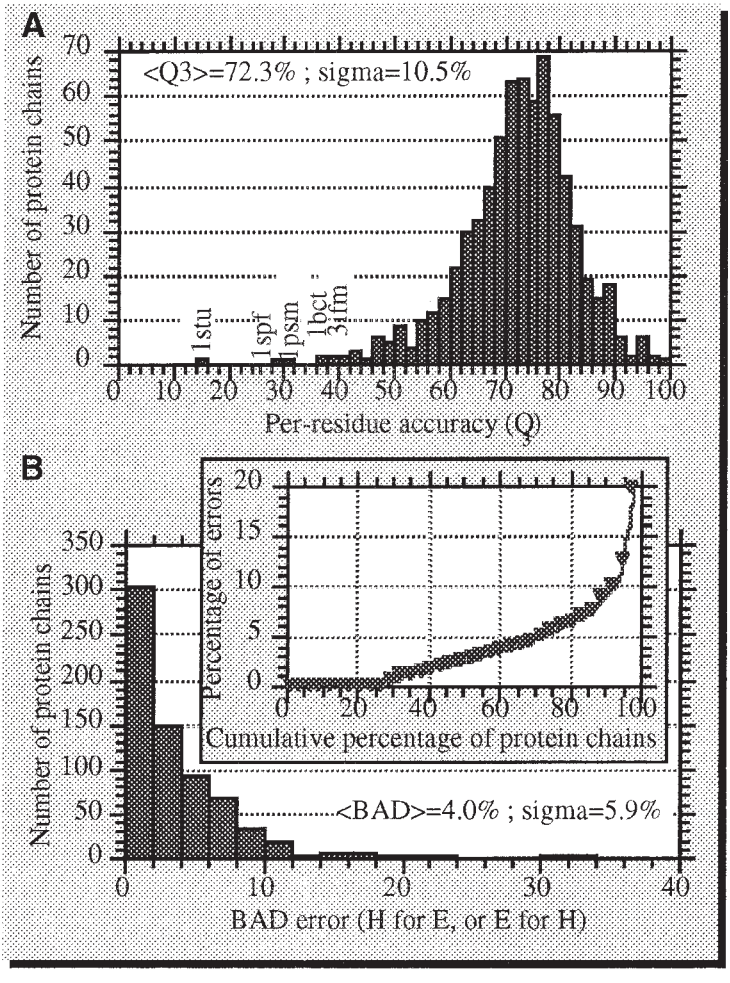


Fig. 7. Expected variation of prediction accuracy with protein chain. **(A)** Three-state per-residue accuracy (*see* **Eq. 1**; PDB identifier given for the proteins predicted worst); **(B)** percentage of BAD predictions, i.e., residues either predicted in helix and observed in strand, or predicted in strand and observed in helix (introduced by **ref. *14***); (B inset) cumulative percentage of proteins with BADly predicted residues (e.g., for 80% of the proteins the percentage of confusing helix and strand residues is <7%; however, for only for 30% of all proteins such a confusion never happened). Given: distributions (over 721 unique protein chains), averages, and one standard deviation.

(18,21,48). Thus, the reliability index offers an excellent tool to focus on some key regions predicted at high levels of expected accuracy. Furthermore, the reliability index averaged over an entire protein correlates with the overall pre-

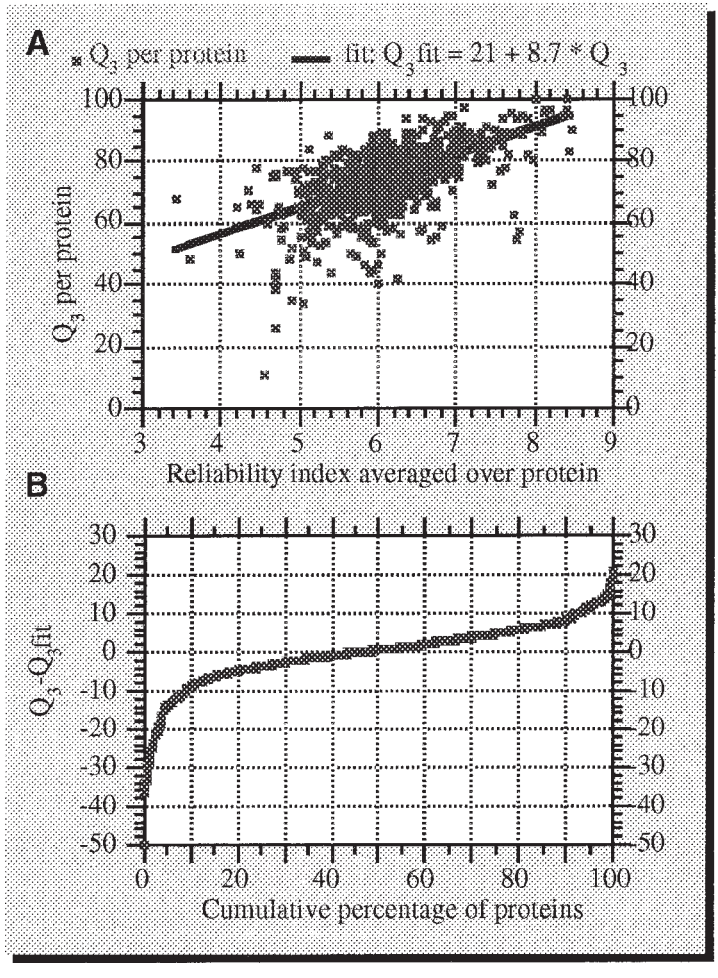


Fig. 8. Correlation between reliability and accuracy. Residues predicted at higher reliability are predicted more accurately ***(18,21,48)***. Here, we plotted the reliability index averaged over a protein with the overall accuracy for that protein **(A)**. Even a simple linear fit **(A)** provided a reasonably accurate estimate of the performance: for more than 80% of all proteins the linear fit yielded estimates in the range of less than ±10% accuracy **(B)**.

diction accuracy for this protein (*see* **Fig. 8**) (Note however, that reliability indices tend to be unusually high for alignments of sequence families without very divergent sequences.)

Do we understand why certain proteins are predicted poorly? For some of the worst predicted proteins, the low level of accuracy could be anticipated from

their unusual features, e.g., for crambin, or the antifreeze glycoprotein type III. However, this procedure turned out to be rather arbitrary. First, some proteins with the same "unusual features" are predicted at high levels of accuracy. Second, occasionally similar proteins are predicted at very different levels of accuracy, e.g., both the phosphotidylinonitol 3-kinase ***(130)*** and the Src-homology domain of cytoskeletal spectrin have homologous structures ***(131)***, but prediction accuracy varies between less than 40% (pik) and more than 70% (spectrin). None of the conclusions from studying poor predictions has yet yielded a way to better predictions. Nevertheless, two observations may be added. First, bad alignments (i.e., noninformative and/or falsely aligned residues) result in bad predictions. Second, the BAD predictions (*see* **Fig. 7B**), i.e., the confusion of helix and strand, are frequently observed in regions that are stabilized by long-range interactions. For example, the peptide around the fourth strand of SH3 (*see* **Fig. 6**) forms a helix in solution (L. Serrano, personal communication). Furthermore, helices and strands that are confused despite a high reliability index often have functional properties, or are correlated to disease states (B. Rost, unpublished data).

3.3.2. Availability of Methods

Internet prediction services for secondary-structure, in general. Programs for the prediction of secondary-structure available as Internet services have mushroomed since the first prediction service PredictProtein went online in 1992 ***(119,132)*** (a list of links is found in **ref. *133***). Unfortunately, not all services are sufficiently tested. In general, prediction accuracy is significantly superior if predictions are based on multiple alignments ***(4,13,16)***.

Completely vs. almost automatic. The PHD prediction method is automatically available via the Internet service PredictProtein ***(18)*** (send the word *help* to `PredictProtein@Columbia.edu`, or use the World Wide Web interface ***[132]***). Users have the choice between the fully automatic procedure taking the query sequence through the entire cycle, or expert intervention into the generation of the alignment. Indeed, without spending much time the result was that predictions could be easily improved ***(134)***.

4. Notes

The following notes result from the experiences one of us (BR) has gathered by offering, and running the PredictProtein ***(132)*** service and during various structure prediction workshops ***(135)***. Some comments apply in particular to the PHD methods ***(18,136)***; however, most hold also for using other secondary-structure prediction methods (we strongly recommend reading the detailed "hints" on the PredictProtein WWW page: ***[132]***).

4.1. What Can You Expect From Secondary Structure Prediction?

How accurate are the predictions? The expected levels of accuracy ($Q_3 = 72 \pm 11\%$) are valid for typical globular, water-soluble proteins when the multiple alignment contains many and diverse sequences. High values for the reliability indices indicate more accurate predictions (Note: for alignments with little variation in the sequences, the reliability indices adopt misleadingly high values.) PHD predictions tend to be relatively accurate for porins ***(18)***; however, for helical membrane proteins, other programs ought to be used ***(5,18,136)***.

How useful are the predictions? The prediction of secondary-structures can be accurate enough to assist chain tracing. Furthermore, PHD predictions are being used as a starting point for modeling 3D structure and predicting function ***(115,116,122,137–143)***.

Is there confusion between strand and helix? PHD (as well as other methods) focuses on predicting hydrogen bonds. Consequently, occasionally strongly predicted (high reliability index) helices are observed as strands and vice versa (*see* **Fig. 7B**).

Is there a strong signal from secondary-structure caps? The ends of helices and strands contain a strong signal. However, on average PHD predicts the core of helices and strands more accurately than do the caps ***(20)***. This seems to also hold for other methods.

Are internal helices poorly predicted? Steven Benner has indicated that internal helices are difficult to predict ***(24,107)***. On average, this is not the case for PHD predictions ***(144)***.

What about protein design and synthesized peptides? The PHD networks are trained on naturally evolved proteins. However, the predictions have been useful in some cases to investigate the influence of single mutations (e.g., for Chameleon ***[145,146]***, or for Janus ***[147]***; B. Rost, unpublished). For short polypeptides, users should bear in mind that the network input consists of 17 adjacent residues. Thus, shorter sequences may be dominated by the ends (which are treated as solvents by the current version of PHD).

4.2. How Can You Avoid Pitfalls?

70% correct implies 30% incorrect. The most accurate methods for predicting secondary-structure reach sustained levels of about 70% accuracy. When interpreting predictions for a particular protein, it is often instructive to mark the 30% of the residues you suspect to be falsely predicted.

Spread of prediction accuracy. An expected accuracy of 70% does *not* imply that for your protein *U* 70% of all residues are correctly predicted. Instead, values published for prediction accuracy are averaged over hundreds of unique proteins. An expected accuracy of $70 \pm 10\%$ (one standard deviation) implies

that, on average, for two-thirds of all proteins between 60 and 80% of the residues will be predicted correctly (*see* **Fig. 7**). Thus, prediction accuracy can be higher than 80% or lower than 60% for your protein. Few methods supply well-tested indices for the reliability of predictions (*see* **Fig. 8**; ***[18,134]***). Such indices can help to reduce or increase your trust in a particular prediction.

Special classes of proteins. Prediction methods are usually derived from knowledge contained in proteins from subsets of current databases. Consequently, they should not be applied to classes of proteins not included in these subsets, e.g., methods for predicting helices in globular proteins are likely to fail when applied to predict transmembrane helices. In general, results should be taken with caution for proteins with unusual features, such as proline-rich regions, unusually many cysteine bonds, or for domain interfaces.

Better alignments yield better predictions. Multiple-alignment-based predictions are substantially more accurate than single-sequence-based predictions. How many sequences do you need in your alignment for an improvement? How sensitive are prediction methods to errors in the alignment? The more divergent sequences contained in the alignment, the better (two distantly related sequences often improve secondary-structure predictions by several percentage points). Regions with few aligned sequences yield less reliable predictions. The sensitivity to alignment errors depends on the methods, e.g., secondary-structure prediction is less sensitive to alignment errors than accessibility prediction.

Better + worse = even better? Today, several automatic services accomplish secondary-structure predictions. Some users fall into the what-is-common-is-correct trap, i.e., they average over all prediction methods and consider identical regions as more reliable. Such a majority vote may be beneficial. However, the result will frequently be the worst-of-all prediction. Often, it is preferable to use reliability indices provided by some methods. Such indices answer the question: how reliably is the tryptophan at position 307 predicted in a surface loop? (Note: the correlation between such indices and prediction accuracy is sufficiently tested for only a few methods.)

1D structure may or may not be sufficient to infer 3D structure. Say you the following as a prediction for a regular secondary-structure: helix-strand-strand-helix-strand-strand (H-E-E-H-E-E). Assume that you find a protein of known structure with the same motif (H-E-E-H-E-E). Can you conclude that the two proteins have the same fold? Yes and no; your guess may be correct, but there are various ways to realize the given motif by completely different structures. For example, at least 16 structurally unrelated proteins contain the secondary-structure motif H-E-E-H-E-E.

Addendum

At the third meeting for the Critical Assessment of Structure Prediction (CASP) in December 1998, David Jones presented a method that extended the basic idea of 3rd generation prediction methods, i.e., using evolutionary information, by replacing previously used sequence alignment procedures with an iterated PSI-BLAST profile ***(149)***. The resulting method PSI-PRED appears to be more than 2–3 percentage points more accurate than any other method published so far ***(150)***. About one percentage point of this improvement can be achieved by simply replacing the alignment profiles (Rost, unpublished).

However, the major step appears to be attributed to the fact that the databases have grown, and developing prediction methods can now be based on data sets more than 10 times larger than those used to develop the first 3rd generation tools (Rost, unpublished). The work of David Jones has reactivated the field, at least one other novel method (JNET: Cuff & Barton, unpublished) appears clearly moved accurate than the original PHD1 referred to in our review.

References

1. Bairoch, A. and Apweiler, R. (1997) The SWISS-PROT protein sequence data bank and its new supplement TrEMBL. *Nucleic Acids Res.* **25,** 31–36.
2. Gaasterland, T. (1997) Genome sequencing projects. WWW document (http://genomes.rockefeller.edu):Rockefeller University.
3. Bernstein, F. C., et al. (1977) The Protein Data Bank: a computer based archival file for macromolecular structures. *J. Mol. Biol.* **112,** 535–542.
4. Rost, B. and Sander, C. (1996) Bridging the protein sequence–structure gap by structure predictions. *Annu. Rev. Biophys. Biomol. Struct.* **25,** 113–136.
5. Rost, B. and O'Donoghue, S. I. (1997) Sisyphus and prediction of protein structure. *CABIOS* **13,** 345–356.
6. CASP1. (1995) Special issue of *Proteins* **23,** 295–462.
7. CASP2. (1997) Special issue of *Proteins* **Suppl 1,** 1–230.
8. Rost, B. and Sander, C. (1994) Structure prediction of proteins — where are we now? *Curr. Opin. Biotech.* **5,** 372–380.
9. Rost, B. (1998) Protein structure prediction in 1D, 2D, and 3D, in *Encyclopedia of Computational Chemistry* (von Ragué Schleyer, P., et al., eds.), Wiley, Chichester, UK, pp. 2242–2255.
10. Fasman, G. D. (1989) *Prediction of Protein Structure and the Principles of Protein Conformation*. Plenum, New York, London.
11. Sternberg, M. J. E. (1992) Secondary structure prediction. *Curr. Opin. Struct. Biol.* **2,** 237–241.
12. Presnell, S. R. and Cohen, F. E. (1993) Artificial neural networks for pattern recognition in biochemical sequences. *Annu. Rev. Biophys. Biomol. Struct.* **22,** 283–298.

13. Barton, G. J. (1995) Protein secondary structure prediction. *Curr. Opin. Struct. Biol.* **5,** 372–376.
14. Defay, T. and Cohen, F. E. (1995) Evaluation of current techniques for *ab initio* protein structure prediction. *Proteins* **23,** 431–445.
15. Russell, R. B. and Sternberg, M. J. E. (1995) How good are we? *Curr. Biol.* **5,** 488–490.
16. Di Francesco, V., Garnier, J., and Munson, P. J. (1996) Improving protein secondary structure prediction with aligned homologous sequences. *Protein Sci.* **5,** 106–113.
17. Garnier, J., Gibrat, J.-F., and Robson, B. (1996) GOR method for predicting protein secondary structure from amino acid sequence. *Methods Enzymol.* **266,** 540–553.
18. Rost, B. (1996) PHD: predicting one-dimensional protein structure by profile based neural networks. *Methods Enzymol.* **266,** 525–539.
19. Kabsch, W. and Sander, C. (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen bonded and geometrical features. *Biopolymers* **22,** 2577–2637.
20. Rost, B. and Sander, C. (1994) 1D secondary structure prediction through evolutionary profiles, in *Protein Structure by Distance Analysis* (Bohr, H., and Brunak, S., eds.), IOS Press, Amsterdam, Oxford, Washington, pp. 257–276.
21. Rost, B. and Sander, C. (1993) Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* **232,** 584–599.
22. Rost, B., Sander, C., and Schneider, R. (1994) Redefining the goals of protein secondary structure prediction. *J. Mol. Biol.* **235,** 13–26.
23. Thornton, J. M., et al. (1992) Prediction of progress at last. *Nature* **354,** 105–106.
24. Benner, S. A. and Gerloff, D. L. (1993) Predicting the conformation of proteins: man versus machine. *FEBS Lett.* **325,** 29–33.
25. Rost, B., Sander, C., and Schneider, R. (1993) Progress in protein structure prediction? *TIBS* **18,** 120–123.
26. Russell, R. B. and Barton, G. J. (1993) The limits of protein secondary structure prediction accuracy from multiple sequence alignment. *J. Mol. Biol.* **234,** 951–957.
27. Rost, B. and Sander, C. (1993) Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proc. Natl. Acad. Sci. USA* **90,** 7558–7562.
28. Sander, C. and Schneider, R. (1991) Database of homology-derived structures and the structural meaning of sequence alignment. *Proteins* **9,** 56–68.
29. Kendrew, J. C., et al. (1960) Structure of myoglobin: a three-dimensional Fourier synthesis at 2Å resolution. *Nature* **185,** 422–427.
30. Perutz, M. F., et al. (1960) Structure of haemoglobin: a three-dimensional Fourier synthesis at 5. 5Å resolution, obtained by X-ray analysis. *Nature* **185,** 416–422.
31. Pauling, L. and Corey, R. B. (1951) Configurations of polypeptide chains with favored orientations around single bonds: two new pleated sheets. *Proc. Natl. Acad. Sci. USA* **37,** 729–740.

32. Pauling, L., Corey, R. B., and Branson, H. R. (1951) The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. *Proc. Natl. Acad. Sci. USA* **37,** 205–234.
33. Szent-Györgyi, A. G. and Cohen, C. (1957) Role of proline in polypeptide chain configuration of proteins. *Science* **126,** 697.
34. Blout, E. R., et al. (1960) Dependence of the conformation of synthetic polypeptides on amino acid composition. *J. Am. Chem. Soc.* **82,** 3787–3789.
35. Blout, E. R. (1962) The dependence of the conformation of polypetides and proteins upon amino acid composition, in *Polyamino Acids, Polypeptides, and Proteins* (Stahman, M., ed.), University of Wisconsin Press, Madison, WI, pp. 275–279.
36. Scheraga, H. A. (1960) Structural studies of ribonuclease III. A model for the secondary and tertiary structure. *J. Am. Chem. Soc.* **82,** 3847–3852.
37. Davies, D. R. (1964) A correlation between amino acid composition and protein structure. *J. Mol. Biol.* **9,** 605–609.
38. Schiffer, M. and Edmundson, A. B. (1967) Use of helical wheels to represent the structures of proteins and to identify segments with helical potential. *Biophys. J.* **7,** 121.
39. Pain, R. H. and Robson, B. (1970) Analysis of the code relating sequence to secondary structure in proteins. *Nature* **227,** 62–63.
40. Finkelstein, A. V. and Ptitsyn, O. B. (1971) Statistical analysis of the correlation among amino acid residues in helical, β-structural and non-regular regions of globular proteins. *J. Mol. Biol.* **62,** 613–624.
41. Robson, B. and Pain, R. H. (1971) Analysis of the code relating sequence to conformation in proteins: possible implications for the mechanism of formation of helical regions. *J. Mol. Biol.* **58,** 237–259.
42. Chou, P. Y. and Fasman, U. D. (1974) Prediction of protein conformation. *Biochemistry* **13,** 211–215.
43. Lim, V. I. (1974) Structural principles of the globular organization of protein chains. A stereochemical theory of globular protein secondary structure. *J. Mol. Biol.* **88,** 857–872.
44. Rose, G. D. (1978) Prediction of chain turns in globular proteins on a hydrophobic basis. *Nature* **272,** 586–590.
45. Kabsch, W. and Sander, C. (1983) How good are predictions of protein secondary structure? *FEBS Lett.* **155,** 179–182.
46. Rost, B. (2000) Neural networks for protein structure prediction: hype or hit? Preprint, (http://cubic.bioc.columbia.edu/papers/Pre 1999_tics/) Columbia, University, New York.
47. Rost, B. and Sander, C. (1993) Secondary structure prediction of all-helical proteins in two states. *Protein Eng.* **6,** 831–836.
48. Rost, B. and Sander, C. (1994) Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins* **19,** 55–72.
49. Kabat, E. A. and Wu, T. T. (1973) The influence of nearest-neighbor amino acids on the conformation of the middle amino acid in proteins: comparison of predicted and experimental determination of β-sheets in concanavalin A. *Proc. Natl. Acad. Sci. USA* **70,** 1473–1477.

50. Maxfield, F. R. and Scheraga, H. A. (1976) Status of empirical methods for the prediction of protein backbone topography. *Biochemistry* **15,** 5138–5153.
51. Robson, B. (1976) Conformational properties of amino acid residues in globular proteins. *J. Mol. Biol.* **107,** 327–356.
52. Nagano, K. (1977) Triplet information in helix prediction applied to the analysis of super-secondary structures. *J. Mol. Biol.* **109,** 251–274.
53. Garnier, J., Osguthorpe, D. J., and Robson, B. (1978) Analysis of the accuracy and Implications of simple methods for predicting the secondary structure of globular proteins. *J. Mol. Biol.* **120,** 97–120.
54. Gibrat, J.-F., Garnier, J., and Robson, B. (1987) Further developments of protein secondary structure prediction using information theory. New parameters and consideration of residue pairs. *J. Mol. Biol.* **198,** 425–443.
55. Biou, V., et al. (1988) Secondary structure prediction: combination of three different methods. *Protein Eng.* **2,** 185–91.
56. Gascuel, O. and Golmard, J. L. (1988) A simple method for predicting the secondary structure of globular proteins: implications and accuracy. *CABIOS* **4,** 357–365.
57. Lupas, A., Van Dyke, M., and Stock, J. (1991) Predicting coiled coils from protein sequences. *Science* **252,** 1162–1164.
58. Viswanadhan, V. N., Denckla, B., and Weinstein, J. N. (1991) New joint prediction algorithm (Q7-JASEP) improves the prediction of protein secondary structure. *Biochemistry* **30,** 11,164–11,172.
59. Juretic, D., et al. (1993) Conformational preference functions for predicting helices in membrane proteins. *Biopolymers* **33,** 255–273.
60. Mamitsuka, H. and Yamanishi, K. (1993) Protein α-helix region prediction based on stochastic-rule learning, in *26th Annual Hawaii International Conference on System Sciences* (eds.), IEEE Computer Society, Maui, HI, pp. 659–668.
61. Donnelly, D., Overington, J. P., and Blundell, T. L. (1994) The prediction and orientation of α-helices from sequence alignments: the combined use of environment-dependent substitution tables, Fourier transform methods and helix capping rules. *Protein Eng.* **7,** 645–653.
62. Ptitsyn, O. B. and Finkelstein, A. V. (1983) Theory of protein secondary structure and algorithm of its prediction. *Biopolymers* **22,** 15–25.
63. Taylor, W. R. and Thornton, J. M. (1983) Prediction of super-secondary structure in proteins. *Nature* **301,** 540–542.
64. Cohen, F. E. and Kuntz, I. D. (1989) Tertiary structure prediction, in *Prediction of Protein Structure and the Principles of Protein Conformation* (Fasman, G. D., eds.), Plenum, New York, London, pp. 647–706.
65. Rooman, M. J., Kocher, J. P., and Wodak, S. J. (1991) Prediction of protein backbone conformation based on seven structure assignments: influence of local interactions. *J. Mol. Biol.* **221,** 961–979.
66. Qian, N. and Sejnowski, T. J. (1988) Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.* **202,** 865–884.
67. Bohr, H., et al. (1988) Protein secondary structure and homology by neural networks. *FEBS Lett.* **241,** 223–228.

68. Holley, H. L. and Karplus, M. (1989) Protein secondary structure prediction with a neural network. *Proc. Natl. Acad. Sci. USA* **86,** 152–156.
69. Kneller, D. G., Cohen, F. E., and Langridge, R. (1990) Improvements in protein secondary structure prediction by an enhanced neural network. *J. Mol. Biol.* **214,** 171–182.
70. Stolorz, P., Lapedes, A., and Xia, Y. (1992) Predicting protein secondary structure using neural net and statistical methods. *J. Mol. Biol.* **225,** 363–377.
71. Zhang, X., Mesirov, J. P., and Waltz, D. L. (1992) Hybrid system for protein secondary structure prediction. *J. Mol. Biol.* **225,** 1049–1063.
72. Maclin, R. and Shavlik, J. W. (1993) Using knowledge-based neural networks to improve algorithms: refining the Chou-Fasman algorithm for protein folding. *Machine Learning* **11,** 195–215.
73. Chandonia, J.-M. and Karplus, M. (1995) Neural networks for secondary structure and structural class predictions. *Protein Sci.* **4,** 275–285.
74. Mitchell, E. M., et al. (1992) Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.* **212,** 151–166.
75. Geourjon, C. and Deléage, G. (1995) SOPMA: significant improvements in protein secondary structure prediction by consensus prediction from multiple alignments. *CABIOS* **11,** 681–684.
76. Kanehisa, M. (1988) A multivariate analysis method for discriminating protein secondary structural segments. *Protein Eng.* **2,** 87–92.
77. Munson, P. J. and Singh, R. K. (1997) Multi-body interactions within the graph of protein structure, in *Fifth International Conference on Intelligent Systems for Molecular Biology* (Gaasterland, T., et al., eds.), AAAI Press, Halkidiki, Greece, pp. 198–201.
78. King, R. D., et al. (1992) Drug design by machine learning: the use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proc. Natl. Acad. Sci. USA* **89,** 11,322–11,326.
79. Muggleton, S., King, R. D., and Sternberg, M. J. E. (1992) Protein secondary structure prediction using logic-based machine learning. *Protein Eng.* **5,** 647–657.
80. Frishman, D. and Argos, P. (1995) Knowledge-based protein secondary structure assignment. *Proteins* **23,** 566–579.
81. Zhu, Z.-Y. and Blundell, T. L. (1996) The use of amino acid patterns of classified helices and strands in secondary structure prediction. *J. Mol. Biol.* **260,** 261–276.
82. Asogawa, M. (1997) Beta-sheet prediction using inter-strand residue pairs and refinement with Hopfield neural network, in *Fifth International Conference on Intelligent Systems for Molecular Biology* (Gaasterland, T., et al., eds.), AAAI Press, Halkidiki, Greece, pp. 48–51.
83. Yi, T.-M. and Lander, E. S. (1993) Protein secondary structure prediction using nearest-neigbor methods. *J. Mol. Biol.* **232,** 1117–1129.
84. Solovyev, V. V. and Salamov, A. A. (1994) Predicting α-helix and β-strand segments of globular proteins. *CABIOS* **10,** 661–669.

85. Salamov, A. A. and Solovyev, V. V. (1995) Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignment. *J. Mol. Biol.* **247,** 11–15.
86. Kabsch, W. and Sander, C. (1983) Segment83. unpublished.
87. Schneider, R. (1989) Sekundärstrukturvorhersage von Proteinen unter Berücksichtigung von Tertiärstrukturaspekten. Diploma thesis: Department of Biology, University of Heidelberg, Heidelberg, Germany.
88. Devereux, J., Haeberli, P., and Smithies, O. (1984) A comprehensive set of sequence analysis programs for the VAX. *Nucleic Acids Res.* **12,** 387–395.
89. Rost, B. and Schneider, R. (1998) Pedestrian guide to analysing sequence databases, in *Core Techniques in Biochemistry* (Ashman, K., ed.) http://cubic.bioc.columbia.edu/papers/1999_pedestrian/, Springer, Heidelberg, pp. in press.
90. Dao-pin, S., et al. (1991) Contributions of surface salt bridges to the stability of bacteriophage T4 lysozyme determined by directed mutagenesis. *Biochemistry* **30,** 7142–7153.
91. Doolittle, R. F. (1986) *Of URFs and ORFs: A Primer on How to Analyze Derived Amino Acid Sequences*. University Science Books, Mill Valley CA.
92. Chothia, C. and Lesk, A. M. (1986) The relation between the divergence of sequence and structure in proteins. *EMBO J.* **5,** 823–826.
93. Lesk, A. M. (1991) *Protein Architecture — A Practical Approach*. Oxford University Press, Oxford, New York, Tokyo.
94. Rost, B. (1998) Twilight zone of protein sequence alignments. *Prof. Enginering* **12,** S85–S94.
95. Rost, B. (1997) Protein structures sustain evolutionary drift. *Fold. Des.* **2,** S19–S24.
96. Rost, B., O'Donoghue, S., and Sander, C. (1998) Midnight zone of protein structure evolution. Preprint (http://cubic.bioc.columbia.edu/papers/Pre 1998_midnight/) Columbia University, New York.
97. Lichtarge, O., Bourne, H. R., and Cohen, F. E. (1996) An evolutionary trace method defines binding surfaces common to protein families. *J. Mol. Biol.* **257,** 342–358.
98. Pazos, F., et al. (1997) Comparative analysis of different methods for the detection of specificity regions in protein families, in *BCEC97: Bio-Computing and Emergent Computation* (Olsson, B., Lundh, D., and Narayanan, A., eds.), World Scientific, Skövde, Sweden, pp. 132–145.
99. Dickerson, R. E., Timkovich, R., and Almassy, R. J. (1976) The cytochrome fold and the evolution of bacterial energy metabolism. *J. Mol. Biol.* **100,** 473–491.
100. Dickerson, R. E. (1971) The structure of cytochrome c and the rates of molecular evolution. *J. Mol. Evol.* **1,** 26–45.
101. Frampton, J., et al. (1989) DNA-binding domain ancestry. *Nature* **342,** 134.
102. Benner, S. A. (1989) Patterns of divergence in homologous proteins as indicators of tertiary and quaternary structure. *Adv. Enzyme Regul.* **28,** 219–236.
103. Bazan, J. F. (1990) Structural design and molecular evolution of a cytokine receptor superfamily. *Proc. Natl. Acad. Sci. USA* **87,** 6934–6938.
104. Benner, S. A. and Gerloff, D. (1990) Patterns of divergence in homologous proteins as indicators of secondary and tertiary structure of the catalytic domain of protein kinases. *Adv. Enzyme Regul.* **31,** 121–181.

105. Niermann, T. and Kirschner, K. (1990) Improving the prediction of secondary structure of "TIM-barrel" enzymes. *Protein Eng.* **4,** 137–147.
106. Barton, G. J., et al. (1991) Amino acid sequence analysis of the annexin supergene family of proteins. *Eur. J. Biochem.* **198,** 749–760.
107. Benner, S. A. (1992) Predicting de novo the folded structure of proteins. *Curr. Opin. Struct. Biol.* **2,** 402–412.
108. Gibson, T. J. (1992) Assignment of α-helices in multiply aligned protein sequences — applications to DNA binding motifs, in *Patterns in Protein Sequence and Structure* (Taylor, W. R., ed.), Berlin-Heidelberg, Springer-Verlag, pp. 99–110.
109. Musacchio, A., et al. (1992) SH3 — an abundant protein domain in search of a function. *FEBS Lett.* **307,** 55–61.
110. Barton, G. J. and Russell, R. B. (1993) Protein structure prediction. *Nature* **361,** 505–506.
111. Boscott, P. E., Barton, G. J., and Richards, W. G. (1993) Secondary structure prediction for homology modelling. *Protein Eng.* **6,** 261–266.
112. Gerloff, D. L., et al. (1993) The nitrogenase MoFe protein. *FEBS Lett.* **318,** 118–124.
113. Gibson, T. J., Thompson, J. D., and Abagyan, R. A. (1993) Proposed structure for the DNA-binding domain of the Helix-Loop-Helix family of eukaryotic gene regulatory proteins. *Protein Eng.* **6,** 41–50.
114. Livingstone, C. D. and Barton, G. J. (1994) Secondary structure prediction from multiple sequence data: blood clotting factor XIII and versinia protein-tyrosine phosphatase. *Int. J. Peptide Protein Res.* **44,** 239–244.
115. Hansen, J. E., et al. (1996) Prediction of the secondary structure of HIV-1 gp120. *Proteins* **25,** 1–11.
116. Valencia, A., et al. (1995) Prediction of the structure of GroES and its interaction with GroEL. *Proteins* **22,** 199–209.
117. Maxfield, F. R. and Scheraga, H. A. (1979) Improvements in the prediction of protein topography by reduction of statistical errors. *Biochemistry* **18,** 697–704.
118. Zvelebil, M. J., et al. (1987) Prediction of protein secondary structure and active sites using alignment of homologous sequences. *J. Mol. Biol.* **195,** 957–961.
119. Rost, B., Sander, C., and Schneider, R. (1994) PHD — an automatic server for protein secondary structure prediction. *CABIOS* **10,** 53–60.
120. Altschul, S. F. and Gish, W. (1996) Local alignment statistics. *Methods Enzymol.* **266,** 460–480.
121. Schneider, R. (1994) Sequenz und Sequenz-Struktur Vergleiche und deren Anwendung für die Struktur- und Funktionsvorhersage von Proteinen. PhD thesis: University of Heidelberg, Heidelberg, Germany.
122. Hubbard, T. J. P. and Park, J. (1995) Fold recognition and *ab initio* structure predictions using Hidden Markov models and β-strand pair potentials. *Proteins* **23,** 398–402.
123. Mehta, P. K., Heringa, J., and Argos, P. (1995) A simple and fast approach to prediction of protein secondary structure from multiply aligned sequences with accuracy above 70%. *Protein Sci.* **4,** 2517–2525.

124. Riis, S. K. and Krogh, A. (1996) Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments. *J. Comp. Biol.* **3,** 163–183.
125. Gerloff, D. L. and Cohen, F. E. (1996) Secondary structure prediction and unrefined tertiary structure prediction for cyclin A, B, and D. *Proteins* **24,** 18–34.
126. Frishman, D. and Argos, P. (1997) 75% accuracy in protein secondary structure prediction. *Proteins* **27,** 329–335.
127. Salamov, A. A. and Solovyev, V. V. (1997) Protein secondary structure prediction using local alignments. *J. Mol. Biol.* **268,** 31–36.
128. Levin, J. M., et al. (1993) Quantification of secondary structure prediction improvement using multiple alignment. *Protein Eng.* **6,** 849–854.
129. King, R. D. and Sternberg, M. J. (1996) Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Protein Sci.* **5,** 2298–2310.
130. Koyama, S., et al. (1993) Structure of the PI3K SH3 domain and analysis of the SH3 family. *Cell* **72,** 945–952.
131. Musacchio, A., et al. (1992) Crystal structure of a Src-homology 3 (SH3) domain. *Nature* **359,** 851–855.
132. Rost, B. (1997) PredictProtein — internet prediction service. WWW document (http://cubic.bioc.columbia.edu/predictprotein):Columbia University, New York.
133. Rost, B. and Schneider, R. (1996) WWW services for sequence analysis. WWW document (http://cubic.bioc.columbia.edu/doc/links_index.html):Columbia University, New York.
134. Rost, B. (1998) Better 1D predictions by experts with machines. *Proteins*, in press.
135. Rost, B. and Valencia, A. (1996) Pitfalls of protein sequence analysis. *Curr. Opin. Biotechnol.* **7,** 457–461.
136. Rost, B., Casadio, R., and Fariselli, P. (1996) Topology prediction for helical transmembrane proteins at 86% accuracy. *Protein Sci.* **5,** 1704–1718.
137. Meitinger, T., et al. (1993) Molecular modelling of the Norrie disease protein predicts a cysteine knot growth factor tertiary structure. *Nature* Gen., **5,** 376–380.
138. Rawlings, D. J., et al. (1993) Mutation of unique region of Bruton's tyrosine kinase in immunodeficient XID M mice. *Science* **261,** 358–361.
139. Lupas, A., et al. (1994) Predicted secondary structure of the 20S proteasome and model structure of the putative peptide channel. *FEBS Lett.* **354,** 45–49.
140. Viguera, E., et al. (1994) Mammalian L-amino acid decarboxylases producing 1,4-diamines: analogies among differences. *TIBS* **19,** 318–319.
141. Fischer, D. and Eisenberg, D. (1996) Fold recognition using sequence-derived properties. *Protein Sci.* **5,** 947–955.
142. Rost, B., Schneider, R., and Sander, C. (1997) Protein fold recognition by prediction-based threading. *J. Mol. Biol.* **270,** 471–480.
143. Springer, T. A. (1997) Folding of the N-terminal, ligand-binding region of integrin α-subunits into a β-propeller domain. *Proc. Natl. Acad. Sci. USA* **94,** 65–72.
144. Rost, B. (1996) Accuracy of predicting buried helices by PHDsec. WWW document (http://cubic.bioc.columbia.edu/results/1996/PredBuriedHelices.html). Columbia University, New York.

145. Minor, D. L. J. and Kim, P. S. (1996) Context-dependent secondary structure formation of a designed protein sequence. *Nature* **380,** 730–734.
146. Rost, B. (1996) 1D structure prediction for Chameleon (IgG binding domain of protein G) WWW document (http://cubic.bioc.columbia.edu/results/1996/PredCameleon.html):Columbia University, New York.
147. Dalal, S., Balasubramanian, S., and Regan, L. (1997) Protein alchemy: changing β-sheet into α-helix. *Nat. Struct. Biol.* **4,** 548–552.
148. Chou, P. Y. and Fasman, G. D. (1978) Prediction of the secondary structure of proteins from their amino acid sequence. *Adv. Enzymol.* **47,** 45–148.
149. PSI-BLAST: Altschul, S., et al. (1997) Gapped Blast and PSI-Blast: a new generation of protein database search programs. *Nucl. Acids Res.* **25,** 3389–3402.
150. PSI-PRED: Jones, D. T. (1999) Protein secondary structure prediction based on position specific scoring matrices. *J. Mol. Biol.* **292,** 195–202.

6

Comparative Protein Structure Modeling

Introduction and Practical Examples with Modeller

Roberto Sánchez and Andrej Šali

1. Introduction

1.1. What is Comparative Protein Structure Modeling?

A useful three-dimensional (3D) model for a protein of unknown structure (the target) can frequently be built based on one or more related proteins of known structure (the templates). This is the aim of comparative or homology protein structure modeling. The necessary conditions are that the similarity between the target sequence and the template structures is detectable and that the correct alignment between them can be constructed. For reviews of comparative modeling, *see* **refs. *1–5***. This approach to structure prediction is possible because a small change in the protein sequence usually results in a small change in its 3D structure ***(6,7)***.

1.2. Why is Comparative Modeling Useful?

The biochemical function of a protein is defined by its interactions with other molecules and the biological function is a consequence of these interactions. Although protein function is best determined experimentally ***(8)***, it can sometimes be predicted by matching the sequence of a protein with proteins of known function ***(8–10)***. One way to improve sequence-based predictions of function is to rely on the known native 3D structure of proteins. The 3D structure of a protein generally provides more information about its function than sequence because interactions of a protein with other molecules are determined by amino acid residues that are close in space but are frequently distant in sequence. For example, several mouse mast cell proteases have a conserved surface region of positively charged residues that binds proteoglycans ***(11)***.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

Table 1
Common Uses of Comparative Protein Structure Models

Designing (site-directed) mutants to test hypotheses about function
Identifying active and binding sites
Searching for ligands of a given binding site
Designing and improving ligands of a given binding site
Modeling substrate specificity
Predicting antigenic epitopes
Protein–protein docking simulations
Inferring function from calculated electrostatic potential around the protein
Molecular replacement in X-ray structure refinement
Testing a given sequence–structure alignment
Rationalizing known experimental observations
Planning new experiments

This region is not easily recognizable in sequence because the constituting residues occur at variable and sequentially nonlocal positions that form a binding site only when the protease is fully folded.

Comparative modeling remains the only method that can reliably predict the 3D structure of a protein with an accuracy comparable to that of low-resolution experimental structures ***(1)***. Even such low resolution models are useful to address biological questions, because function can sometimes be predicted from only coarse structural features of a model. Typical uses of comparative models are listed in **Table 1**. For a review of comparative modeling applications *see* **refs.** ***2*** and ***3***.

Three-dimensional structure of proteins from the same family is more conserved than their sequences ***(12)***. Therefore, if similarity between two proteins is detectable at the sequence level, structural similarity can usually be assumed. Moreover, proteins that share low or even nondetectable sequence similarity many times also have similar structures. It has been estimated that approximately one third of all sequences are related to at least one protein of known structure ***(13)***. Because there are approx 450,000 known protein sequences ***(14)***, comparative modeling could, in principle, be applied to approx 150,000 proteins. This is an order of magnitude more proteins than the number of experimentally determined protein structures (approx 10,000) ***(15)***. Furthermore, the usefulness of comparative modeling is steadily increasing because the number of different structural folds that proteins adopt is limited ***(16)***, and because the number of experimentally determined new structures is increasing exponentially ***(17)***. It is predicted that, in less than 10 yr, at least one example

of most structural folds will be known, making comparative modeling applicable to most globular domains in most protein sequences ***(1,17)***.

2. Steps in Comparative Modeling

Comparative modeling usually consists of the following five steps: search for templates, selection of one or more templates, target–template alignment, model building, and model evaluation (*see* **Fig. 1**). If the model is not satisfactory, some or all of the steps can be repeated. Each of these steps is described as follows.

2.1. Search for Templates

Comparative modeling usually starts by searching the database of known protein structures (Protein Data bank, PDB) ***(15)*** using the target sequence as the query. This is generally done by comparing the target sequence with the sequence of each of the structures in the database. A variety of sequence-sequence comparison methods can be used ***(18–20)***. Sometimes, the availability of many sequences related to the target makes it possible to do more sensitive searching with profile methods and Hidden Markov Models (HMM) ***(21–24)***. It is also possible to search for templates by evaluating directly the compatibility between the target sequence and each of the structures in the database. This is achieved by fold-recognition methods also known as "threading" ***(25–29)***. Threading uses sequence–structure fitness functions, such as low-resolution, knowledge-based force-fields, to evaluate potential target-template matches. In doing so, threading methods generally do not rely on sequence similarity. This sometimes allows recognition of structural similarity between proteins with no detectable sequence similarity ***(30)***.

A good starting point for template searches are the many database search servers on the World Wide Web (WWW) (*see* **Table 2**). The most useful ones are those that search directly against the PDB. If nothing is found with sequence similarity searches, threading programs and fold-recognition WWW servers can be used (**Table 2**). In general, it is useful to try many different methods to find as many templates as possible. This is especially important when the target sequence is only remotely related to known structures.

2.2. Template Selection

Once a list of potential templates has been obtained using one or more template searching methods, it is necessary to select the templates that are appropriate for the particular modeling problem. Usually, the higher the overall sequence similarity (i.e., higher percentage of identical residues, and lower number and shorter length of gaps in the alignment) between the target and the template sequences, the better the template is likely to be. Other factors should also be taken into account when selecting a template:

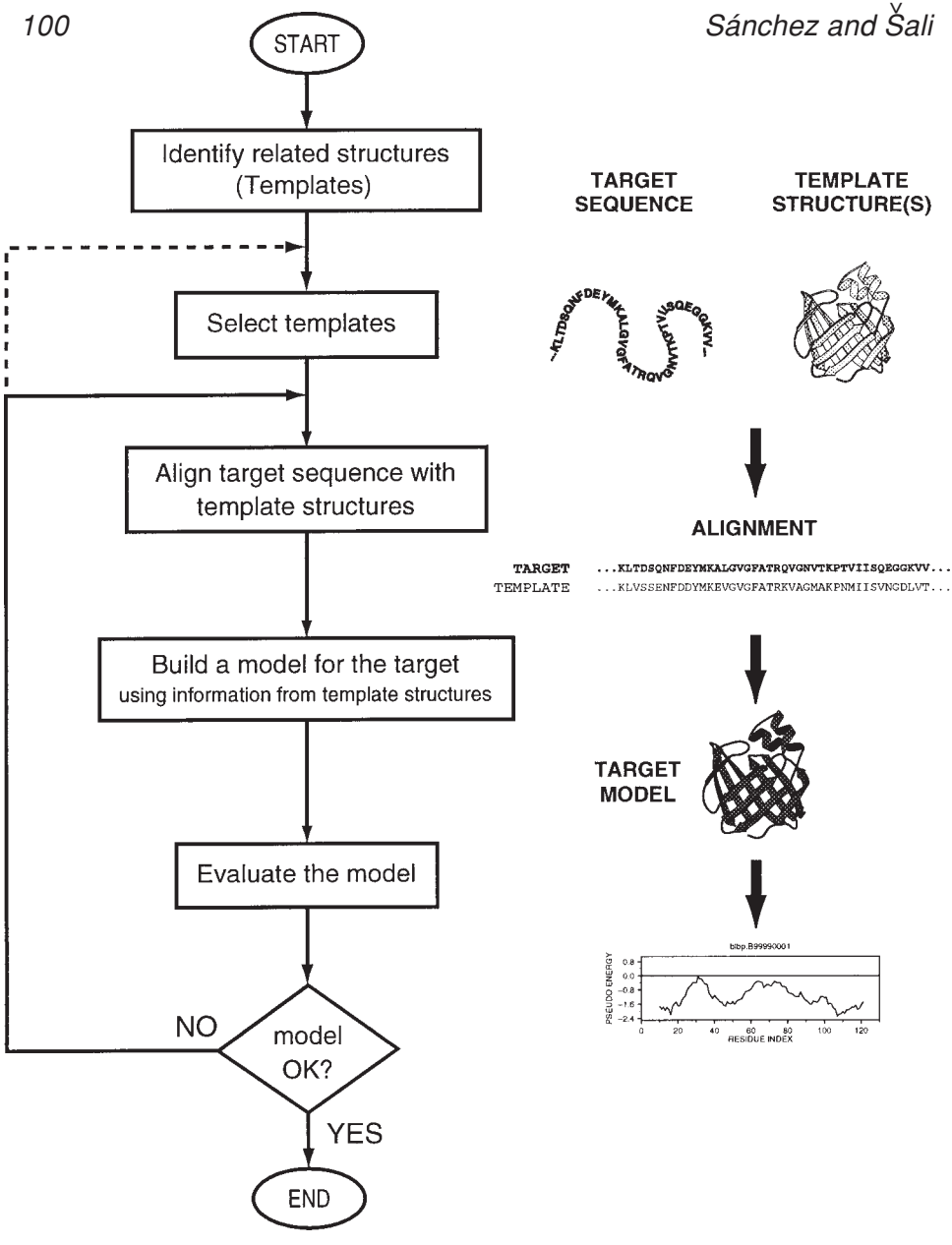


Fig. 1. Steps in comparative protein structure modeling. See text for description of each step.

1. The family of proteins that includes the target and the templates frequently can be organized in subfamilies. The construction of a multiple alignment and a phy-

Table 2
Programs and World Wide Web Servers Useful in Comparative Modeling

Name	Type[a]	World Wide Web or e-mail address	Reference
Template search			
BLASTP	S	www.ncbi.nlm.nih.gov/BLAST/	***(64)***
FASTA	S	www.pdb.bnl.gov/pdb-bin/pdbmain	***(65)***
123D	S	www-lmmb.ncifcrf.gov/~nicka/123D.html	***(66)***
PHDTHREADER	S	www.embl-heidelberg.de/predictprotein/predictprotein.html	***(67)***
UCLA-DOE FRSVR	S	www.doe-mbi.ucla.edu/people/frsvr/frsvr.html	***(68)***
PROFIT	P	www.came.sbg.ac.at	***(69)***
THREADER	P	globin.bio.warwick.ac.uk/~jones/threader.html	***(26)***
MATCHMAKER	P	www.scripps.edu/adam/home.html	***(27)***
Modeling			
COMPOSER	P	felix.bioc.cam.ac.uk/soft-base.html	***(70)***
CONGEN	P	bruc@dino.squibb.com	***(71)***
DRAGON	P	www.nimr.mrc.ac.uk/~mathbio/a-aszodi/dragon.html	***(42)***
MODELLER	P	guitar.rockefeller.edu/modeller/modeller.html	***(40)***
NAOMI	P		***(41)***
WHAT IF	P	www.sander.embl-heidelberg.de/whatif/	***(72)***
INSIGHTII	P	www.msi.com	(a)
LOOK	P	www.mag.com	***(37)***
QUANTA	P	www.msi.com	(a)
SYBYL	P	www.tripos.com	(b)
SWISS-MOD	S	www.expasy.ch/SWISS-MODEL.html	***(73)***
Model evaluation			
PROCHECK	P	www.biochem.ucl.ac.uk/~roman/procheck/procheck.html	***(48)***
WHATCHECK[c]	P	www.sander.embl-heidelberg.de/whatcheck/	***(49)***
PROSAII[c]	P	www.came.sbg.ac.at	***(47)***
PROCYON[d]	P	www.horus.com/sippl/	***(47,69)***
BIOTECH	S	biotech.embl-ebi.ac.uk:8400/	***(48,49)***
VERIFY3D	S	www.doe-mbi.ucla.edu/verify3d.html	***(46)***
ERRAT	S	www.doe-mbi.ucla.edu/errat_server.html	***(74)***

[a] S = server, P = program.
[b] (a) Molecular Simulations Inc., San Diego (b) Tripos, St Louis.
[c] PROCYON is a new software package that includes PROSAII, PROFIT, and other programs.
[d] The BIOTECH server uses PROCHECK and WHATCHECK for structure evaluation.

logenetic tree ***(31)*** can help in selecting the template from the subfamily that is closest to the target sequence.

2. The similarity between the "environment" of the template and the environment in which the target needs to be modeled should also be considered. The word "environment" is used here in a broad sense, including everything that is not the protein itself: solvent, pH, ligands, quaternary interactions, and the like (*see* **Subheadings 3.1.2.** and **4.2.**). In particular, the template(s) bound to the same or similar ligand(s) as the model should be used whenever possible.
3. The quality of the experimental template structure is another important factor in template selection. The resolution and R-factor of a crystallographic structure and the number of restraints per residue for a nuclear magnetic resonance (NMR) structure are indicative of the accuracy of the structure. This information can generally be obtained from the template PDB files or from the articles describing structure determination. If two templates have comparable sequence similarity to the target, the one determined at the highest resolution should be used.

The criteria for selecting templates also depend on the purpose of a comparative model. For instance, if a protein–ligand model is to be constructed, the choice of the template that contains a similar ligand is probably more important than the resolution of the template. On the other hand, if the model is to be used to analyze the geometry of the active site of an enzyme, it is preferable to use a high-resolution template. It is not necessary to select only one template. In fact, the use of several templates generally increases the model accuracy (*see* **Subheading 3.2.** and **Notes**).

2.3. Target-Template Alignment

To build a model, all comparative modeling programs depend on a list that establishes structural equivalences between the target and template residues. This is defined by the alignment of the target and template sequences. Although many template search methods will produce such an alignment, it is usually not the optimal target–template alignment. Search methods tend to be tuned for detection of remote relationships, not for optimal alignments. Therefore, once templates have been selected, a specialized method should be used to align them with the target sequence. The alignment is relatively simple to obtain when the target–template sequence identity is above 40%. In most such cases, an accurate alignment can be obtained automatically using standard sequence–sequence alignment methods. If the target–template sequence identity is lower than 40%, the alignment generally has gaps and needs manual intervention to minimize the number of misaligned residues. In these low-sequence identity cases, the alignment accuracy is the most important factor affecting the quality of the resulting model. Alignments can be improved by including structural information from the template. For example, gaps should be avoided in secondary-structure elements, in buried regions, or between two residues that are

far apart in space. Some alignment methods take such criteria into account (*see* **Subheading 3.1.3.**). However, it is always important to check and edit the alignment by inspecting the template structure visually, especially if the target–template sequence identity is low. A misalignment by only one residue position will result in an error of approximately 4 Å in the model because the current modeling methods cannot recover from errors in the alignment.

2.4. Model Building

Once an initial target–template alignment has been built, a variety of methods can be used to construct a 3D model for the target protein. The original and still most widely used method is modeling by rigid-body assembly ***(5,32,33)***. This method constructs the model from a few core regions and from loops and sidechains, that are obtained from dissecting related structures. Another family of methods, modeling by segment matching, relies on the approximate positions of conserved atoms from the templates to calculate the coordinates of other atoms ***(34–37)***. The third group of methods, modeling by satisfaction of spatial restraints, uses either distance geometry or optimization techniques to satisfy spatial restraints obtained from the alignment of the target sequence with the template structures ***(38–42)***. Accuracies of the various model-building methods are relatively similar when used optimally. Other factors such as template selection and alignment accuracy usually have a larger impact on the model accuracy, especially for models based on less than 40% sequence identity to the templates. However, it is important that a modeling method allows a degree of flexibility and automation, which will make it easier and faster to obtain better models. For example, a method should allow for an easy recalculation of a model when a change is made in the alignment; it should be straightforward to calculate models based on several templates; and the method should provide the tools to incorporate prior knowledge about the target (e.g., experimental data, or predicted features such as secondary-structure). Here we will describe automated comparative model building by satisfaction of spatial restraints as implemented in program MODELLER ***(40)***. Reviews of comparative model building methods have been published elsewhere ***(1–4)***. Several programs for comparative modeling are listed in **Table 2**.

2.4.1. Comparative Modeling with Program MODELLER

MODELLER is a computer program that models protein structure by satisfaction of spatial restraints (*see* the Appendix at the end of the chapter for information on how to obtain MODELLER). It can be used in all stages of comparative modeling described so far, including template search, target–template alignment and model building. Once a target–template alignment is obtained, the calculation of the 3D model of the target by MODELLER is com-

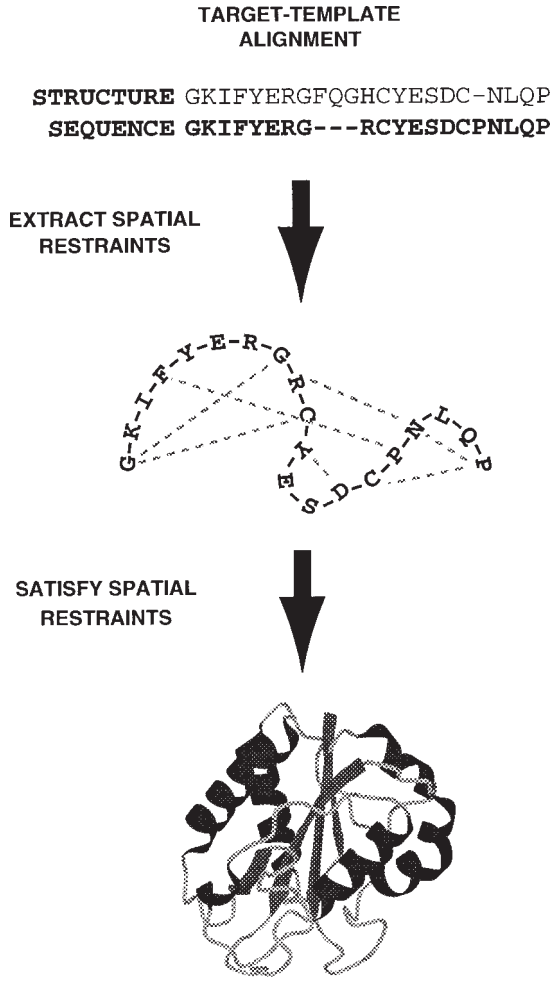


Fig. 2. Comparative modeling by program MODELLER. First, spatial restraints in the form of atom–atom distances and dihedral angles are extracted from the template structure(s). The alignment is used to determine equivalent residues between the target and the template. The restraints are combined into an objective function. Finally, the model for the target is optimized until a model that best satisfies the spatial restraints is obtained. This procedure is similar to the one used in structure determination by NMR.

pletely automated. The program extracts atom–atom distance and dihedral angle restraints on the target from the template structure(s) and combines them with general rules of protein structure such as bond length and angle preferences. The model is then calculated by an optimization procedure that minimizes violations of the spatial restraints (*see* **Fig. 2**). The procedure is

conceptually similar to the one used in the determination of protein structures from NMR data. More detailed descriptions of MODELLER can be found elsewhere ***(40,43–45)***.

2.5. Model Evaluation

After a model has been built, it is important to check it for possible errors. Two types of evaluation should be carried out: (1) "internal" evaluation of self-consistency that checks whether or not the model satisfies the restraints used to calculate it and (2) "external" evaluation that relies on information that was not used in calculating the model ***(46,47)***.

When the model is based on less than approx 30% sequence identity to the template, the first purpose of the external evaluation is to test whether or not a correct template was used. This is especially important when the alignment is only marginally significant or several alternative templates with different structures are to be evaluated. A complication is that at low similarities the alignment generally contains many errors, making it difficult to distinguish between an incorrect template on one hand and an incorrect alignment with a correct template on the other hand. It is only possible to recognize a correct template if the alignment is also approximately correct. This complication can sometimes be overcome by trying several alternative alignments for each template. One way to predict whether or not a template is correct is to compare the PROSAII Z-score ***(47)*** for the model and the template structure(s). The Z-score of a model is a measure of compatibility between its sequence and structure. The model Z-score should be comparable to the Z-score obtained for the template. However, this evaluation does not always work. It is sometimes possible that good models have bad Z-scores because the potential function used in PROSAII is not suitable for certain fold types.

The second kind of external evaluation is to recognize unreliable regions in the model. One way to approach this problem is to calculate an energy profile of the model by a program such as PROSAII. The profile reports the energy for each position in the model. It is sometimes possible to detect errors in the model because they appear as peaks of positive energy in the profile. Such regions of the model should be inspected carefully. Another way of finding unreliable regions of a model is to evaluate the stereochemistry (bond length and angles, dihedral angles, atom-atom overlaps, etc.) of the model with programs such as PROCHECK ***(48)*** and WHATCHECK ***(49)***. Although errors in stereochemistry are rare and less informative than errors detected by profiles, a cluster of stereochemical errors in the same segment of the model could indicate that the corresponding region also contains other errors (*see* **Table 2** for a list of evaluation programs and servers). Finally, an important evaluation tool is the experimental knowledge about the protein structure and its function. A model should

be consistent with experimental observations such as site-directed mutagenesis, crosslinking data, ligand binding, and so on.

2.6. The Cycle of Alignment–Modeling–Evaluation

In cases where the best template selection and alignment are not clear, one powerful way of improving a comparative model is to change the alignment and/or the template selection and recalculate the model iteratively until no improvement in the model is detected *(50,51)*. The more exhaustive is the exploration of the templates and alignments, the more likely it is that the accuracy of the final model will improve.

3. Examples

This section contains examples of typical comparative modeling cases. All the examples use program MODELLER and other freely available software. The first example shows each of the five steps of comparative modeling. The other three examples concentrate on specific variations of the basic modeling procedure. The examples are necessarily concise. For more information, the MODELLER manual *(52)* and the literature *(40,43–45,50,53–55)* should be consulted. All the example files can be obtained as explained in the Appendix at the end of the chapter.

3.1. Example 1: Modeling with a Single Template

THE CASE OF HUMAN BRAIN LIPID-BINDING PROTEIN

Brain lipid-binding protein (BLBP) is a brain-specific member of the fatty acid-binding protein (FABP) family. When the sequence of this protein was determined, its function was not known. Thus, a model of the structure of BLBP was built by comparative modeling, and combined with site-directed mutagenesis and binding experiments to understand its ligand specificity *(56)*. The individual modeling steps are described in **Subheading 3.1.1**.

3.1.1. Search for Templates

First, it is necessary to put the target sequence (BLBP sequence) into a format that is readable by MODELLER.MODELLER reads files is the format similar to the widely used FASTA format *(65)*.

File: `blbp.seq`

```
>P1;blbp
sequence:blbp:::::::
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEE
TSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA*
```

The first line contains `'>P1;` followed by the sequence name, `'blbp'` in this case. The second line has 10 fields (separated by colons ":") of which only

two are used in this case: `'sequence'` (indicating that the file contains a sequence without known structure) and `'blbp'`, the sequence name again. The rest of the file contains the sequence of BLBP, with `'*'` marking the end of the sequence. A search for structures that have similar sequence can be performed by the `SEQUENCE_SEARCH` command of MODELLER. The following command file (`TOP` file) will use the query sequence with the name `'blbp'` (`ALIGN_CODES`) from the file `blbp.seq`.

File: `search.top`

```
SET SEARCH_RANDOMIZATIONS = 100
SEQUENCE_SEARCH FILE = 'blbp.seq', ALIGN_CODES = 'blbp'
```

The `SEQUENCE_SEARCH` command has many options ***(52)***, but in this example only `SEARCH_RANDOMIZATIONS` is set to a nondefault value. `SEARCH_RANDOMIZATIONS` specifies the number of times the query sequence is randomized during the calculation of the significance score for each sequence–sequence comparison. The higher the number of randomizations, the more accurate the significance scores will be. To execute the `TOP` command file, type `'mod search.top'`.

3.1.2. Template Selection

The output of the search.top command file is written to the search.log file. If there is any problem with the command file, it will be reported in the log file.* At the end of this long file, MODELLER lists the hits sorted by alignment significance. The example shows only the top 10 hits.

File: `search.log`

```
  # CODE_1  CODE_2  LEN1  LEN2   NID    %ID    %ID    SCORE  SIGNI  SIGNI2  SIGNI3
-------------------------------------------------------------------------------
  1 blbp    1hmt     131   131    81  61.8   61.8   96904.   29.9  -999.0  -999.0
  2 blbp    1cbs     131   137    55  40.1   42.0   83725.   19.9  -999.0  -999.0
  3 blbp    1ifc     131   131    37  28.2   28.2   76909.   15.1  -999.0  -999.0
  4 blbp    1mdc     131   130    37  28.2   28.5   72299.    9.7  -999.0  -999.0
  5 blbp    1eal     131   127    34  26.0   26.8   69104.    9.1  -999.0  -999.0
  6 blbp    1iltA    131   143    25  17.5   19.1   64604.    3.8  -999.0  -999.0
  7 blbp    1bgk     131    37    18  13.7   48.6    7774.    3.5  -999.0  -999.0
  8 blbp    1tdx     131   133    25  18.8   19.1   64750.    3.3  -999.0  -999.0
  9 blbp    1thjA    131   213    43  20.2   32.8   59771.    3.3  -999.0  -999.0
 10 blbp    1amy     131   403    55  13.6   42.0   35790.    3.3  -999.0  -999.0
```

The most important columns in the `SEQUENCE_SEARCH` output are the `'CODE_2'`, `'%ID'` and `'SIGNI'` columns. The `'CODE_2'` column reports the code of the PDB sequence that was compared with the target sequence. The

*MODELLER always produces a log file. Errors and warnings in log files can be found by searching for the `'_E>'` and `'_W>'` strings (e.g., with the UNIX grep utility).

PDB code in each line is the representative of a group of PDB sequences that share 30% or more sequence identity to each other and have less than 30 residues or 30% sequence length difference. All the members of the group can be found in MODELLER's `CHAINS_3.0_30_XN.grp` file. The `'%ID'` column reports the percentage sequence identity between the two sequences (BLBP and each PDB sequence in this case). In general, a `'%ID'` value above 25–30% indicates a suitable template unless the alignment is short (less than 100 residues). A better measure of the significance of the alignment is given by the SIGNI column ***(52)***. A value above 6.0 is generally significant regardless of the sequence identity. In the foregoing example, five PDB structures have significant alignments with the BLBP sequence: 1HMT, 1CBS, 1IFC, 1MDC, 1EAL. All five proteins belong to the family of fatty acid binding proteins. The most similar to BLBP is 1HMT (human muscle fatty acid binding protein) with 61.8% sequence identity and a significance score of 29.9. By inspecting the PDB database (`http://www.pdb.bnl.gov`) or the `CHAINS_3.0_30_XN.grp` file, we find additional structures for the same sequence: 1HMS, 1HMR, 2HMB, and 1HMT all have identical sequences. The main difference between these four structures is the ligand to which the protein is bound. The ligands are stearic acid, oleic acid, elaidic acid, and 1-hexyldecanoic acid for 1HMT, 1HMS, 1HMR, and 2HMB, respectively. Thus, the four proteins are in different "environments." Assuming the interest is in studying the BLBP/oleic acid interaction, the template of choice is 1HMS. 1HMS is also a good template because it is a high resolution structure (1.4 Å). The coordinate file for 1HMS can be retrieved from the PDB database.

3.1.3. Target–Template Alignment

A good way of aligning a sequence (BLBP) and a structure (1HMS) is the `ALIGN2D` command in MODELLER. Although this command is based on the dynamic programming algorithm ***(57)***, it is different from standard sequence–sequence alignment methods because it takes into account structural information from the template when constructing an alignment. This is achieved through a variable gap penalty function that tends to place gaps in solvent exposed and curved regions, outside secondary-structure segments, and between two C_α positions that are close in space ***(58)***. As a result, the alignment errors are reduced to approximately one-half of those that occur with standard sequence alignment techniques. This becomes more important as the similarity (sequence identity) between the sequences decreases and the number of gaps increases. In this example, the similarity between template and target is so high that almost any alignment method with reasonable parameters will result in the same alignment. The following MODELLER TOP file will align

the BLBP sequence in file `blbp.seq` with the 1HMS structure in file 1hms.pdb, which is the coordinate file retrieved from the PDB database.

File: `align2d-1.top`

```
READ_MODEL FILE = '1hms.pdb'
SEQUENCE_TO_ALI ALIGN_CODES = '1hms'
READ_ALIGNMENT FILE = 'blbp.seq', ALIGN_CODES = ALLIGN_CODES 'blbp',
ADD_SEQUENCE = on
ALIGN2D
WRITE_ALIGNMENT FILE = 'blbp-1hms.ali', ALIGNMENT_FORMAT = 'PIR'
WRITE_ALIGNMENT FILE = 'blbp-1hms.pap', ALIGNMENT_FORMAT = 'PAP'
```

In the first line, MODELLER reads the 1HMS structure. The `SEQUENCE_TO_ALI` command transfers the sequence from the structure to the alignment in memory and assigns it the name `'1hms'` (`ALIGN_CODES`). The third line reads the BLBP sequence from file blbp.seq, assigns it the name `'blbp'` (`ALIGN_CODES`) and adds it to the alignment in memory (`'ADD_SEQUENCE = on'`). The fourth line calls the `ALIGN2D` command to perform the alignment. Finally, the alignment is written out in two formats, `'PIR'` and `'PAP'`. The PIR format is used by MODELLER in the subsequent model building stage. The PAP alignment is easier to inspect visually. The TOP file is executed by typing `'mod align2d-1.top'`. The output goes to files `blbp-1hms.ali` and `blbp-1hms.pap`:

File: `blbp-1hms.ali`

```
>P1;1hms
structureX:1hms:   1  :  :  131  :  :undefined:undefined:-1.00:-1.00
VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGVEFDETTA
DDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKE*
>P1;blbp
sequence:blbp:     :  :     :  :  :  :  0.00:  0.00
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEETSI
DDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA*
```

File: `blbp-1hms.pap`

```
 _aln.pos        10        20        30        40        50        60
1hms      VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNT
blbp      VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNT
 _consrvd **** **** ** *** *** **********   **** **   *      *  *****

 _aln.pos        70       80        90       100       110       120
1hms      EISFKLGVEFDETTADDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHG
blbp      EINFQLGEEFEETSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFG
 _consrvd ** * ** ** **  ***  ** * *** ** * ***** **   **  ***   *** *

 _aln.pos       130
1hms      TAVCTRTYEKE
blbp      DIVAVRCYEKA
 _consrvd   *  * ***
```

Due to the high similarity and equal lengths of BLBP and 1HMS, there are no gaps in the alignment. In the PAP format, all identical positions are marked with a '*'. The PIR format contains the starting and ending residue numbers from the 1HMS PDB file (1 and 131, in this case).

3.1.4. Model Building

Once a target–template alignment has been constructed, MODELLER calculates a 3D model of the target in a completely automated way. The following TOP file will generate one model for BLBP based on the 1HMS template structure and the alignment in file blbp-1hms.ali.

File: model1.top

```
INCLUDE
SET ALNFILE = 'blbp-1hms.ali'
SET KNOWNS = '1hms'
SET SEQUENCE = 'blbp'
SET STARTING_MODEL = 1
SET ENDING_MODEL = 1
CALL ROUTINE = 'model'
```

The first line includes many standard variable and routine definitions. The following five lines set parameter values for the 'model' routine. ALNFILE is the name of the file that contains the target–template alignment in the PIR format. KNOWNS is the name that corresponds to the template(s) (the known structure(s)) in ALNFILE (blbp-1hms.ali). SEQUENCE corresponds to the name of the target sequence in ALNFILE. STARTING_MODEL and ENDING_MODEL define the number of models that will be calculated for this alignment. Since STARTING_MODEL and ENDING_MODEL are the same in this case, only one model will be calculated. The last line in the file calls the 'model' routine that actually calculates the model. Typing 'mod model1.top' will execute the command file. The most important output files are:

1. model1.log: This file reports warnings, errors, and other useful information including restraints that remain violated in the final model.
2. blbp.B99990001: The actual model coordinates in the PDB format. This file can be viewed by any program that reads the PDB format (e.g., RASMOL ***[59]*** http://www.umass.edu/microbio/rasmol/).

3.1.5. Model Evaluation

As discussed before, there are many alternatives for model evaluation. In this example, PROSAII ***(47)*** is used to evaluate the model fold and PROCHECK ***(48)*** is used to check the model's stereochemistry. Before doing any external evaluation of the model, one should check the log file from the

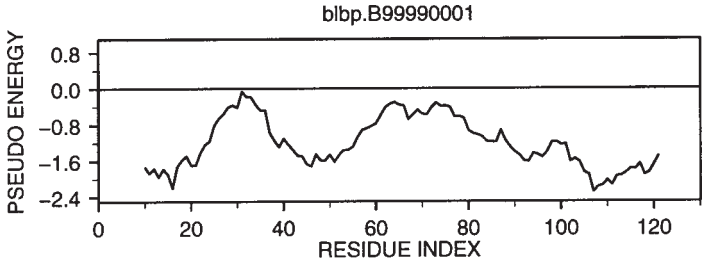


Fig. 3. PROSAII *(47)* energy profile for the BLBP model (*see* Example 1).

modeling run for errors (`model1.log` in this example) and restraint violations (*see* the MODELLER manual for more details on this *(52)*).

First, an energy profile of the model is obtained using the PROSAII program. It is sometimes possible to identify errors in the model because they appear as regions of positive energy in the PROSAII profile. In the case of the BLBP model, no errors were found (*see* **Fig. 3**). This is not surprising given the high similarity between the template and the target. PROSAII is not able to detect all errors, but if a region of the model has a positive profile, one should try alternative alignments in that region.* The stereochemistry of the model can be checked by program PROCHECK. The output of PROCHECK is a series of POSTSCRIPT files with evaluations of different aspects of the model's stereochemistry. One of the most important charts is the Ramachandran plot (*see* **Fig. 4**) which points out those residues that have anomalous combinations of ϕ and ψ angles. As mentioned before, a few deviations of this type are usual even in experimentally determined structures. For example, in **Fig. 4**, alanine 6 and aspartate 98 are in disallowed regions of the plot. However, if several errors cluster in the same region of the model, it is likely that other errors, such as misalignments, have occurred. In this example, both PROSAII and PROCHECK confirm that a good quality model was obtained.

3.2. Example 2: Modeling of a Protein/Ligand Complex

ADDING OLEIC ACID TO BLBP

A better way of analyzing the interaction between BLBP and oleic acid is to add the ligand molecule to the model. To add the ligand that is present in the

*When using profiles, one should always calculate the profile for the template as well. Sometimes a positive peak appears in the model's profile as a consequence of a similar peak in the template's profile. This does not necessarily mean that there is an error in the template structure but more likely the evaluation method is reporting a false error for that particular structure. In such a case, the positive peak in the model probably does not correspond to an error.

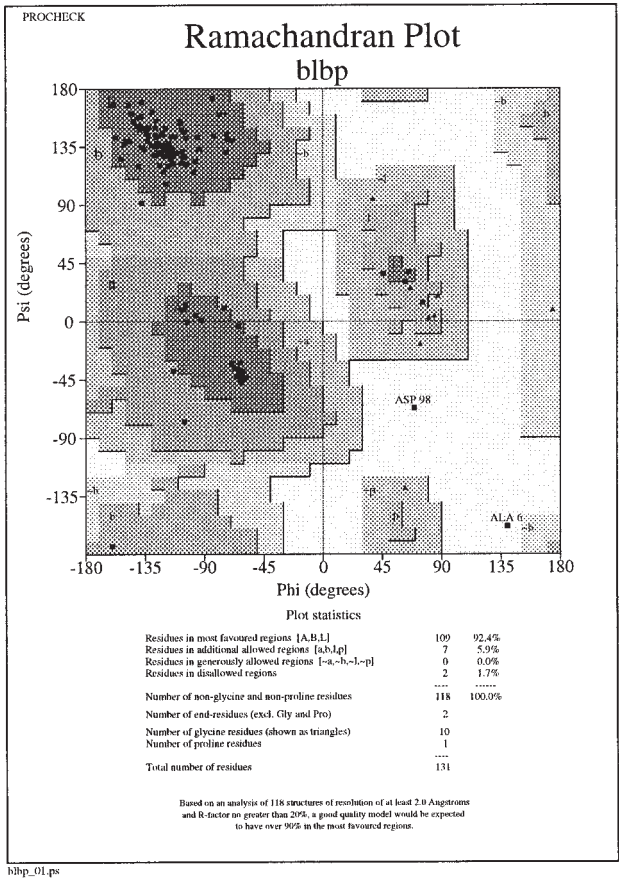


Fig. 4. Evaluation of model stereochemistry. The Ramachandran plot was created for the BLBP model by the PROCHECK program ***(48)*** (*see* Example 1).

1HMS template (oleic acid) to the BLBP model, all we need to modify is the alignment file `blbp-1hms.ali` and the modeling TOP file `model1.top`. The new files are shown next:

File: `blbp-1hms-ola.ali`

```
>P1;1hms
structureX:1hms:   1  :  :  133 : :undefined:undefined:-1.00:-1.00
VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGVEFDETTA
DDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKE.*
>P1;blbp
sequence:blbp:     : :     : : : : 0.00: 0.00
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEETSI
DDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA.*
```

The second line in the alignment file now specifies that the template is to be used from residue 1 to residue 133 (the oleic acid molecule is residue 133 in 1HMS). The second change in this file is the appearance of the '.' character at the end of each sequence. This character represents the oleic acid molecule in the alignment.*

The modeling command file `model2.top` has two changes with respect to `model1.top`. First, the name of the alignment file assigned to `ALNFILE` was updated. The second change is the addition of `'SET HETATM_IO = on'`. `HETATM_IO` is a flag that indicates to MODELLER whether or not heteroatoms (e.g., nonstandard residues, such as oleic acid) should be read in from the PDB files.

File: `model2.top`

```
INCLUDE
SET ALNFILE = 'blbp-1hms-ola.ali'
SET KNOWNS = '1hms'
SET SEQUENCE = 'blbp'
SET STARTING_MODEL = 1
SET ENDING_MODEL = 1
SET HETATM_IO = on
CALL ROUTINE = 'model'
```

MODELLER can be started with this TOP file by typing `'mod model2.top'`. The BLBP model containing the oleic acid residue docked into the binding pocket will be written to `blbp.B99990001`.

It is possible to add ligands which are not present in the template by using predefined ligands in the MODELLER residue topology libraries. These ligands include water molecules, metal ions, heme groups, and others. To place such ligands in the model, additional protein–ligand distance restraints have to be supplied to MODELLER ***(52)***.

3.3. Example 3: Modeling Based on More Than One Template

IMPROVING THE BLBP MODEL

Using more than one template usually improves the quality of the model because MODELLER is generally able to combine the best regions from each template when constructing the model ***(50)***. Another good template for modeling of BLBP is adipocyte lipid binding protein (ALBP), which is 56% identical to BLBP. Furthermore, a structure of ALBP in complex with oleic acid is available (PDB code 1LID). To calculate a model for BLBP using both templates, an alignment of all three sequences was constructed.

*The dot ('.') character in MODELLER represents a generic residue called a "block" residue. It can be used to represent any nonstandard residue. For more details, *see* the MODELLER manual ***(52)***.

File: align2d-3.top

```
SET ALIGN_CODES = '1hms' '1lid'
SET ATOM_FILES = '1hms.pdb' '1lid.pdb'
MALIGN3D
SET ADD_SEQUENCE = on, ALIGN_BLOCK = NUMB_OF_SEQUENCES
READ_ALIGNMENT FILE = 'blbp.seq', ALIGN_CODES = ALIGN_CODES 'blbp'
ALIGN2D
WRITE_ALIGNMENT FILE = 'blbp-1hms-1lid.ali'
WRITE_ALIGNMENT FILE = 'blbp-1hms-1lid.pap', ALIGNMENT_FORMAT = 'PAP'
```

The first three lines in the Top file produce a structural alignment of 1HMS and 1LID using the MALIGN3D command. The BLBP sequence in file blbp.seq is then added to the structural alignment using the ALIGN2D command (lines 4–6). The resulting alignment file in the PIR format, blbp-1hms-1lid.ali, has to be edited manually to include the oleic acid residues as block residues (*see* previous example). The edited file is shown here.

File: blbp-1hms-1lid-2.ali

```
>P1;1hms
structureX:1hms:1    : :133    : :undefined:undefined:-1.00:-1.00
VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGVEFDETTA
DDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKE.*
>P1;1lid
structureX:1lid:1    : :131    : :undefined:undefined:-1.00:-1.00
CDAFVGTWKLVSSENFDDYMKEVGVGFATRKVAGMAKPNMIISVNGDLVTIRSESTFKNTEISFKLGVEFDEITA
DDRKVKSIITLDGGALVQVQKWDGKSTTIKRKRDGDKLVVECVMKGVTSTRVYERA-*
>P1;blbp
sequence:blbp:     : :     : : : : 0.00: 0.00
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEETSI
DDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA.*
```

Because the conformations of the oleic acid molecules in 1HMS and 1LID are different, only the 1HMS oleic acid is used as a template. This is done by replacing the 1LID oleic acid residue in the alignment by a gap character ('-'). It would be straightforward to produce a BLBP model with the 1LID oleic acid molecule by changing the blbp-1hms-1lid.ali alignment. Models for both complexes could be used to design mutants that discriminate between the two binding modes.

Using the TOP file shown below, MODELLER will generate an "ensemble" of five models. Because MODELLER uses different starting coordinates for each model, it is possible that the final models have different conformation in some regions, especially for sidechains. Those regions of the structure that are more variable among the models are likely to be modeled less reliably than the structurally more conserved regions.

File: `model3.top`

```
INCLUDE
SET ALNFILE = 'blbp-1hms-1lid-2.ali'
SET KNOWNS = '1hms' '1lid'
SET SEQUENCE = 'blbp'
SET STARTING_MODEL = 1
SET ENDING_MODEL = 5
SET HETATM_IO = on
CALL ROUTINE = 'model'
```

After execution of the Top file, the models will be contained in five files `blbp.B99990001` through `blbp.B99990005`. A quick way of evaluating the variability of the models is to superpose their structures. This can be done with the `MALIGN3D` command of MODELLER.

File: `malign3d.top`

```
SET ATOM_FILES = 'blbp.B99990001' 'blbp.B99990002' 'blbp.B99990003' ;
 'blbp.B99990004' 'blbp.B99990005'
SET WRITE_FIT = on
MALIGN3D
```

The first line specifies the five coordinate files containing the models. The second line directs MODELLER to write the superposed structures to new files. The `MALIGN3D` command finally superposes the five models and actually writes the superposed structures in the new orientations to five files `blbp.B99990001.fit` through `blbp.B99990005.fit`. An easy way to view the superposed models is to concatenate the files with the UNIX `'cat'` command, `'cat blbp.B9999*.fit > sup.pdb'` and display the `sup.pdb` file with RASMOL. The superposed models are shown in **Fig. 5**.

The "best" model can be selected by looking at the value of the MODELLER objective function in the second line of the model PDB files and choosing the one with the lowest value. The value of the objective function in MODELLER is not an absolute measure. It can only be used to compare models calculated from the same templates and alignments, and rank them accordingly.

File: `blbp.B99990001`

```
REMARK Produced by MODELLER: 19-Dec-97 00:49:51                          1
REMARK MODELLER OBJECTIVE FUNCTION:      623.0785
ATOM      1  N    VAL   1      27.443  41.227  41.628  1.00  0.15      1SG   2
ATOM      2  CA   VAL   1      26.733  41.202  42.923  1.00  0.15      1SG   3
ATOM      3  CB   VAL   1      27.576  41.899  43.956  1.00  0.15      1SG   4
.
.
.
```

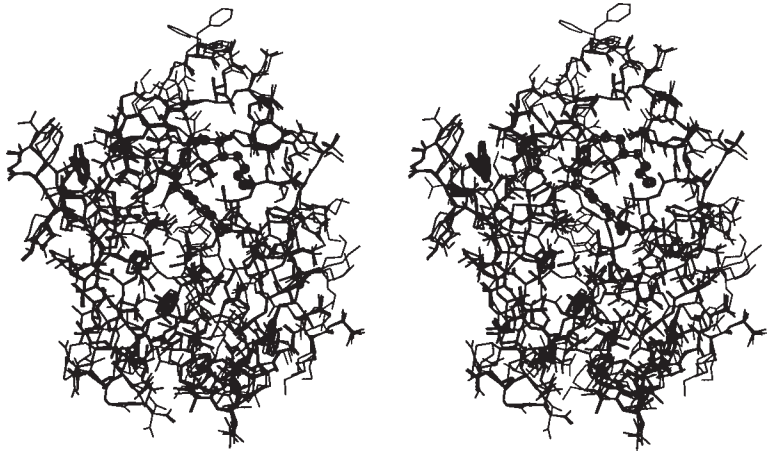


Fig. 5. Stereo plot of the superposition of five BLBP models from Example 3. The oleic acid molecule is shown in ball-and-stick representation ***(75)***.

3.4. Example 4: The Alignment–Modeling–Evaluation Cycle

THE CASE OF *Haloferax Volcanii* DIHYDROFOLATE REDUCTASE

Several structures of dihydrofolate reductase (DHFR) are known. However, the structure of DHFR from *Haloferax volcanii* was not known and its sequence identity with DHFRs of known structure is rather low (approx 30%). A model of *H. volcanii* DHFR (HVDFR) was constructed before the experimental structure was solved. Once the crystallographic structure was available, it was possible to compare it with the model ***(50)***. This example illustrates the power of the iterative alignment–modeling–evaluation approach to comparative modeling.

Of all the available DHFR structures, HVDHFR has the sequence most similar to DHFR from *Escherichia coli*. The PDB entry 4DFR corresponds to a high resolution (1.7 Å) *E. coli* DHFR structure. It contains two copies of the molecule — named chain A and chain B. According to the authors, the structure for chain B is of better quality than that of chain A ***(60)***. The following TOP file aligns HVDFR and chain B of 4DFR.

File: `align2d-4.top`

```
READ_MODEL FILE = '4dfr.pdb', MODEL_SEGMENT '@:B' 'X:B'
SEQUENCE_TO_ALI ALIGN_CODES = '4dfr'
READ_ALIGNMENT FILE = 'hvdfr.seq', ALIGN_CODES = ALIGN_CODES 'hvdfr', ADD_SEQUENCE
= on
ALIGN2D
WRITE_ALIGNMENT FILE = 'hvdfr-4dfr.ali'
WRITE_ALIGNMENT FILE = 'hvdfr-4dfr.pap', ALIGNMENT_FORMAT = 'PAP', ;
ALIGNMENT_FEATURES = 'indices helix beta'
```

The new options used in this example include `MODEL_SEGMENT`, which is used to indicate chain B of 4DFR; and `ALIGNMENT_FEATURES`, which is used to output information such as secondary-structure, to the alignment file in the PAP format.

File: `hvdfr-4dfr.pap`

```
 _aln.pos        10        20        30        40        50        60
4dfr      M-ISLIAALAVDRVIGMENAMPW-NLPADLAWFKRNTLDKPVIMGRHTWESIGRPLPGRK
hvdfr     MELVSVAALAENRVIGRDGELPWPSIPADKKQYRSRIADDPVVLGRTTFESMRDDLPGSA
 _helix                           999999999999      999999999
 _beta    9 999999999                          999999           999

 _aln.pos        70        80        90       100       110       120
4dfr      NIILSSQPGT--DDRVTWVKSVDEA--IAACGDVPEIMVIGGGRVYEQFLPKAQKLYLTH
hvdfr     QIVMSRSERSFSVDTAHRAASVEEAVDIAASLDAETAYVIGGAAIYALFQPHLDRMVLSR
 _helix                    99999  99999         99999999
 _beta    99999          99999             9999999           9999999

 _aln.pos       130       140       150       160
4dfr      IDAEVEGDTHFPDYEPDDWESVFSEFHDADAQNSHSYCFKILERR
hvdfr     VPGEYEGDTYYPEWDAAEWELDAETDHEGF--TLQEWVRSASSR
 _helix
_beta     99                  999999999999      999999999999
```

Using the alignment file `hvdfr-4dfr.ali`, an initial model is calculated.

File: `model4.top`

```
INCLUDE
SET ALNFILE = 'hvdfr-4dfr.ali'
SET KNOWNS = '4dfr'
SET SEQUENCE = 'hvdfr'
SET STARTING_MODEL = 1
SET ENDING_MODEL = 1
CALL ROUTINE = 'model'
```

Because the sequence identity between 4DFR and HVDFR is relatively low (30%), the automated alignment is likely to contain errors. The PROSAII evaluation of the model (*see* **Fig. 6**, upper panel) shows two regions with positive energy. The first region is around residue 85, the second region is at the C-terminal end of the protein. Referring to the target–template alignment shown, (`hvdfr-4dfr.pap`), it is easy to understand why the first positive peak appears. The insertion between position 85 and 88 of the alignment was placed in the middle of an a-helix in the template (the "9" characters on the first line below the sequence mark the helices). Moving the insertion to the end of the α-helix may improve the model. The second problem, which occurs in the C-terminal region of the alignment, is less clear. The deletion in that region of

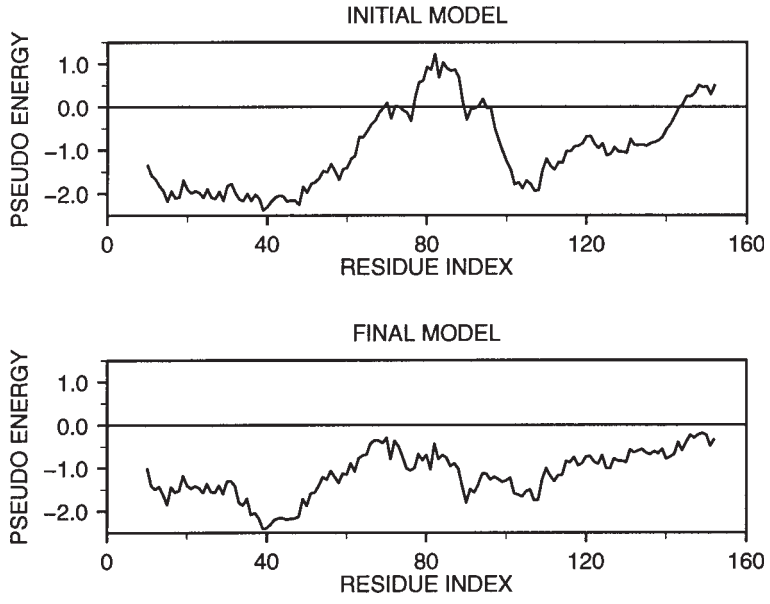


Fig. 6. PROSAII energy profiles for the initial and final HVDFR models (*see* Example 4).

the alignment corresponds to the loop between the last two β-strands of 4DFR (a β-hairpin). Since the profile suggests that this region is in error, an alternative alignment should be tried. One possibility is that the deletion is actually longer, making the C-terminal β-hairpin shorter in HVDFR. One plausible alignment based on this considerations is shown here.

File: `hvdfr-4dfr-2.pap`

```
 _aln.pos        10        20        30        40        50        60
4dfr      M-ISLIAALAVDRVIGMENAMPW-NLPADLAWFKRNTLDKPVIMGRHTWESIGRPLPGRK
hvdfr     MELVSVAALAENRVIGRDGELPWPSIPADKKQYRSRIADDPVVLGRTTFESMRDDLPGSA
 _helix                           999999999999      999999999
 _beta    9 999999999                          999999           999

 _aln.pos        70        80        90       100       110       120
4dfr      NIILSSQPGT--DDRVTWVKSVDEAIAACG--DVPEIMVIGGGRVYEQFLPKAQKLYLTH
hvdfr     QIVMSRSERSFSVDTAHRAASVEEAVDIAASLDAETAYVIGGAAIYALFQPHLDRMVLSR
 _helix                      99999  99999         99999999
 _beta    99999          99999              9999999           9999999

 _aln.pos       130       140       150       160
4dfr  IDAEVEGDTHFPDYEPDDWESVFSEFHDADAQNSHSYCFKILERR----
hvdfr VPGEYEGDTYYPEWDAAEWELDAETDHE------GFTLQEWVRSASSR
 _helix
_beta    99                999999999999      999999999999
```

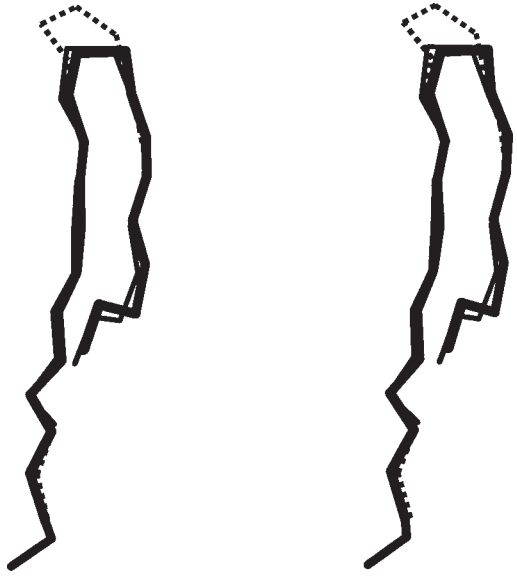


Fig. 7. Stereo plot of the superposition of the C-terminal region of the HVDFR models and the experimental structure (*see* Example 4). Initial model, dotted line; final model, thick line; experimental structure, thin line.

A new model was calculated. Its PROSAII profile is shown in **Fig. 6** (lower panel). Both positive peaks disappeared and the new profile does not contain any positive regions. **Figure 7** shows the comparison of the C-terminal β-hairpin of both models and the actual experimental structure ***(50)***. This confirms that the correct choice for the final alignment was made and that PROSAII was indeed able to detect the error in the initial alignment.

The examples shown here correspond only to the most basic comparative modeling problems. MODELLER can be used for many more complex projects, such as multiple chain models (multimers or protein–protein complexes), symmetry-constrained models, modeling of chimeric structures, and so on. It is also possible to add experimental or predicted data in the form of additional restraints (e.g., NMR or fluorescence distance measurements, disulfide bridges, secondary-structure prediction, and the like). For details and more examples, *see* the MODELLER manual ***(52)***.

4. Notes

4.1. Errors in Comparative Modeling

As the similarity between the target and the templates decreases, the errors in a model increase (*see* **Subheading 4.2.**). Errors in comparative models can be ex-

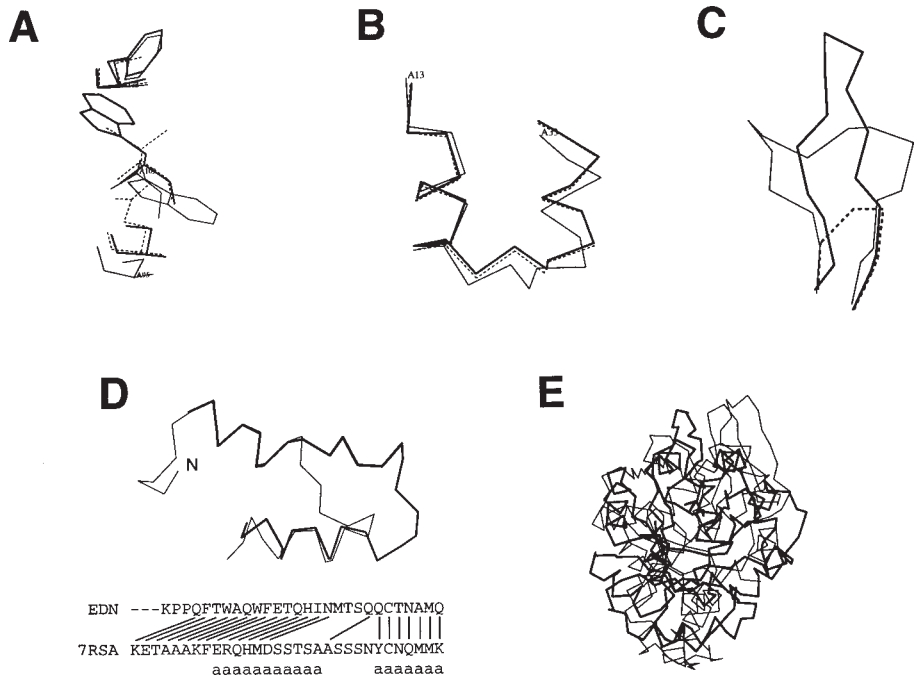


Fig. 8. Typical errors in comparative modeling ***(54)*** **(A)** Errors in sidechain packing. The `Trp` 109 residue in the crystal structure of mouse cellular retinoic acid binding protein I (thin line) is compared with its model (thick line), and with the template mouse adipocyte lipid-binding protein (broken line). **(B)** Distortions and shifts in correctly aligned regions. A region in the crystal structure of mouse cellular retinoic acid binding protein I (thin line) is compared with its model (thick line), and with the template fatty acid binding protein (broken line). **(C)** Errors in regions without a template. The C_α trace of the 112–117 loop is shown for the X- ray structure of human eosinophil neurotoxin (thin line), its model (thick line), and the template ribonuclease A structure (residues 111–117; broken line). **(D)** Errors due to misalignments. The N-terminal region in the crystal structure of human eosinophil neurotoxin (thin line) is compared with its model (thick line). The corresponding region of the alignment with the template ribonuclease A is shown. The black lines show correct equivalences, i.e., residues whose C_α atoms are within 5 Å of each other in the optimal least-squares superposition of the two X-ray structures. The `'a'` characters in the bottom line indicate helical residues (e) Incorrect template. The X-ray structure of α-trichosanthin (thin line) is compared with its model (thick line), which was calculated using indole–3-glycerophosphate synthase as a template.

plained based on the facts that the model resembles the templates as much as possible, and that the modeling procedure cannot recover from misalignments. The typical errors in comparative models include ***(45,50,54)*** (*see* **Fig. 8**):

1. Errors in sidechain packing: As the sequences diverge, the packing of sidechains in the protein core changes. Sometimes even the conformation of identical sidechains is not conserved, a pitfall for many comparative modeling methods. The sidechain errors are generally not important unless they occur in regions that are involved in function, such as active sites and ligand-binding sites.
2. Distortions and shifts in correctly aligned regions: As a consequence of sequence divergence, the mainchain conformation also changes even if the overall fold remains the same (*see* **Fig. 9**). Therefore, it is possible that in some correctly aligned segments of a model, the template is locally different (<3 Å) from the target, resulting in an incorrect model in that region. Sometimes the target–template differences are not due to differences in sequence but are a consequence of artifacts in structure determination (e.g., crystal packing) or structure determination in different environments. The simultaneous use of several templates minimizes this kind of error ***(50)***.
3. Errors in regions without a template: Segments of the target sequence that have no equivalent region in the template structure (insertions) are the most difficult regions to model. If the insertion is relatively short (usually less than eight residues), some methods are able to predict reliably the conformation of the backbone, but they usually need special attention ***(1,2)***. Conditions for the successful prediction of the conformation of an insertion are the correct alignment and an accurately modeled environment around the insertion. Insertions longer than 8 residues are generally not possible to model correctly with the current methods.
4. Errors due to misalignments: The largest source of errors in comparative modeling are misalignments, especially when the target–template similarity decreases below 40% (*see* **Fig. 9**). For example, at 30% sequence identity on the average 20% of the residues are misaligned ***(61)***. A misalignment of a residue by a single position produces a positional error of approx 4 Å in the model. The current comparative modeling methods cannot recover from alignment errors because the model building procedure is not able to modify the target–template alignment. However, alignment errors can be corrected or avoided in two ways. First, it is usually possible to use a large number of sequences, even if most of them do not have known structures, to construct a family alignment. Multiple alignments are generally more reliable than pairwise alignments ***(62)***. The second way of improving the alignment is to modify those regions of the alignment that correspond to predicted errors in the model in an iterative way, as described in **Subheading 2.6**.
5. Incorrect templates: This is a potential problem when distantly related proteins are used as templates (i.e., less than 30% sequence identity). As discussed before, models based on incorrect templates can generally be identified at the evaluation stage. The largest practical problem is to distinguish between a model based on an incorrect template and a model based on a mostly incorrect alignment with a correct template. In both cases, the evaluation methods will predict an unreliable model. A possible solution to this problem is to explore several different alignments for the target–template pair. In theory, it should be possible to find align-

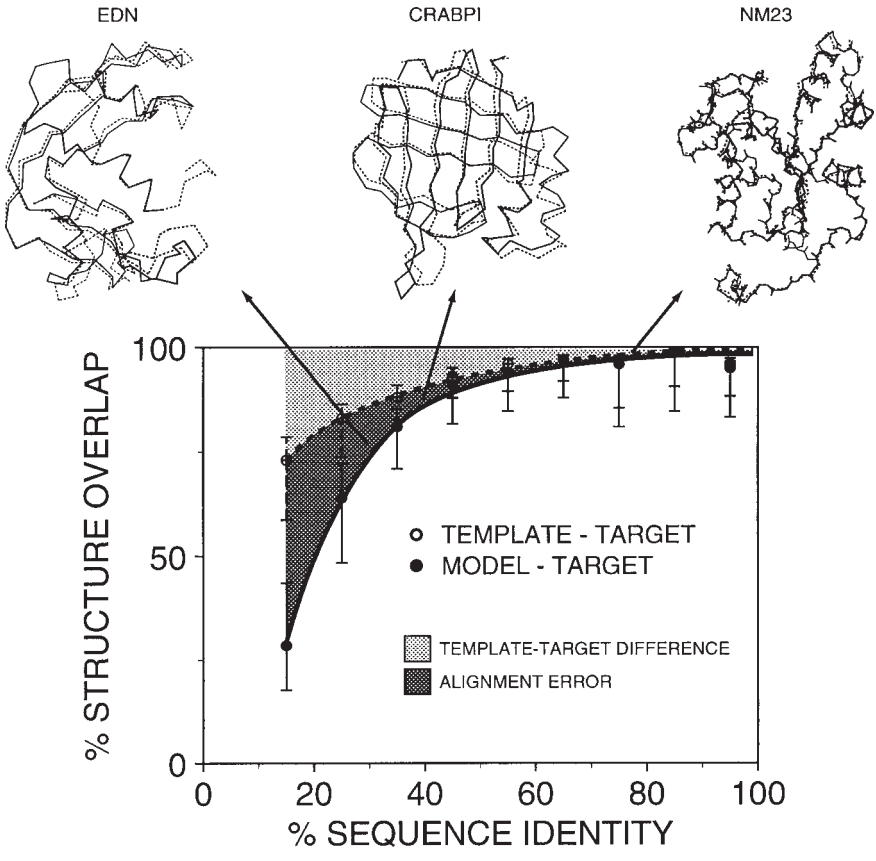


Fig. 9. Average model accuracy as a function of sequence identity. As the sequence identity between the target sequence and the template structure decreases, the average structural similarity between the template and the target also decreases (dotted line, open circles). Structural overlap is defined as the fraction of equivalent C_α atoms. For the comparison of the model with the actual structure (filled circles), two C_α atoms were considered equivalent if they were within 3.5 Å of each other and belonged to the same residue. For comparison of the template structure with the actual target structure (open circles), two C_α atoms were considered equivalent if they were within 3.5 Å of each other after alignment and rigid-body superposition by the `ALIGN3D` command in MODELLER. At high-sequence identities, the models are close to the templates, and therefore also close to the experimental target structure (solid line, filled circles). At low-sequence identities, errors in the target–template alignment become more frequent and the structural similarity of the model with the experimental target structure falls below the target–template structural similarity. The difference between the model and the actual target structure is a combination of the target–template differences (light area) and the alignment errors (dark area). The figure was constructed by calculating 3993 comparative models based on single templates of varying similarity

ments that are accurate enough to produce a good model if the template is correct. However, in practice the number of possibilities that need to be explored to find a sufficiently accurate alignment may be too large. Therefore, the only way to assure that a certain template is incorrect for a particular target is by finding another template with different structure that produces a better model for the same target.

4.2. Relationship Between Target–Template Similarity and Model Accuracy

The quality of a model can be approximately predicted from the sequence similarity between the target and the template (**Fig. 9**). Sequence identity above 30% is a relatively good predictor of the expected accuracy of a model. However, other factors, including the environment, can strongly influence the accuracy of a model. For instance, some calcium-binding proteins undergo large conformational changes when bound to calcium. If a calcium-free template is used to model the calcium-bound state of a target, it is likely that the model will be incorrect irrespective of the target–template similarity. This also applies to experimental determination of protein structure. A structure must be determined in the functionally meaningful environment. If the target–template sequence identity falls below 30%, the sequence identity becomes unreliable as a measure of expected accuracy of a single model. The reason is that the dispersion of the model–target structural overlap increases with the decrease in sequence identity. Below 30% sequence identity, it is relatively frequent to obtain models that deviate significantly, in both directions, from the average accuracy. It is in such cases, that model evaluation methods (*see* **Subheading 2.5.**) are most important to use.

4.3. Are Comparative Models Better than Their Templates?

In general, models are as close to the target structure as the templates, or slightly closer if the alignment is correct ***(50)***. This is not a trivial achievement because of the many residue substitutions, deletions, and insertions that occur when the sequence of one protein is transformed into the sequence of another. Even in a favorable modeling case with a template that is 50% identical to the target, half of the sidechains change and have to be packed in the protein core such that they avoid atom clashes and violations of stereochemical restraints.

---

to the targets. All targets had known (experimentally determined) structures, and therefore the comparison of the models and templates with the experimental structures was possible ***(63)***. The top part of the figure shows three models (solid line) compared with their corresponding experimental structures (dotted line). The models were calculated with MODELLER in a completely automated fashion before the experimental structures were available ***(54)***. The arrows indicate the target–template similarity in each case.

When more than one template is used for modeling, it is sometimes possible to obtain a model that is significantly closer to the target structure than any of the templates ***(50)***. This is so because the model tends to inherit the best regions from each template, thus minimizing some of the distortions in the correctly aligned regions. Alignment errors are the main factor that may make models worse than the templates. However, to represent the target, it is always better to use a comparative model rather than the template. The reason is that the errors in the alignment affect similarly the use of the template as a representation of the target as well as the comparative model based on the same template ***(50)***.

4.4. Establishing Remote Protein–Protein Relationships by Model Evaluation

Evaluation of a comparative model implied by a target–template alignment is a powerful way of confirming the significance of the alignment. It is often the case that a sequence similarity search of a database results in only a marginal or nonsignificant hit even when two proteins are homologous. A good way of confirming such a hit, when one of the proteins happens to have a known structure, is to build a comparative model for the sequence of unknown structure. If the resulting model is of good quality, according to the evaluation methods described in **Subheading 2.5.**, it is likely that the two proteins have similar structures ***(50,51,63)***. This approach is also useful when structural similarity is suspected in the absence of sequence similarity.

Acknowledgments

The authors are grateful to Azat Badretdinov, Eric Feyfant, and András Fiser for discussions about comparative modeling. RS is a Howard Hughes Medical Institute predoctoral fellow. AS is an Alexandrine and Alexander Sinsheimer Medical Fund Scholar. The investigation has also been aided by grants from the National Institutes of Health (GM 54762) and the National Science Foundation (BIR-9601845).

Appendix: How to Obtain MODELLER and the Example Files

MODELLER

MODELLER is freely available to academic users. It runs on most UNIX systems, including PCs running LINUX. The program and data files can be accessed on the Web at `http://guitar.rockefeller.edu/modeler/modeller.html` or can be downloaded by FTP from `guitar.rockefeller.edu` using the anonymous account. MODELLER, with a graphical interface, is also available as part of QUANTA, INSIGHTII, and GENEEXPLORER (Molecular Simulations Inc., San Diego, CA, e-mail: `dje@msi.com`).

Example Files

All example files used in the text, some additional data files, as well as the links in **Table 2** can be accessed on the Web at `http://guitar.rockefeller.edu/modeller/psp/`

References

1. Sánchez, R. and Šali, A. (1997) Advances in comparative protein-structure modeling. *Curr. Opin. Struct. Biol.* **7,** 206–214.
2. Marti-Renom, M. A., Stuart, A., Fiser, A., Sá-nchez, R., and Šali, A. (????) Comparative protein structure modeling of genes and genomes. *Ann. Rev. Biophys. Biomolec. Struct.*, in press.
3. Johnson, M. S., Srinivasan, N., Sowdhamini, R., and Blundell, T. L. (1994) Knowledge-based protein modelling. *CRC Crit. Rev. Biochem. Mol. Biol.* **29,** 1–68.
4. Bajorath, J., Stenkamp, R., and Aruffo, A. (1994) Knowledge-based model building of proteins: concepts and examples. *Protein Sci.* **2,** 1798–1810.
5. Blundell, T. L., Sibanda, B. L., Sternberg, M. J. E., and Thornton, J. M. (1987) Knowledge-based prediction of protein structures and the design of novel molecules. *Nature* **326,** 347–352.
6. Chothia, C. and Lesk, A. M. (1986) The relation between the divergence of sequence and structure in proteins. *EMBO J.* **5,** 823–826.
7. Hubbard, T. J. P. and Blundell, T. L. (1987) Comparison of solvent inaccessible cores of homologous proteins: definitions useful for protein modelling. *Protein Eng.* **1,** 159–171.
8. Oliver, S. G. (1996) From DNA sequence to biological function. *Nature* **379,** 597–600.
9. Koonin, E. V. and Mushegian, A. R. (1996) Complete genome sequences of cellular life forms: glimpses of theoretical evolutionary genomics. *Curr. Biol.* **6,** 757–762.
10. Dujon, B. (1996) The yeast genome project: what did we learn? *Trends Genet.* **12,** 263–270.
11. Matsumoto, R., Šali, A., Ghildyal, N., Karplus, M., and Stevens, R. L. (1995) Packaging of proteases and proteoglycans in the granules of mast cells and other hematopoietic cells. A cluster of histidines in mouse mast cell protease-7 regulates its binding to heparin serglycin proteoglycan. *J. Biol. Chem.* **270,** 19524–19531.
12. Lesk, A. M. and Chothia, C. (1980) How different amino acid sequences determine similar protein structures: the structure and evolutionary dynamics of the globins. *J. Mol. Biol.* **136,** 225–270.
13. Rost, B. and Sander, C. (1996) Bridging the protein sequence–structure gap by structure predictions. *Annu. Rev. Biophys. Biomol. Struct.* **25,** 113–136.
14. Benson, D. A., Boguski, M. S., Lipman, D. J., Ostell, J., and Ouellette, B. F. F. (1997) GenBank. *Nucleic Acids Res.* **26,** 1–7.
15. Abola, E. E., Bernstein, F. C., Bryant, S. H., Koetzle, T., and Weng, J. (1987) Protein data bank, in *Crystallographic Databases — Information, Content, Software Systems, Scientific Applications* (Allen, F. H., Bergerhoff, G., and Sievers, R., eds.), Data Commission of the International Union of Crystallography, Cambridge, pp. 107–132.

16. Chothia, C. (1992) One thousand families for the molecular biologist. *Nature* **360,** 543–544.
17. Holm, L. and Sander, C. (1996) Mapping the protein universe. *Science* **273,** 595–602.
18. Doolittle, R. F. (1990) Searching through sequence databases. *Methods Enzymol.* **183,** 99–110.
19. Altschul, S. F., Boguski, M. S., Gish, W., and Wootton, J. C. (1994) Issues in searching molecular sequence databases. *Nat. Genet.* **6,** 119–129.
20. Pearson, W. R. (1996) Effective protein sequence comparison. *Methods Enzymol.* **266,** 227–258.
21. Gribskov, M., McLachlan, A. D., and Eisenberg, D. (1987) Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. USA* **84,** 4355–4358.
22. Gribskov, M. (1994) Profile analysis. *Methods Mol. Biol.* **25,** 247–266.
23. Krogh, A., Brown, M., Mian, I. S., Sjolander, K., and Haussler, D. (1994) Hidden Markov Models in computational biology: applications to protein modeling. *J. Mol. Biol.* **235,** 1501–1531.
24. Eddy, S. R. (1996) Hidden Markov models. *Curr. Opin. Struct. Biol.* **6,** 361–365.
25. Bowie, J. U., Lüthy, R., and Eisenberg, D. (1991) A method to identify protein sequences that fold into a known three-dimensional structure. *Science* **253,** 164–170.
26. Jones, D. T., Taylor, W. R., and Thornton, J. M. (1992) A new approach to protein fold recognition. *Nature* **358,** 86–89.
27. Godzik, A., Kolinski, A., and Skolnick, J. (1992) Topology fingerprint approach to the inverse protein folding problem. *J. Mol. Biol.* **227,** 227–238.
28. Sippl, M. J. and Flöckner, H. (1996) Threading thrills and threats. Structure **4,** 15–19.
29. Torda, A. E. (1997) Perspectives in protein-fold recognition. *Curr. Opin. Struct. Biol.* **7,** 200–205.
30. Dunbrack Jr., R. L., Gerloff, D. L., Bower, M., Chen, X., Lichtarge, O., and Cohen, F. E. (1997) Meeting review: the second meeting on the critical assessment of techniques for protein structure prediction (CASP2), Asilomar, CA, December 13–16, 1996. *Fold. Des.* **2,** R27–R42.
31. Felsenstein, J. (1985) Confidence limits on phylogenies: an approach using the bootstrap. Evolution **39,** 783–791.
32. Browne, W. J., North, A. C. T., Phillips, D. C., Brew, K., Vanaman, T. C., and Hill, R. C. (1969) A possible three-dimensional structure of bovine α-lactalbumin based on that of hen's egg-white lysozyme. *J. Mol. Biol.* **42,** 65–86.
33. Greer, J. (1981) Comparative model-building of the mammalian serine proteases. *J. Mol. Biol.* **153,** 1027–1042.
34. Jones, T. H. and Thirup, S. (1986) Using known substructures in protein model building and crystallography. *EMBO J.* **5,** 819–822.
35. Unger, R., Harel, D., Wherland, S., and Sussman, J. L. (1989) A 3-D building blocks approach to analyzing and predicting structure of proteins. *Proteins* **5,** 355–373.
36. Claessens, M., Cutsem, E. V., Lasters, I., and Wodak, S. (1989) Modelling the polypeptide backbone with "spare parts" from known protein structures. *Protein Eng.* **4,** 335–345.

37. Levitt, M. (1992) Accurate modeling of protein conformation by automatic segment matching. *J. Mol. Biol.* **226,** 507–533.
38. Havel, T. F. and Snow, M. E. (1991) A new method for building protein conformations from sequence alignments with homologues of known structure. *J. Mol. Biol.* **217,** 1–7.
39. Srinivasan, S., March, C. J., and Sudarsanam, S. (1993) An automated method for modeling proteins on known templates using distance geometry. *Protein Sci.* **2,** 227–289.
40. Šali, A. and Blundell, T. L. (1993) Comparative protein modelling by satisfaction of spatial restraints. *J. Mol. Biol.* **234,** 779–815.
41. Brocklehurst, S. M. and Perham, R. N. (1993) Prediction of the three-dimensional structures of the biotinylated domain from yeast pyruvate carboxylase and of the lipolyated H-protein from the pea leaf glycine cleavage system: a new automated methods for the prediction of protein tertiary structure. *Protein Sci.* **2,** 626–639.
42. Aszodi, A. and Taylor, W. R. (1996) Homology modelling by distance geometry. *Fold. Des.* **1,** 325–334.
43. Šali, A. and Overington, J. (1994) Derivation of rules for comparative protein modeling from a database of protein structure alignments. *Protein Sci.* **3,** 1582–1596.
44. Šali, A. (1995) Protein modeling by satisfaction of spatial restraints. *Mol. Med. Today* **1,** 270–277.
45. Sánchez, R. and Šali, A. (1997) Comparative protein modeling as an optimization problem. *J. Mol. Struct. (Theochem.)* **398,** 489–496.
46. Lüthy, R., Bowie, J. U., and Eisenberg, D. (1992) Assessment of protein models with three-dimensional profiles. *Nature* **356,** 83–85.
47. Sippl, M. J. (1993) Recognition of errors in three-dimensional structures of proteins. *Proteins* **17,** 355–362.
48. Laskowski, R. A., McArthur, M. W., Moss, D. S., and Thornton, J. M. (1993) PROCHECK: a program to check the stereochemical quality of protein structures. *J. Appl. Cryst.* **26,** 283–291.
49. Hooft, R., Vriend, G., Sander, C., and Abola, E. (1996) Errors in protein structures. *Nature* **381,** 272.
50. Sánchez, R. and Šali, A. (1997) Evaluation of comparative protein structure modeling by MODELLER-3. *Proteins* **(Suppl.),** 50–58.
51. Guenther, B., Onrust, R., Šali, A., O'Donnell, M., and Kuriyan, J. (1997) Crystal structure of the δ' subunit of the clamp-loader complex of *E. coli* DNA polymerase III. *Cell* **91,** 335–345.
52. Šali, A., Sánchez, R., and Badretdinov, A. (1997) MODELLER, *A Protein Structure Modeling Program,* URL `http://guitar.rockefeller.edu./Modeller/Modeller.html`
53. Šali, A. and Blundell, T. L. (1994) Comparative protein modelling by satisfaction of spatial restraints, in *Protein Structure by Distance Analysis* (Bohr, H., and Brunak, S., eds.), IOS Press, Amsterdam, The Netherlands, pp. 64–86.
54. Šali, A., Potterton, L., Yuan, F., van Vlijmen, H., and Karplus, M. (1995) Evaluation of comparative protein modeling by MODELLER. *Proteins* **23,** 318–326.

55. Sánchez, R., Badretdinov, A. Y., Feyfant, E., and Šali, A. (1997) Homology protein structure modeling. *Trans. Am. Cryst. Assoc.* **32,** 81–91.
56. Xu, L. Z., Sánchez, R., Šali, A., and Heintz, N. (1996) Ligand specificity of brain lipid binding protein. *J. Biol. Chem.* **271,** 24711–24719.
57. Needleman, S. B. and Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48,** 443–453.
58. Sánchez, R. and Šali, A. Variable gap penalty function for protein sequence–structure alignments. In preparation.
59. Sayle, R. and Milner-White, E. J. (1995) RasMol: biomolecular graphics for all. *Trends Biochem. Sci.* **20,** 374.
60. Bolin, J. T., Filman, D. J., Matthews, D. A., Hamlin, R. C., and Kraut, J. (1982) Crystal structures of *Escherichia coli* and *Lactobacillus casei* dihydrofolate reductase refined at 1. 7 angstroms resolution. I. General features and binding of methotrexate. *J. Biol. Chem.* **257,** 13,650.
61. Johnson, M. S. and Overington, J. P. (1993) A structural basis for sequence comparisons: an evaluation of scoring methodologies. *J. Mol. Biol.* **233,** 716–738.
62. Barton, G. J. and Sternberg, M. J. E. (1987) A strategy for the rapid multiple alignment of protein sequences; confidence levels from tertiary structure comparisons. *J. Mol. Biol.* **198,** 327–337.
63. Sánchez, R. and Šali, A. (1998) Large-scale protein structure modeling of the saccharomyces cerevisiae genome. *Proc. Natl. Acad. Sci.* **95,** 13,597–13,602
64. Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990) Basic local alignment search tool. *J. Mol. Biol.* **215,** 403–410.
65. Pearson, W. R. (1990) Rapid and sensitive comparison with FASTA and FASTP. *Methods Enzymol.* **183,** 63–98.
66. Alexandrov, N. N., Nussinov, R., and Zimmer, R. M. (1995) Fast protein fold recognition via sequence to structure alignment and contact capacity potentials. World Scientific Publishing Co., Singapore, pp. 53–72
67. Rost, B. (1995) TOPITS: threading one-dimensional predictions into three-dimensional structures. AAAI Press, Menlo Park, CA, pp. 314–321.
68. Ficher, D. and Eisenberg, D. (1996) Fold recognition using sequence-derived predictions. *Prot. Sci.* **5,** 947–955.
69. Flockner, H., Braxenthaler, M., Lackner, P., Jaritz, M., Ortner, M., and Sippl, M. J. (1995) Progress in fold recognition. *Proteins* **23,** 376–386.
70. Sutcliffe, M. J., Haneef, I., Carney, D., and Blundell, T. L. (1987) Knowledge based modelling of homologous proteins, part I: three dimensional frameworks derived from the simultaneous superposition of multiple structures. *Protein Eng.* **1,** 377–384.
71. Bruccoleri, R. E. (1993) Application of systematic conformational search to protein modeling. *Mol. Sim.* **10,** 151–174.
72. Vriend, G. (1990) WHAT IF: a molecular modeling and drug design program. *J. Mol. Graph.* **8,** 52–56.
73. Peitsch, M. C. and Jongeneel, C. V. (1993) A 3-D model for the CD40 ligand predicts that it is a compact trimer similar to the tumor necrosis factors. *Int. Immunol.* **5,** 233–238.

74. Colovos, C. and Yeates, T. O. (1993) Verification of protein structures: patterns of non-bonded atomic interactions. *Protein Sci.* **2,** 1511–1519.
75. Kraulis, P. (1991) MOLSCRIPT: a program to produce both detailed and schematic plots of protein structure. *J. Appl. Cryst.* **24,** 946–950.

7

A Practical Guide to Protein Structure Prediction

David T. Jones

1. Introduction

The protein-folding problem is one of the greatest remaining challenges in structural molecular biology (if not the whole of biology). How do proteins translate from their primary structure (sequence) to tertiary structure? How is the information encoded? Basically, how do proteins fold? Often, the protein-folding problem is seen as a computational problem — do we know enough about the rules of protein structure to program a computer to read in a protein sequence and output a correct tertiary structure? Aside from the academic interest in understanding the physics and chemistry of protein folding, why are so many people interested in finding an algorithm (i.e., a method) for predicting the native structure of a protein given just its sequence?

The ideal way to derive structural information for a given protein is to determine the structure by X-ray crystallography or nuclear magnetic resonance (NMR) spectroscopy. There are a number of problems:

1. Some proteins cannot be (easily) crystallized for one reason or another.
2. Crystallography can take anywhere from several months to several years to determine the structure of a single protein.
3. NMR is, on average, quicker than crystallography but cannot currently be applied to proteins larger than about 100 residues.
4. There are currently around 100,000 protein sequences known, but only 2000 or so X-ray structures have been determined to date.
5. As a result of genome projects, the number of known protein sequences is likely to reach 500,000 by the year 2000.
6. There is unlikely to be a significant increase in the rate of structure determination in this time frame.
7. Assuming that it was possible to solve the structure of all these unsolved proteins, it would take approx 500 yr to achieve this goal.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

Protein–structure prediction is, therefore, going to be vital to bridge the gap between structure and sequence determination. Membrane-bound proteins are a particular problem — there are only five or so known structures for membrane-bound proteins and yet sequences are known for tens of thousands of membrane-bound proteins.

Modeling protein structure also allows the planning of protein design and modification experiments. Molecular modeling has been used to enhance the activity of enzymes and to design potential new therapeutic agents.

2. Strategies for Protein Structure Prediction

A number of different strategies are available to compute a structure for a protein sequence (e.g., **ref. *1***). These can be classified into three broad categories: comparative modeling, fold recognition, and methods.

3. Comparative Modeling

At present, the modeling of unknown protein structures by homology represents the best known method for protein–structure prediction. The modeling process consists of six basic steps: aligning the target sequence on the backbone of the parent structure, building a framework structure, adding and optimizing side chains, building loops, refining the model, and validating (including estimates of reliability).

3.1. Aligning of the Target Sequence on the Backbone of the Parent Structures

This step is by far the most important. None of the later steps can compensate for errors made in the initial alignment of the target protein with the parent. Where the degree of sequence similarity is below 40% sequence identity, standard alignment methods can produce structurally incorrect results for one or more regions ***(2)***. Fortunately, manual adjusting automatically generated alignments can correct these mistakes.

3.2. Building a Structural Framework

Once a sequence–structure alignment has been produced, a decision must be made as to which parts of which parent structures are similar enough in conformation to the target protein for use in the initial model. Models can be built from a single parent structure, but significant improvement can be gained from the used of related structures. The difficulty in using multiple structures is to decide which parent to use for each part of the target protein, and where to assume conformational similarity breaks down. Sometimes, structures with less overall sequence similarity offer the best choice for segments of structures, and local sequence similarity can provide a rough indicator of that situation

(3). Generally, sequence similarity is quite obvious for stretches of the alignment, usually approximately corresponding to regions of secondary-structure, and then becomes unreliable in loop regions.

3.3. Constructing of Core Side Chains

A number of algorithms have been published that accurately build back side chains with high accuracy given a main-chain conformation ***(4)***. When applying these algorithms to real modeling problems, however, it was found that the side chains are often built with less accuracy than expected, even for core residues. The explanation for this appears to be that the main-chain conformation is generally not modeled accurately enough to allow the side-chain conformations to be predicted accurately ***(5)***.

3.4. Building the Loops

After the core is complete, short regions of chain usually remain to be built (typically loop regions). A number of algorithms have been published that appear to be able to accurately build loops (e.g., **refs. *6*** and ***7***), but in practice, loop building is still a difficult part of the modeling process, especially for long loops.

3.5. Refining the Models

Once an initial model has been built, the next question is whether the structure can be improved. Most groups attempt to refine the starting model by either simple energy minimization or molecular dynamics, but there is little evidence that the overall similarity between the model and the true native structure is increased by any of these approaches.

3.6. Estimating the Reliability of Models

A number of simple checks can be made to the final model structure to assess its quality. Two commonly used programs are PROCHECK ***(8)*** and WHATCHECK ***(9)***. These programs can, e.g., check the stereochemical quality of the structures, and can check for unfavorable side chain environments (usually a good indicator of incorrectly folded protein structures), but it is all to common to find poor models scoring very well when tested with these programs.

3.7. Recent Developments

There have been very few significant developments in the actual process of comparative modeling in recent years, but rather than taking this as a sign of lack of progress, this indicates a significant maturity in the modeling field in that people are concentrating on the use of existing tools, rather than the unnecessary development of new ones.

One of the most impressive recent developments in the modeling field is the wide availability of reliable automatic comparative modeling programs. In particular the SWISS-MODEL server ***(10)*** has been shown to produce excellent results in comparison to the results from much more elaborate protocols when the degree of sequence similarity between parent and target is relatively high. For more remotely related target-parent pairs, SWISS-MODEL fares less well due to the fact that it uses automatic sequence alignments. Note that SWISS-MODEL does allow manually edited alignments to be submitted rather than just a single sequence, and it is to be expected that given the same initial alignments SWISS-MODEL would produce results comparable to those obtained by experts in comparative modeling. The success of SWISS-MODEL is believed to be a very important progression in the modeling field. Although it perhaps marks the hammering-in of a single nail into the coffin of "professional" comparative modeling as a trade, it does mean that nonspecialists (academics at least — SWISS-MODEL is not an option for commercial researchers) can now produce excellent models without recourse to an expert, or the purchase of expensive software. Of course this high degree of automation in the basic process of comparative modeling means that the professionals can now concentrate on the much more important aspects of the modeling process, i.e., validation (checking the model) and evaluation (actually gaining biological insight from it).

4. Fold Recognition

It has long been recognized that proteins often adopt similar folds despite there being no significant sequence or functional similarity. It has also been estimated that for 50–70% of new proteins there will be a suitable structure in the database from which to build a 3D model. Unfortunately, due to insignificant sequence similarity, many of these go undetected until after 3D structure of the new protein is solved. Over recent years methods have been developed that attempt to recognize these fold similarities for pairs of proteins with very low sequence similarity. These methods are known as fold-recognition methods or more commonly as threading methods (*see* **Fig. 1**).

The term "fold recognition" covers two variations on the same theme. The problem of finding one or more sequences that are compatible with a fold is commonly called inverse protein folding. In contrast to this, perhaps a more useful formulation of fold recognition is detecting of a matching fold given a sequence. Simply put, if you have a single structure and are trying to find either one or more sequences that might fold into this structure, then the method is inverse protein folding. On the other hand, if you have a single sequence and are interested in finding a structure (from a library of known structures) that is most likely to resemble the native conformation of the sequence, then the method is protein folding (by means of fold recognition). There is, of course,

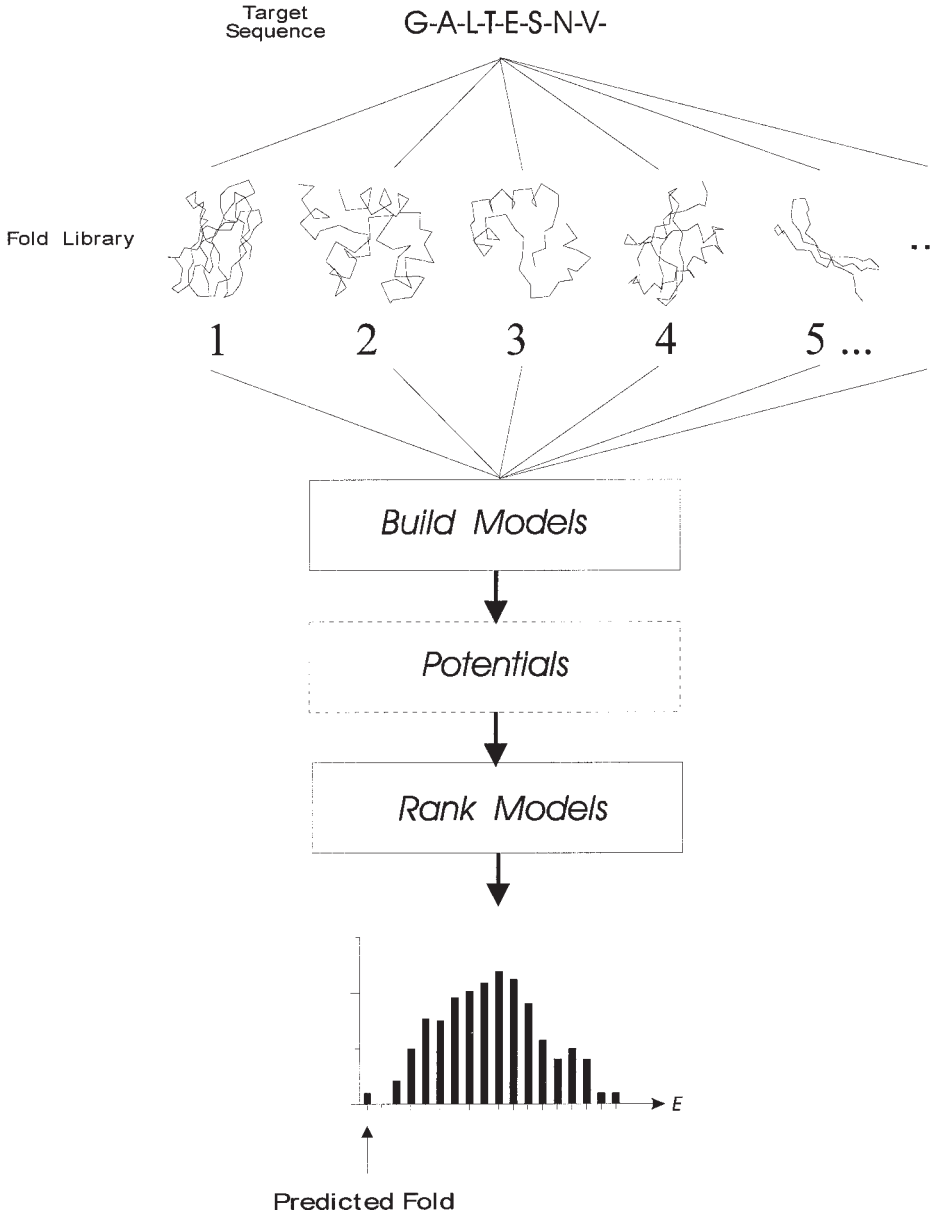


Fig. 1. A conceptual outline of fold recognition as a solution to the protein-folding problem. A given sequence (target) is fitted to the backbones of known structures (fold library), and the goodness-of-fit in each case is evaluated by one of many available model evaluation procedures (potentials).

some overlap between inverse protein folding and threading. In particular, both methods employ some measure of sequence–structure comparability and some means for performing the sequence–structure alignment. These measures are either based on 1D sequence properties, such as solvent accessibility ***(11–13)***, or incorporate interresidue pairwise interactions ***(14–21)***. The substitution matrices described by Overington et al. ***(22)*** represents an alternative approach to measuring sequence–structure compatibility for inverse protein folding.

5. Ab Initio Prediction

In many cases, neither comparative modeling nor threading can provide a useful model for a sequence under study. At present, there is roughly a 50% chance of finding a protein fold in the protein structure databank, which is significantly similar to a newly solved protein domain (C. A. Orengo, personal communication), but of course this chance will increase steadily as more structures are solved. The real problem in prediction is to know when a suitable structure is present in the databank. Although it is readily apparent when comparative modeling is not viable (e.g., because of lack of significant sequence similarity to a template protein), knowing when a threading prediction is wrong is somewhat harder.

By far the most widely applied *ab initio* prediction methods are those relating to the prediction of secondary-structure. For many years it seemed that 60% accuracy was the absolute limit for secondary-structure prediction methods, but the availability of large families of homologous sequences has recently revolutionized secondary-structure prediction. Traditional methods such as GOR ***(23)*** or Chou and Fasman ***(24)***, when applied to a family of proteins rather than a single sequence, proved to be much more accurate at identifying core secondary-structure elements.

There have been many reviews on secondary-structure prediction (e.g., **ref. *25***), and there has been very little significant progress in secondary-structure prediction in recent years. This is not as bad as it sounds, however, as modern methods have already achieved a useful level of accuracy by taking into account evolutionary information extracted from multiply aligned protein sequences ***(26)***. The PHD method of Rost and Sander ***(26)*** is probably the most widely used method today, and the results presented at the CASP2 meeting (***[27]***;and *see* later) indicate that PHD is still the method to beat. In the hands of the authors, PHD manages an average Q_3 score (i.e., accuracy) of around 73%. In other words, 73% of the residues in a protein are expected to be classified correctly into three secondary structural types (helix, sheet, or coil). However, it must be realized that this is an average performance when tested over many proteins. An individual protein may have a much lower or much higher Q_3 score than this average.

Despite the success of the PHD method, other methods are certainly catching up. Of these, an interesting new contender is the DSC method ***(28)***. What makes this method particularly interesting is that it is relatively simple to implement, and indeed the authors have made the program code available to other researchers. This is a very important difference from PHD, which remains accessible solely via e-mail or the World Wide Web.

Given that secondary-structure prediction methods can often produce quite accurate predictions in cases where many related sequences are available for analysis, it has been natural to ask whether it is possible to assemble correct folds for proteins based on these predicted elements of secondary-structure and some means for predicting how these elements pack together. This, of course, leads naturally to protein-folding simulation, which will be discussed shortly, but an interesting intermediate method comes from the possibility that contacts in a protein structure can be predicted , by analyzing multiple sequence alignments and looking for correlated mutations. Several studies have been made on this aspect of protein structure prediction ***(28–32)*** with some differences of opinion evident in the conclusions. Although it is certainly possible to predict specific contacts in protein structures from multiply aligned sequences, it is difficult to use this information due to the relatively large numbers of false positives that are output. It is also fair to say that a very large number of related sequences (i.e., more than 30) is required to make any attempt at such contact prediction at all.

6. *Ab Initio* Prediction

Ab initio prediction of protein tertiary structure is, of course, the "holy grail" of the prediction field. However, there is little evidence that significant progress toward this goal has been made to date.

In general, methods for *ab initio* tertiary structure prediction employ some means for generating different protein-chain conformations and a potential function with which to evaluate each conformation. Unsurprisingly, there is a lot of overlap between such *ab initio* methods and threading methods, and many potentials used for threading can be used for folding simulations. For folding simulations, however, other terms often need to be added to take into account steric hindrance, hydrogen bonding, and general chain compactness. These terms are not necessary for threading, as the conformations of the structures onto which the target sequence is being threaded will satisfy all these requirements.

Some *ab initio* methods diverge very little from the basic recipe described and attempt to minimize a given potential function using some simplified representation of a polypeptide chain. Conformations of this chain can be restricted to points on a lattice (e.g., **refs. *33–43***) or restricted by choosing discrete main

chain torsion angles (e.g., **refs.** ***44–47***). Generally speaking, some kind of Monte Carlo optimization is used, either based on some variant of "simulated annealing" or more recently based on a genetic algorithm ***(48)***.

In some cases, external information is used to bias the simulation toward particular regions of conformational space. The most common bias that is often applied is to use predicted, or even experimentally derived, secondary structural information. Some recent work has also looked at the possibility of obtaining reasonable chain folds based on a small number of distance constraints ***(49–50)***, and this kind of approach may be useful not only for structure prediction but also for assisting in the determination of NMR structures where only limited data are available.

7. Structure Prediction

This section gives some practical advice on predicting the structure for a newly determined protein sequence. Before starting out, it is important to be very clear about what level of detail is required from the prediction. If it is sufficient to produce a crude topological model for the protein under study, then there is a much greater chance of success than if an accurate all-atom model is required. It is important to have specific questions in mind when attempting structure prediction.

The main rule to follow is to use the simplest and most reliable method that can do the job. If your protein has a high degree of sequence similarity to a protein of known structure, then build a model using standard comparative modeling techniques and leave it alone. If you are able to build a good comparative model of your protein, do not bother with more complicated and less reliable methods such as secondary-structure prediction, threading, or *ab initio* folding simulations. In rare circumstances there might be some cause to enhance a model using a more advanced method.

Following is an outline of a recommended procedure for attempting to predict the structure of a newly characterized protein sequence. The emphasis here is to encourage the use of multiple methods rather than relying on a single technique.

7.1. STAGE 1: Sequence Searching

Before moving onto more advanced methods for prediction, it is advisable to expend effort in detecting homology between the target protein and proteins of known 3D structure. This seems like an obvious thing to do, but many people skimp on this step and end up leading themselves astray.

There are a number of sequence comparison software packages that can be obtained freely from the authors. Two of these packages are recommended: FASTA3 ***(51)*** and PSI-BLAST ***(52)***. The FASTA3 package includes not only FASTA, which is a fast and sensitive sequence-comparison method in its own

right, but also a good implementation of the widely used Smith-Waterman algorithm (SSEARCH) ***(53)***. PSI-BLAST is the latest incarnation of the original BLAST package ***(56)***, and is now capable of generating gapped alignments. Of even more interest are the new Position Specific Iteration (PSI) Features of PSI-BLAST. Here, rather than searching a databank of sequences with a single sequence and then finishing, the program builds a profile based on the initial search results and uses this profile to search the databank again. If the profile pulls any more sequences out of the databank, then these new sequences are added to the profile and the new profile used again to search for more sequences. This procedure can be terminated after a fixed number of iterations, or can be allowed to continue until no more sequences are detected (convergence).

One major improvement that has occurred recently with the foregoing sequence searching programs is that they now output useful measures of statistical significance. Generally, the programs produce an "*E*-value," which corresponds to the expected number of hits that would be expected to achieve an equivalent search score purely by chance.

To use FASTA or PSI-BLAST for structure prediction it is necessary to create a databank of sequences for which the 3D structure is known. In other words, it is necessary to extract sequence information from Brookhaven PDB formatted files ***(57)*** and convert these data into a flat-file sequence format (usually the FASTA file format). Fortunately, already-converted sequence files are readily available via the World Wide Web or FTP (*see* resources) — e.g., the NRL3D data bank.

Let us look at an example for both FASTA3 and PSI-BLAST, starting with FASTA3. In this case we take, as an example, one of the proteins that were prediction targets in the CASP2 experiment ***(27)***. The sequence in question was as follows:

```
>T0004 Polyribonucleotide Nucleotidyltransferase, S1 motif
AEIEVGRVYTGKVTRIVDFGAFVAIGGGKEGLVHISQIADKRVEKVTDYLQM
GQEVPVKVLEVDRQGRIRLSIKEATEQSQPAA
```

FASTA3 was used to search the sequences extracted from the Brookhaven PDB with ktup = 1 (which is the most sensitive setting for FASTA3). After a few seconds, the following results were produced:

```
The best scores are:                                  initn init1 opt z-sc E(3576)
1CSP TRANSCRIPTION REGULATION              ( 67)     43    43     64    106.3   1.5
1DPRA TRANSCRIPTION REGULATION             ( 226)    64    64     67    100.2   3.2
1TDX DNA BINDING REGULATORY PROTEIN        ( 226)    64    64     67    100.2   3.2
1GPR PHOSPHOTRANSFERASE                    ( 162)    40    40     65    100.1   3.3
1NEF REGULATORY FACTOR                     ( 136)    61    61     61     95.5   5.8
1PREA TOXIN (HEMOLYTIC POLYPEPTIDE)        ( 470)    46    46     68     95.3   6
2FCR ELECTRON TRANSPORT                    ( 173)    38    38     62     95.0   6.3
.
.
```

The first thing to look at here is the value in the *E*-value column. In this case there are 3576 sequences in the databank (i.e., there are 3576 unique sequences with known 3D structure). The best-scoring match between the target protein produces an *E*-value of 1.5, which indicates that we would expect a similar score ***(64)*** to occur by chance about 1.5 times on average. This is clearly not a significant result. A 99% confidence would require us to only accept matches with *E*-values of 0.01 or less. This is certainly good advice, but what is surprising in this case is that the correct match has in fact been found. The structure of the protein represented by PDB entry 1CSP (Cold Shock Protein) is indeed an excellent model for the structure of target T0004.

Should you always believe the top scoring match in a FASTA3 search? Of course not. Without any additional information there would be no reason at all to believe that 1CSP has any relationship to the target protein under study. However, when the biological relevance of the top hit is considered, then a match to the target protein looks more plausible. Nevertheless, it is quite risky to ignore the statistical significance of the search results. However, spotting a useful match at this stage can be a very useful shortcut to the prediction process.

Assuming that no credible match can be found using FASTA, or the SSEARCH program included alongside FASTA (which is much slower than FASTA but slightly more sensitive), then more sensitive sequence-comparison methods can be tried. In the past, the advice here would have been to use sequence profiles ***(54)*** or Hidden Markov Models ***(55)*** to look for very remote sequence similarities. But then the PSI-BLAST program was made available ***(52)***, which permits very sensitive sequence searches to be carried out with little more effort than a normal BLAST search.

To use PSI-BLAST to maximum effect, a hybrid sequence databank is required, which contains the same Brookhaven PDB sequences described, but also includes other protein sequences taken from a standard protein sequence databank such as TREMBL, SWISS-PROT, or OWL. In this example, the databank includes the same 3576 sequences as used with FASTA3, but also includes 244,827 sequences taken from SWISS-PROT, TREMBL, and OWL. Why add these 244,827 additional sequences? Recall that PSI-BLAST works by constructing sequence profiles as it runs. Sequence profiles are generally improved by including as many diverse sequences as possible, and so these additional sequences can greatly extend the range of PSI-BLAST in detecting homology between the target sequence and a PDB sequence.

The target sequence in this case is again taken from the CASP2 experiment:

```
>T0031; Exfoliative toxin A from Staphylococcus aureus, 242a.a.
EVSAEEIKKHEEKWNKYYGVNAFNLPKELFSKVDEKDRQKYPYNTIGNVFVKGQTSATGVLIGKNTVLT
NRHIAKFANGDPSKVSFRPSINTDDNGNTETPYGEYEVKEILQEPFGAGVDLALIRLKPDQNGVSLGDK
ISPAKIGTSNDLKDGDKLELIGYPFDHKVNQMHRSEIELTTLSRGLRYYGFTVPGNSGSGIFNSNGELV
GIHSSKVSHLDREHQINYGVGIGNYVKRIINEKNE
```

PSI-BLAST is run on this sequence using the following parameters: -h 0.001 -j 10, which allows PSI BLAST to make as many as 10 iterations, but with new sequences being added at each iteration only where the sequences match the current profile with an *E*-value of 0.001 or less.

The first iteration from PSI-BLAST produces the following matches:

```
ETA_STAAU EX8FOLIATIVE TOXIN A PRECURSOR (EC 3.4.21.-) (EPIDERMO...          499     e-141
ETB_STAAU EXFOLIATIVE TOXIN B PRECURSOR (EC 3.4.21.-) (EPIDERMO... 197     4e-50
S21758 glutamic acid-specific endopeptidase - Staphylococcus au...            94     5e-19
STSP_STAAU GLUTAMYL ENDOPEPTIDASE PRECURSOR (EC 3.4.21.19) (STA...            93     1e-18
SAU60589 SAU60589 NID: g1407783 - Staphylococcus aureus.                      84     7e-16
SAU63529 SAU63529 NID: g1488694 - Staphylococcus aureus.                      72     2e-12
S25140 serine proteinase homolog - Enterococcus faecalis                      65     3e-10
C64647 serine proteinase (EC 3.4.21.-) - Helicobacter pylori (s...            50     1e-05
D78376 D78376 NID: g1526427 - Yersinia enterocolitica (strain:W...            48     4e-05
YEHTRA YEHTRA NID: g1419350 - Yersinia enterocolitica.                        48     4e-05
CJHTRA CJHTRA NID: g2077988 - Campylobacter jejuni.                           46     1e-04
CJU27271 CJU27271 NID: g881374 - Campylobacter jejuni.                        46     1e-04
E1181491 YKDA. 45 5e-04
DEGQ_ECOLI PROTEASE DEGQ PRECURSOR (EC 3.4.21.-).                             43     0.001
```

These hits are all significant, but at this stage none of the hits are to any of the sequences taken from PDB, i.e., sequences with known 3D structure. However, on the third iteration, the following output is produced:

```
.
.
POLG_PPVNA GENOME POLYPROTEIN (CONTAINS: N-TERMINAL PROTEIN; HE...            53     1e-06
PSU05771 PSU05771 NID: g1335723 - Peanut stripe virus.                        53     1e-06
PSU34972 PSU34972 NID: g1016234 - Peanut stripe virus.                        53     1e-06
YNM3_YEAST HYPOTHETICAL 110.9 KD PROTEIN IN SPC98-TOM70 INTERGE...            53     1e-06
PVCHYMOA PVCHYMOA NID: g2462646 - Penaeus vannamei.                           52     3e-06
PVCHYMOB PVCHYMOB NID: g2462648 - Penaeus vannamei.                           51     4e-06
pdb|1TRY|1TRY trypsin                                                         51   7e-06 fl-
CTR2_VESOR CHYMOTRYPSIN II (EC 3.4.21.1).                                     49     2e-05
STMSAMP20 STMSAMP20 NID: g474021 - Streptomyces albogriseolus (...            49     2e-05
BCU19287 BCU19287 NID: g625062 - Bean common mosaic virus.                    49     2e-05
YMU425966 YMU42596 NID: g1552411 - Yam mosaic virus.                          49     2e-05
.
.
```

Although quite a long way down the list of hits, a statistically significant match is reported between the target protein and PDB entry 1TRY (*E*-value = 7×10^{-6}). In subsequent iterations, the *E*-value for this match drops to as little as 10^{-20}, which is clearly significant.

In the event of a similarity being detected at this stage, it is advisable to stop here and consider the construction of an homology model. This is not as easy as it sounds. Unless the similarity is very obvious, it is will not be trivial to

produce a correct alignment between the target protein and the protein of known structure *(2)*.

7.2. Stage 2: Secondary-Structure Prediction

Assuming that no useful matches were found in the first stage, it is necessary to consider using secondary-structure prediction methods. Although secondary-structure information alone is generally of only limited use, it is nonetheless helpful to be able to refer to a reliable secondary-structure prediction when attempting to predict the tertiary structure by fold recognition. The following structural clues can sometimes be obtained through inspection of predicted secondary structural elements:

1. The structural class of the target protein may be ascertained (all-α, all-β, or α–β).
2. Structural repeats may be detected. By identifying a repeating sequence of secondary-structures, it is sometimes possible to identify repeated domains in a the target protein.
3. The sequence of secondary structural elements can be compared to the folds matched by fold recognition. For fold-recognition methods, which do not use predicted secondary structure, this "second opinion" is of great value in determining the degree of confidence to assign to the prediction.

It is worth taking time at this stage to get the best prediction of secondary structure possible. At present, PHD *(26)* is the method of choice, but results from PHD are highly dependent on the number of sequences in the aligned family and on the quality of the alignment. As a rule, better predictions are obtained from PHD by submitting a hand-crafted multiple sequence alignment to the server, rather than relying on the server to make the alignment. In this way, it is possible to search many different sequence databanks to find extra related sequences before submitting the alignment to the server.

7.3. Stage 3: Fold Recognition

The following section outlines the practical application of a widely used threading program, THREADER2 *(15,58–59)*. This is, of course, not the only threading method, but it is one of the few programs to be made widely available to the academic community. Although the specifics of this section relates solely to THREADER2, the principles can be applied to almost any threading program. The THREADER2 program can be obtained from the following Web address: `http://globin.bio.warwick.ac.uk/~jones/threader.html`

THREADER2 can be used either via a UNIX command-line interface, or as shown in **Fig. 2**, using a graphical user interface. The examples here assume that you are using THREADER2 directly from the command line.

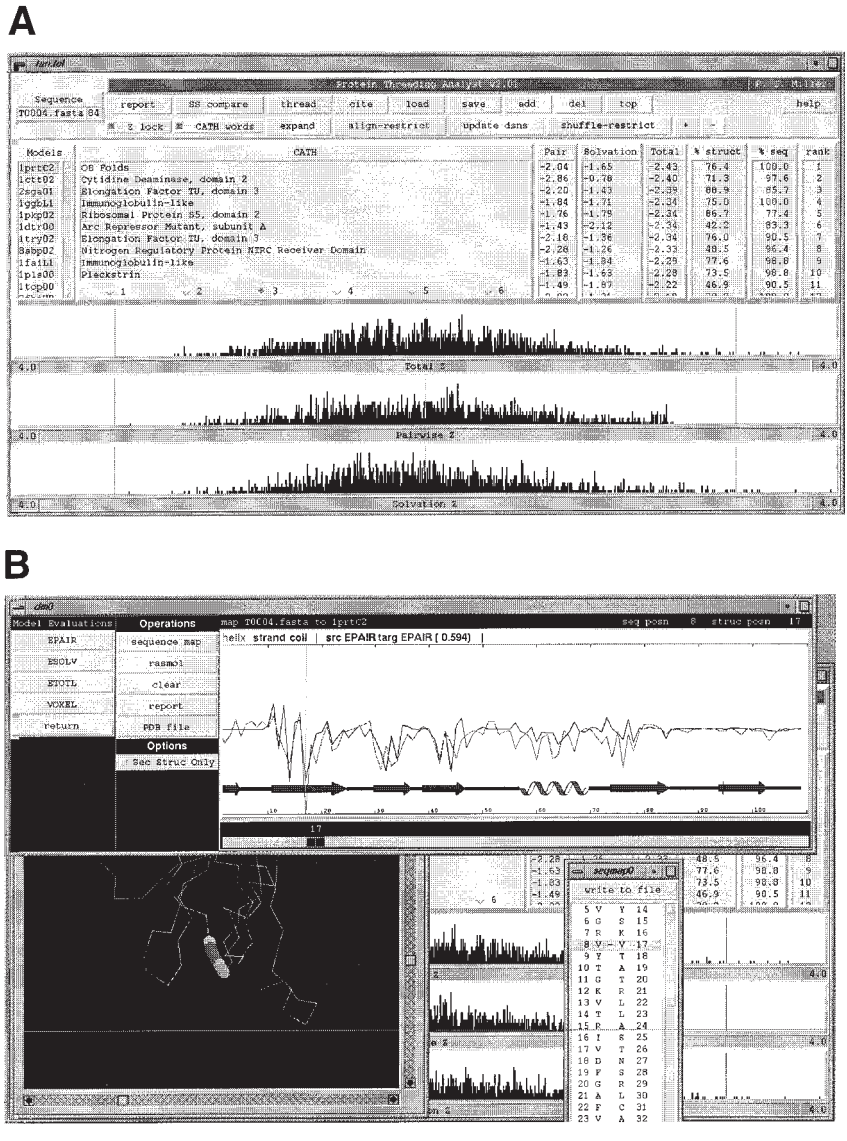


Fig. 2. THREADER2 plus Threading Analyst in use. The images are screenshots of THREADER2 being used within a graphical user interface (GUI) developed by Rob Miller *(58)*. **(A)** The initial screen shows the distribution of threading scores, along with a brief description of the type of fold for each match in the fold library. **(B)** An analysis of the first model in progress. The alignment is being examined in the context of the 3D structure. The two plots shown in the upper window are of the threading energy profiles for both the template and target protein. Where the two profiles are correlated it is possible to be more confident about the accuracy of the model.

7.3.1. Starting Out

As an example, we take the following sequence:

```
>GUN1_STRRE Endo-1,4-beta-glucanase
VEQVRNGTFDTTTDPWWTSNVTAGLSDGRLCADVPGGTTNRWDSAIGQNDITLVKGETYR
FSFHASGIPEGHVVRAVVGLAVSPYDTWQEASPVLTEADGSYSYTFTAPVDTTQGQVAFQ
VGGSTDAWRFCVDDVSLLGGVP
```

This sequence has no apparent sequence similarity to any other sequence with a known structure when tested using FASTA3 and PSI-BLAST.

THREADER2 generates a lot of output — usually in the form of a rather unfriendly table of numbers. However, only a few of them are really important in most cases. This is really unavoidable, as different scoring schemes each tell a different side of the story. Matches that score very well in terms of pairwise energy, but very badly in terms of solvation energy, may be telling something about the multimeric state of your protein sequence. Fitting a hemoglobin sequence onto a myoglobin template will give a good pairwise energy score, but a poor solvation score because hemoglobin is a tetrameric structure and myoglobin monomeric — the solvent accessibilities for myoglobin will not, therefore, be an ideal match for those of hemoglobin.

After running THREADER2 on the example sequence, ranking the results by column 13, which is the Z-score for the combined pairwise-solvation energy, gives the following top-10 matches:

```
-191.9488   3.94  4.53  1.55  2.50  2.50  -5.0186  1.73  1.73  -272.1923  2.55  -272.1923  2.55     85.1
67.8          01igcH1
-169.9143  -2.33  3.55  1.06  2.14  2.14  -5.1257  1.76  1.76  -251.8700  2.35  -251.8700  2.35     93.7
68.4          01hplA2
-133.8000   1.52  3.45  1.35  1.54  1.54  -6.2484  2.05  2.05  -233.7066  2.17  -233.7066  2.17     94.2
64.5          03rp2A2
-108.0762  -2.63  2.59  0.10  1.12  1.12  -7.7790  2.46  2.46  -232.4569  2.15  -232.4569  2.15     87.0
78.9          02aaiB1
-156.1925  -2.46  3.47  0.89  1.91  1.91  -4.5850  1.62  1.62  -229.5032  2.13  -229.5032  2.13     86.1
89.5          01dlc02
-116.9902  -0.79  1.49 -1.00  1.27  1.27  -6.1849  2.04  2.04  -215.8823  1.99  -215.8823  1.99     87.5
69.1          01brbE1
-149.4035   1.33  3.57  0.87  1.80  1.80  -4.0816  1.48  1.48  -214.6647  1.98  -214.6647  1.98     84.0
82.9          01f3g00
-131.2358  -0.33  2.74  0.57  1.50  1.50  -4.8574  1.69  1.69  -208.9015  1.92  -208.9015  1.92     94.5
78.9          01eta10
-116.8615  -4.12  2.14 -0.58  1.26  1.26  -4.8859  1.70  1.70  -194.9833  1.78  -194.9833  1.78     97.5
77.6          01sriA0
-110.9676  -0.88  1.65 -1.06  1.17  1.17  -4.9621  1.72  1.72  -190.3084  1.74  -190.3084  1.74     89.5
90.1          01sdyA0
              .    .
              .    .
```

This is the normal result of an initial threading run for a difficult target sequence. In this case, one very close structural match for the target sequence is included in the top 10 matches, but not in first place. Also, in this case, the

accuracy of the sequence–structure alignment is also remarkably good. The task in this case is to identify which of these matches are false positives, and which is really the best matching fold. Several things in the initial results indicate that further processing is required to make a more reliable prediction.

The first thing to notice about these results is that the combined *Z*-scores in column 13 are rather low. The highest scoring fold only produces a *Z*-score of 2.55, and this is not a significant score. In tests, the following interpretation of this *Z*-score appears valid:

$Z > 3.5$	Very significant — probably a correct prediction
$Z > 3.0$	Significant — good chance of being correct
$2.7 < Z < 3.0$	Borderline significant — possibly correct
$2.0 < Z < 2.7$	Poor score — could be right, but needs other confirmation
$Z < 2.0$	Very poor score — probably there are no suitable folds in the library

Note that these *Z*-score ranges depend on the number of folds in the fold library. If you use a much smaller library than the default (which has over 700 folds), the *Z*-scores will not be useful measures of significance. If you use a larger library, the significance cutoffs should be increased.

More evidence against the top ranked fold can be seen in columns 2–4:

```
-191.9488  3.94 4.53  1.55  2.50  2.50  -5.0186  1.73  1.73 -272.1923  2.55 -272.1923  2.55     85.1
67.8          01igcH1
-169.9143 -2.33 3.55  1.06  2.14  2.14  -5.1257  1.76  1.76 -251.8700  2.35 -251.8700  2.35     93.7
68.4          01hplA2
-133.8000  1.52 3.45  1.35  1.54  1.54  -6.2484  2.05  2.05 -233.7066  2.17 -233.7066  2.17     94.2
64.5          03rp2A2
-108.0762 -2.63 2.59  0.10  1.12  1.12  -7.7790  2.46  2.46 -232.4569  2.15 -232.4569  2.15     87.0
78.9          02aaiB1
-156.1925 -2.46 3.47  0.89  1.91  1.91  -4.5850  1.62  1.62 -229.5032  2.13 -229.5032  2.13     86.1
89.5          01dlc02
-116.9902 -0.79 1.49 -1.00  1.27  1.27  -6.1849  2.04  2.04 -215.8823  1.99 -215.8823  1.99     87.5
69.1          01brbE1
-149.4035  1.33 3.57  0.87  1.80  1.80  -4.0816  1.48  1.48 -214.6647  1.98 -214.6647  1.98     84.0
82.9          01f3g00
-131.2358 -0.33 2.74  0.57  1.50  1.50  -4.8574  1.69  1.69 -208.9015  1.92 -208.9015  1.92     94.5
78.9          01eta10
-116.8615 -4.12 2.14 -0.58  1.26  1.26  -4.8859  1.70  1.70 -194.9833  1.78 -194.9833  1.78     97.5
77.6          01sriA0
-110.9676 -0.88 1.65 -1.06  1.17  1.17  -4.9621  1.72  1.72 -190.3084  1.74 -190.3084  1.74     89.5
90.1          01sdyA0
```

The scores in columns 2–4 give the core-shuffled scores for the relevant match. This simple shuffling procedure is not as effective as performing a full shuffling test, but can identify false positives with very little additional computing time. For the first match, the scores in columns 3 and 4 are actually quite acceptable in that column 3 is >2.7 and column 4 is >0. Unfortunately, the score in column 2 is >>0, which indicates that the threaded model has a lower energy than the

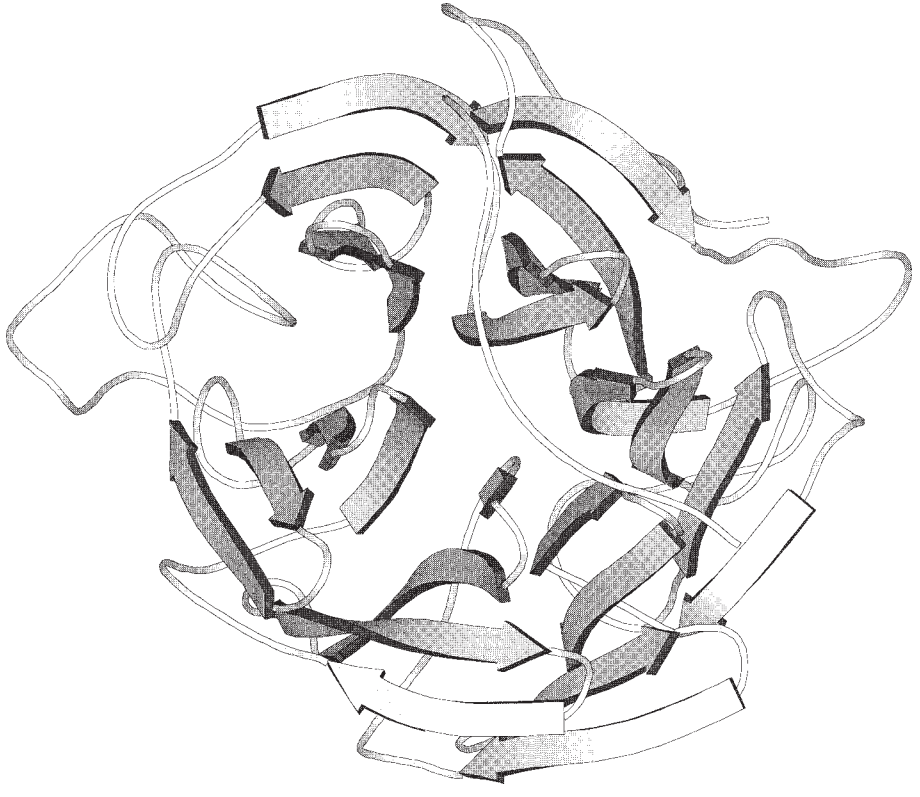


Fig. 3. MOLSCRIPT ***(61)*** diagram of influenza virus neuraminidase ***(63)***, which has a β-propellor fold. This fold has an unusually large number of interresidue contacts for its size, and this seems to allow it to accommodate sequences with a high proportion of hydrophobic residues. This is a possible explanation why this fold often appears as a false positive match in fold-recognition experiments.

native structure itself, which usually indicates a false–positive match. One does not expect the energy for an approximate model to be better than that for a sequence threaded onto its native structure. Using these criteria, only 4 of the top 10 matches are really acceptable.

Of these 4 matches, it is often useful to concentrate on matches which align the largest proportion of the target sequence, and on this basis the match to 1dlc02 would be the preferred match. Despite this, however, more confidence can be gained by running a set of more rigorous randomization tests. To save computer time it is sensible to limit the rigorous randomization tests to only those matches that looked promising from the initial results, but if time is not

an issue then extensive randomizations can be performed on the whole fold library. In this example, the top 20 matches from the initial threading pass are reevaluated by sequence shuffling.

7.3.2. Why Shuffle the Target Sequence?

Most threading potentials include terms that strongly favor contacts between hydrophobic residues. An ideal threading model when evaluated with this kind of potential function is one where as many contacts between hydrophobic residues are made as possible. For most target sequences and folds this is not a problem, but in some cases a threading program can find an alignment between a particularly hydrophobic sequence and a structure that produces a threading score far better than that of the native protein sequence fitted onto its own structure. These "superstable" threadings are a prime cause of false positive matches in fold recognition. One example of a protein fold that causes particular problems is the β-propeller (*see* **Fig. 3**). This fold is unusual in that it has a large hydrophobic core and many possible sites that can allow pairs of hydrophobic to make contacts. The implication of this for fold recognition is that a target sequence with a high proportion of hydrophobic residues will find the β-propeller fold a particularly stable arrangement.

8. Combining Predicted Secondary-Structure with Threading

To provide a final clue to the correct fold, it is often worthwhile considering the predicted secondary-structure for the target protein. For example, the following sequence was analyzed using the PHD program and then threaded through a library of folds using THREADER2:

```
>T0026; ArgR N-term domain from E.coli, 79 a.a.
MRSSAKQEELVKAFKALLKEEKFSSQGEIVAALQEQGFDNINQSKVSRMLTKFGAVRTRNAKMEMVYCL
PAELGVPTTS
```

Analysis of the threading results for this sequence suggests two possible folds. Both folds are somewhat similar to the native fold of the target protein, but one is clearly a better match when the predicted secondary-structure is used to annotate the sequence–structure alignment.

1.

```
10        20        30        40        50
----CCCCHHHHHHHHHHCCCCCCCEEHHHHHHHHHHHHCCCCCCHHHHHHHHHHHHHHC
----SHPTYSEMIAAAIRAEKSRGGSSRQSIQKYIKSHYKVGHNADLQIKLSIRRLLAA
               |    ||    ||   |                        |
MRSSAKQEELVKAFKALLKEEK--FSSQGEIVAALQEQG-FDNINQSKVSRMLTKFGAV
CCCCHHHHHHHHHHHHHHHHCC--CCCHHHHHHHHHHC-CCCCCCHHHHHHHHHCCCC
        10        20          30        40        50
```

```
   60        70
CCEEEECCCCCCCEEEECC---------
GVLKQTKGVGASGSFRLAK---------

-RTRNA----KMEMVYCLPAELGVPTTS
-EEECC----EEEEEEECCCCCCCCCCC
   60           70
  2.
10         20              30           40
--------CHHHHHHHHHHHCCC-CHHHHHHHHCC------CHHHHHHHHCC----CCCC
--------SISSRVKSKRIQLGL-NQAELAQKVGT------TQQSIEQLENG----KTKR
             |       |  |             |            |
MRSSAKQEELVKAFKALLKEEKFSSQGEIVAALQEQGFDNINQSKVSRMLTKFGAVRTRN
CCCCHHHHHHHHHHHHHHHHHCCCCCHHHHHHHHHHHCCCCCCCHHHHHHHHHCCCCEEEC
       10        20        30        40        50        60
      50        60
CCCHHHHHHHCCCCHHHHHHCC-
PRFLPELASALGVSVDWLLNGT-
|
A-KMEMVYCLP--AELGVPTTS
C-EEEEEEECC--CCCCCCCCC
        70
```

Key to secondary-structure notation: C = coil, H = helix, E = strand.

In this example, two β-strands have been predicted at the C-terminus of the target protein. In alignment (1) these two strands have clearly been aligned with two strands in the template protein structure, whereas in alignment (2) the strands are aligned with a helix in the template structure. Clearly, alignment (1) is preferable. It is important to take into account the estimated reliability of the secondary-structure prediction in cases such as this. If the small C-terminal β-sheet had been predicted with very low confidence by the prediction program then the decision between alignments (1) and (2) would be far less clear-cut. In this case, however, the two C-terminal strands are very strongly predicted, and so the decision between the two alignments is easy to make.

9. Other Considerations

There is no substitute for biological knowledge in structure prediction. If the structure at position 3 in the ranked list of folds looks more plausible than the structure at the top of the list on the basis of biological function, the biologically plausible match should be carefully considered. It is also worth noting whether the predicted structure is a "superfold" ***(60)***. A matches to a superfold (e.g., TIM barrels, globins, immunoglobulins, and the like) does not imply any functional similarity with the target protein. However, if the target protein has a significant match to a trypsin-like serine protease fold (which is not a superfold), then this match should probably discounted unless it is suspected that the target protein is also a serine protease.

A further consideration is to check for consistency in the prediction. Extra confidence in the prediction can be gained if similar folds are found to be in the top positions of the ranked list. Obviously if three different TIM barrel folds are in the top three positions, then this is additional support for a TIM barrel-like fold prediction. Similarly, the whole prediction process can be repeated with one or more sequences homologous to the target protein ***(61)*** to see if the same prediction is made for each member of the sequence family. If a different fold is predicted for each member of the family, there can be little confidence in any of the predicted structures.

Finally, it is definitely worth trying more than one prediction program. Apart from THREADER2, several other fold-recognition methods are available via the Web, and if several different methods produce the same prediction, this significantly increases the degree of confidence in the prediction.

10. Problems with Threading

It must be very clearly understood that most threading programs are aimed at recognizing single globular protein domains, and perform very poorly when tried on proteins that are far from this ideal. If you are trying to thread a very large sequence, say 800 residues, unless you know where the domain boundaries are, you will not be very successful. Threading cannot be reliably used for identifying domain boundaries. If you do know where the domain boundaries are in your target sequence, then the sequence should be divided into domains before threading it, with each domain being threaded separately. Where predictions are attempted on very long multidomain sequences then you can be very suspicious of the results, unless it is clear that the matched protein has a similar domain structure to the target. For example, the periplasmic small-molecule binding proteins (e.g., arabinose-binding protein) are two domain structures (two doubly wound parallel ab domains), but they all match each other fairly well, as they all have identical domain organization. In contrast, however, pyruvate kinase has a number of quite distinct structural domains, and this organization is quite probably unique to pyruvate kinase. If a target is matched to pyruvate kinase without allowing for these distinct domains, bogus results willl result. In general, if it is possible the target sequence should be divided into likely domains before threading is attempted.

11. How Reliable Is Structure Prediction?

Although the published results for the fold-recognition methods look impressive, it is fair to argue that in all cases the correct answers were already known, so it is not clear how well they would perform in real situations where the answers are not known at the time the predictions are made. The results of a very ambitious worldwide experiment have recently been published in a spe-

cial issue of the journal Proteins ***(1)***, where an attempt was made to find out how successful different prediction methods were when rigorously blind-tested. In 1994, John Moult and colleagues approached X-ray crystallographers and NMR spectroscopists around the world and asked them to deposit the sequences for any structures they were close to solving in a database. Before these structures were made public, various teams around the world were then challenged with the task of predicting each structure. The results of this experiment were announced at a meeting held at Asilomar, CA, and this ambitious experiment has now become widely known as the Asilomar Experiment (or more commonly the Asilomar Competition). A second experiment was carried out in 1996, when this series of international experiments was given an official title: CASP (Critical Assessment in Structure Prediction), but the results of this second experiment are still as yet not published. Up-to-date information can be obtained from the following Web address, however:

```
http://predictioncenter.llnl.gov
```

In CASP1, the results for the comparative modeling and *ab initio* sections offered few surprises, in that the *ab initio* methods were reasonably successful in predicting secondary-structure but not tertiary, and homology modeling worked well when the proteins concerned had very high sequence similarity. The results for the fold-recognition section ***(62)***, however, showed great promise. Overall, roughly half of the structures in this part of the competition were found to have previously observed folds. Almost all of these structures were correctly predicted by at least one of the teams. The threading method of Jones et al. ***(58)*** proved to be the most successful method, with 5 out of 9 folds correctly identified, and with a looser definition of structural similarity, 8 out of 11 correct. These results show that, despite their relative early stage of development, fold-recognition methods (and threading methods in particular) offer very exciting prospects for prediction of protein tertiary structure in the near future. One point that should be made about the predictions from all of the fold-recognition groups was that the sequence–structure alignments were not as accurate as might have been hoped when judged against the alignments obtained from structural superposition of the two structures concerned. This is disappointing, but it is obvious that this will be improved as the methods mature.

12. The Future for Structure Prediction

One major difference between the academic challenge of the protein-folding problem and the challenge of producing viable prediction tools is that in the latter case there is an eventual end in sight. As more structures are solved, more target sequences will find matches in the existing library of known struc-

tures — matched either by sequence similarity, or by future developments of threading methods. In terms of practical applications, the protein-folding problem will begin to vanish. There will, of course, still be a need to better understand protein folding for applications such as protein design, and the problem of modeling membrane protein structure will remain unsolved for quite some time to come, but nonetheless, from a practical viewpoint, the problem will be salved (rather than solved). When will this point be reached? Given the variety of estimates for the number of naturally occurring protein folds, it is difficult to say, but with intelligent guesswork it seems likely that that when we have 1500 different folds in our fold libraries we will be in a position to build useful models for almost every globular protein sequence in a given proteome. At the present rate at which protein structures are being solved, this point is possibly 20 yr away, but there is now talk of large-scale attempts to crystallize every globular protein in a typical bacterial proteome. If such projects get underway, a complete fold library may be only 5–10 yr away. After this point, the protein folding problem will become largely an academic problem, but it will still be a challenge to understand how proteins fold.

References

1. Lattman, E. E. (1995) Protein structure prediction: a special issue. *Proteins* **23,** 295–460.
2. Read, J., Brayer, G., Jurek, L., and James, M. N. G. (1984) Critical evaluation of comparative model building of Streptomyces griseus trypsin. *Biochemistry* **23,** 6570–6575.
3. Greer, J. (1990) Comparative model building methods: application to the family of the mammalian serine proteases. *Proteins* **7,** 317–334.
4. Vásquez M. (1996) Modeling side chain conformation. *Curr. Opin. Struct. Biol.* **6,** 217–221.
5. Chung, S. Y. and Subbiah, S. (1996) How similar must a template protein be for homology modeling by side-chain packing methods, in *Proceedings of the First Pacific Symposium on Biocomputing: 1996* Jan 2–6; Kona, HI. (Hunter, L. and Klein, T., eds.) World Scientific, Singapore, pp. 126–141.
6. Moult, J. and James, M. N. G. (1986) An algorithm for determining the conformation of polypeptide segments in proteins by systematic search. *Proteins* **1,** 146–163.
7. Bruccoleri, R. E. and Karplus, M. (1987) Prediction of the folding of short polypeptide segments by uniform conformation sampling. *Biopolymers* **26,** 137–168.
8. Laskowski, R. A., MacArthur, M. W., Moss, D., and Thornton, J. M. (1993) PROCHECK, a program to check the stereochemical quality of protein structures. *J. Appl. Cryst.* **26,** 283–291.
9. Hooft, R. W. W., Sander, C., and Vriend, G. (1997) Objectively judging the quality of a protein structure from a Ramachandran plot. *CABIOS* **13,** 425–430.

10. Peitsch, M. C. (1996) PROMOD and SWISS-MODEL - Internet-based tools for automated comparative protein modeling. *Biochem. Soc. Trans.* **24,** 274–279.
11. Bowie, J. U., Lüthy, R., and Eisenberg, D. (1991) A method to identify protein sequences that fold into a known three-dimensional structure. *Science* **253,** 164–170.
12. Holm, L. and Sander, C. (1992) Evaluation of protein models by atomic solvation preference. *J. Mol. Biol.* **225,** 93–105.
13. Luthardt, G. and Frommel, C. (1994) Local polarity analysis: a sensitive method that discriminates between native proteins and incorrectly folded models. *Protein Eng.* **7,** 627–631.
14. Hendlich, M., Lackner, P., Weitckus, S., Floeckner, H., Froschauer, R., Gottsbacher, K., Casari, G., and Sippl, M. J. (1990) Identification of native protein folds amongst a large number of incorrect models: the calculation of low energy conformations from potentials of mean force. *J. Mol. Biol.* **216**, 167–180.
15. Jones, D. T., Taylor, W. R., and Thornton, J. M. (1992) A new approach to protein fold recognition. *Nature.* **358**, 86–89.
16. Sippl, M. J. and Weitckus, S. (1992) Detection of native-like models for amino acid sequences of unknown three-dimensional structure in a data base of known protein conformations. *Proteins* **13,** 258-271.
17. Godzik, A. and Skolnick, J. (1992) Sequence-structure matching in globular proteins: application to supersecondary and tertiary structure determination. *Proc. Natl. Acad. Sci. U. S. A.* **89,** 12,098–12,102.
18. Maiorov, V. N. and Crippen, G. M. (1992) Contact potential that recognizes the correct folding of globular proteins. *J. Mol. Biol.* **227,** 876–888.
19. Bryant, S. H. and Lawrence, C. E. (1993) An empirical energy function for threading protein-sequence through the folding motif. *Proteins: Struct. Funct. Genet.* **16,** 92–112.
20. Ouzounis, C., Sander, C., Scharf, M., and Schneider, R. (1993) Prediction of protein structure by evaluation of sequence-structure fitness. Aligning sequences to contact profiles derived from three-dimensional structures. *J. Mol. Biol.* **232,** 805–825.
21. Abagyan, R., Frishman, D., and Argos, P. (1994) Recognition of distantly related proteins through energy calculations. *Proteins: Struct. Funct. Genet.* **19,** 132–140.
22. Overington, J., Donnelly, D., Johnson, M. S., Sali, A., and Blundell, T. L. (1992) Environment-specific amino-acid substitution tables — tertiary templates and prediction of protein folds. *Prot. Sci.* **1,** 216–226.
23. Garnier, J., Gibrat, J. F., and Robson, B. (1996) GOR method for predicting protein secondary structure from amino acid sequence. *Methods Enzymol.* **266,** 540–553.
24. Chou, P. Y. and Fasman, G. D. (1978) Prediction of the secondary structure of proteins from their amino RT acid sequence. *Adv. Enzymol.* **47,** 45–148.
25. Barton, G. J. (1995) Protein secondary structure prediction. *Curr. Opin. Struct. Biol.* **5,** 372–376.
26. Rost, B. and Sander, C. (1995) Progress of 1D protein structure prediction at last. *Proteins* **23,** 295–300.
27. Eisenberg, D. (1997) Into the black of night. *Nat. Struct. Biol.* **4,** 95–97.

28. King, R. D. and Sternberg, M. J. E. (1996) Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Prot. Sci.* **5,** 2298–2310.
29. Gobel, U., Sander, C., Schneider, R., and Valencia, A. (1994) Correlated mutations and residue contacts in proteins. *Proteins* **18,** 309–317.
30. Shindyalov, I. N., Kolchanov, N. A., and Sander, C. (1994) Can three-dimensional contacts in protein structures be predicted by analysis of correlated mutations? *Protein Eng.* **7,** 349–358.
31. Taylor, W. R. and Hatrick, K. (1994) Compensating changes in protein multiple sequence alignments. *Protein Eng.* **7,** 341–348.
32. Thomas, D. J., Casari, G., and Sander, C. (1996) The prediction of protein contacts from multiple sequence alignments. *Protein Eng.* **9,** 941–948.
33. Kolinski, A. and Skolnick, J. (1994) Monte Carlo simulations of protein folding. II. Application to protein A, ROP, and crambin. *Proteins: Struct. Funct. Genet.* **18,** 353–366.
34. Kolinski, A. and Skolnick, J. (1994) Monte Carlo simulations of protein folding. I. Lattice model and interaction scheme. *Proteins: Struct. Funct. Genet.* **18,** 338–352.
35. Yee, D. P., Chan, H. S., Havel, T. F., and Dill, K. A. (1994) Does compactness induce secondary structure in proteins? A study of poly-alanine chains computed by distance geometry. *J. Mol. Biol.* **241,** 557–573.
36. Hinds, D. A. and Levitt, M. (1994) Exploring conformational space with a simple lattice model for protein structure. *J. Mol. Biol.* **243,** 668–682.
37. Yue, K., Fiebig, K. M., Thomas, P. D., Hue Sun, Chan, Shakhnovich, E. I., and Dill, K. A. (1995) A test of lattice protein folding algorithms. *Proc. Natl. Acad. Sci. USA* **92,** 325–329.
38. Dill, K. A., Bromberg, S., Yue, K., Fiebig, K. M., Yee, D. P., Thomas, P. D., and Hue Sun, Chan (1995) Principles of protein folding — a perspective from simple exact models. *Prot. Sci.* **4,** 561–602.
39. Park, B. H. and Levitt, M. (1995) The complexity and accuracy of discrete state models of protein structure. *J. Mol. Biol.* **249,** 493–507.
40. Abkevich, V. I., Gutin, A. M., and Shakhnovich, E. I. (1995) Domains in folding of model proteins. *Prot. Sci.* **4,** 1167–1177.
41. Rykunov, D. S., Reva, B. A., and Finkelstein, A. V. (1995) Accurate general method for lattice approximation of three-dimensional structure of a chain molecule. *Proteins: Struct. Funct. Genet.* **22,** 100–109.
42. Dewitte, R. S., Michnick, S. W., and Shakhnovich, E. I. (1995) Exhaustive enumeration of protein conformations using experimental restraints. *Prot. Sci.* **4,** 1780–1791.
43. Covell, D. G. (1994) Lattice model simulations of polypeptide chain folding. *J. Mol. Biol.* **235,** 1032–1043.
44. Srinivasan, R. and Rose, G. D. (1995) Linus - a hierarchical procedure to predict the fold of a protein. *Proteins* **22,** 81–99.
45. Dandekar, T. and Argos, P. (1994) Folding the main chain of small proteins with the genetic algorithm. *J. Mol. Biol.* **236,** 844–861.

46. Sun, S. (1995) A genetic algorithm that seeks native states of peptides and proteins. *Biophys. J.* **69,** 340–355.
47. Pederson, J. T. and Moult, J. (1995) Ab initio structure prediction for small polypeptides and protein fragments using genetic algorithms. *Proteins* **23,** 454–460.
48. Pedersen, J. T. and Moult, J. (1996) Genetic algorithms for protein structure prediction. *Curr. Opin. Struct. Biol.* **6,** 227–231.
49. Aszodi, A. and Taylor, W. R. (1996) Homology modelling by distance geometry. *Fold. Des.* **1,** 325–334.
50. Skolnick, J., Kolinski, A., and Ortiz, A. R. (1997) MONSSTER: a method for folding globular proteins with a small number of distance restraints. *J. Mol. Biol.* **265,** 217–241.
51. Pearson, W. R. (1990) Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol.* **183,** 63–98.
52. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J. H., Zhang, Z., Miller, W., and Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25,** 3389–3402.
53. Smith, T. F. and Waterman, M. S. (1981) Comparison of bio-sequences. *Adv. Appl. Math.* **2,** 482–489.
54. Gribskov, M., Lüthy, R., and Eisenberg, D. (1990) *Meth. Enzymol.* **188,** 146-159.
55. Krogh A., Brown M., Mian I. S., Sjoelander K., and Haussler D. (1994) Hidden Markov model in computational biology. Applications to protein modelling. *J. Mol. Biol.* **235,** 1501–1531.
56. Altschul S. F., Gish W., Miller W., Myers E. W., and Lipman D. J. (1990) Basic local alignment search tool. *J. Mol. Biol.* **215,** 403–410.
57. Abola, E. E., Bernstein, F. C., Bryant, S. H., Koetzle, T. F., and Weng, J. (1987) Protein Data Bank, in *Crystallographic Databases*, Data Commission of the International Union of Crystallography, Bonn/Cambridge/Chester, pp. 107–132.
58. Jones, D. T., Miller, R. T., and Thornton, J. M. (1995) Successful protein fold recognition by optimal sequence threading validated by rigorous blind testing. *Proteins* **23,** 387–397.
59. Miller, R. T., Jones, D. T., and Thornton, J. M. (1996) Protein fold recognition by sequence threading — tools and assessment techniques. *FASEB J.* **10,** 171–178.
60. Orengo, C. A., Jones, D. T., and Thornton, J. M. (1994) Protein superfamilies and domain superfolds. *Nature* **372,** 631-634.
61. Edwards, Y. J. K. and Perkins, S. J. (1996) Assessment of protein fold predictions from sequence information: the pedicted alpha/beta doubly wound fold of the von Willebrand factor ype a domain is similar to its crystal structure. *J. Mol. Biol.* **260,** 277–285.
62. Lemer, C. M. R., Rooman, M. J., and Wodak, S. J. (1995) Protein structure prediction by threading methods: evaluation of current techniques. *Proteins* **23,** 337–355.
63. Burmeister, W. P., Henrissat, B., Bosso, C., Cusack, S., and Ruigrok, R. W. (1993) Influenza B virus neuraminidase can synthesize its own inhibitor. *Structure* **1,** 19–26.

8

Derivation and Testing Residue–Residue Mean–Force Potentials for Use in Protein Structure Recognition

Boris A. Reva, Alexei V. Finkelstein, and Jeffrey Skolnick

1. Introduction

In protein-structure prediction, simplified energy functions are necessarily used to allow fast sorting over many conformations. As a rule, these functions are derived from residue–residue approximation, which attributes all atomic interactions between residues to a single point within each residue. Physically, the simplified energies should result from averaging of the atomic interactions over various positions and conformations of the interacting amino acid residues, as well as the surrounding solvent molecules. Unfortunately, direct calculation of such mean-force potentials is not possible today both because of methodological difficulties and the lack of reliable atom-based energy functions.

However, the rapidly increasing database of protein structures induced many attempts to derive potentials from structural information of proteins ***(1–8)***. Most of these approaches exploit Boltzmann's equation, which stresses that frequently observed states are the low-energy states. The exponential occurrence-on-energy dependency has been shown to be valid also for fixed and nonfluctuating native protein structures ***(9)***, although, as it has been shown recently ***(10)***, the Boltzmann-like statistics of native protein structures is maintained by the sequence mutations rather than by thermal fluctuations of the structure, i.e., its physical origin is absolutely different (although its mathematical form is similar) from that of the conventional Boltzmann statistics of thermodynamic ensembles.

Here we apply the results of that analysis to derive energy functions from known protein structures. Our approach ***(11,12)*** is in many, but not in all,

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

respects similar to the one originally used by Sippl ***(1,2)***. We also derive pairwise, distance-dependent "mean-force" potentials, treating long-range and short-range interactions separately. However, our methods of choosing the reference state for long-range interactions and our treatment of short-range interactions differ from those used by Sippl.

2. Derivation of Pairwise Potentials

2.1. Preparation of the Database

The protein structures used for derivation of energy functions were taken from the 25% similarity list of Hobohm et al. ***(13)***. Any pair of proteins in this list has a similarity of less than 25% according to the Smith and Waterman ***(14)*** sequence alignment with gap (open-gap penalty, 3.0, gap-elongation penalty, 0.05). From the Hobhom et al. list of October 1997, we entered into our database 372 proteins having no chain breaks, with a resolution better than 2.5 Å and an *R* factor less than 0.2. To avoid structurally similar proteins, we determine all the cases of low root-mean-square derivation (less than 10 Å) between Ca atom-traced structural pairs. Each of these pairs was analyzed with the structural classification protein (SCOP) ***(15)*** to find if the proteins of the pair belonged to the same protein family or superfamily. As a result of this analysis, 13 protein chains chosen as the shortest among the homologous pairs, were removed from the database of 372 proteins. The resulting database includes 359 proteins.

2.2. Extraction of Energy Functions from Protein Statistics

The main task was to estimate the interaction potential $\varepsilon_{\alpha\beta}(r)$ for each pair of residues α and β (α, β = Gly, Ala, …) divided by a distance r; r is defined from the positions of the C_β (or, for some short-range interactions [*see* below], of the C_α) atoms. In these estimates, we followed the theory of Boltzmann-like statistics of protein structures by Finkelstein et al. ***(10)***.

This theory describes a Boltzmann-like form of protein statistics not using, as usually assumed, ***(1,5)*** a model of a gaslike distribution of residues in a protein globule (which have been recently criticized ***(16)*** for its physical incorrectness), but a more natural assumption that protein sequences change with random mutations that have to maintain the stability of the protein structure. As a result, each low-energy structural detail (low-energy, residue-to-residue contact, bend, and so on) increases the number of "protein-stabilizing" sequences (and therefore this detail is observed often), whereas each high-energy detail decreases this number and therefore is observed rarely. It is shown also ***(10)*** that a change in the number of protein-stabilizing sequences is exponentially (as in Boltzmann's law) dependent on the energy of the detail in question.

The results of this theory, as applied to the obtaining of energy parameters from protein statistics, can be summarized as follows:

Let us consider a large 3D database of protein structures, and define $N^s_{\alpha\beta}$ as the number of the αβ-pairs occupying positions i,i+s along a chain (α and β are amino acids, i is any position in a chain), and $N^s_{\alpha\beta}(r)$ as the number of such pairs at a distance between α_i and β_{i+s} in the database.

According to **ref. *10***, the expected value of $N^s_{\alpha\beta}(r)$ in the limit of very large statistics, is:

$$N^s_{\alpha\beta}(r) = AN^s_{\alpha\beta}(r)\, w^s(r)\exp[-\Delta E^s_{\alpha\beta}(r)/RT_c] \quad (1)$$

where A is a distance-independent normalization constant; $w^s(r)$ is a probability of finding i,i+s residues at a distance r in the total set of globular folds ($w^s(r) = N^s(r)/\sum_r N^s(r)$, where $N^s(r) = \sum_\alpha \sum_\beta N^s_{\alpha\beta}(r)$ is the number of cases where i,i+s residues are at a distance r, T_c is a "conformatiomal temperature" ***(9)***, which is close to the characteriztic temperature of freezing of native folds approx 300 K ***(10)***; R is the gas constant; and $\Delta E^s_{\alpha\beta}(r)$ is the effective interaction energy:

$$\Delta E^s_{\alpha\beta}(r) = \varepsilon^s_{\alpha\beta}(r) + \tilde{E}^s_{\alpha\beta}(r) \quad (2)$$

Here, $\varepsilon^s_{\alpha\beta}(r)$ is the energy of direct interaction between residues α and β at a distance r, and $\tilde{E}^s_{\alpha\beta}(r)$ is the mean (averaged over all the possible environments of the pair αβ in stable protein structures) energy of indirect interaction of α and β, i.e., of the interaction mediated by all the surrounding residues.

Thus, taking into account the proportionality $w^s(r) \sim N^s(r)$, one can write

$$N^s_{\alpha\beta}(r_1)/N^s_{\alpha\beta}(r_2) = N^s(r_1)/N^s_{\alpha\beta}(r_2) \cdot \exp(-[\varepsilon^s_{\alpha\beta}(r_1) - \varepsilon^s_{\alpha\beta}(r_2)] + \tilde{E}^s_{\alpha\beta}(r_1) - \tilde{E}^s_{\alpha\beta}(r_2)]/RT_c) \quad (3)$$

which corresponds to **Eq. 10** of ***(10)***, where the term Δ_E therein would now include $\varepsilon^s_{\alpha\beta}(r_1) - \varepsilon^s_{\alpha\beta}(r_2)$, whereas $\tilde{E}^s_{\alpha\beta}(r_1) - \tilde{E}^s_{\alpha\beta}(r_2)$, which depends on the possible amino acid environments of the αβ pair, will contribute to both Δ_E and $\Delta_\sigma/2RT_c$ terms in that work.

The direct residue–residue interaction energy estimated from **Eqs. 1** and **2** gives:

$$\varepsilon^s_{\alpha\beta}(r) = -RT_c\ln[N^s_{\alpha\beta}(r)/N^s_{\alpha\beta} \cdot w^s(r)] + RT_c \cdot ln\ \mathrm{A} - \tilde{E}^s_{\alpha\beta}(r) \quad (4)$$

It is noteworthy that, because the Boltzmann-like statistics of proteins originates from amino acid mutations, the reference (zero-energy) state for the energy $\varepsilon^s_{\alpha\beta}(r)$ obtained from these statistics is a pair of "average" amino acid residues (in a compact "proteinlike" environment with a secondary-structure and no sequence specificity) separated by a distance in the chain and in space rather than an amino acid pair in vacuum or water environment (cf. **refs. *4–6***).

Equation 4 is valid only when the expected $w^s(r)$ value is not zero. When $w^s(r) = 0$, $\varepsilon^s_{\alpha\beta}(r)$ cannot be defined from **Eq. 4**, but must be set to infinity to make impossible any structure with the distance r between any residues.

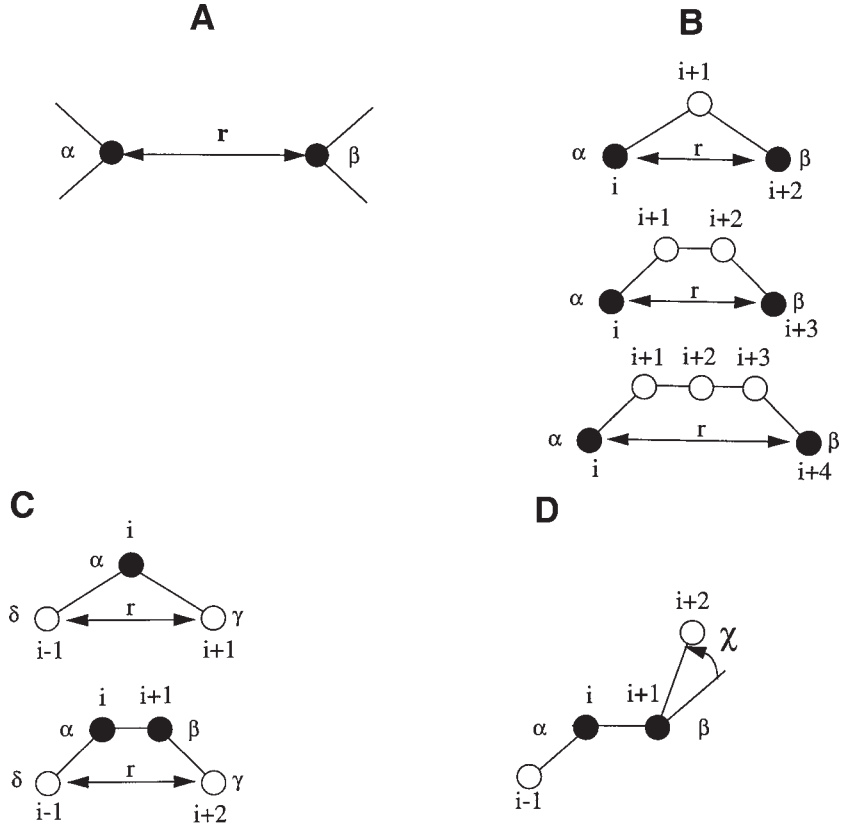


Fig. 1. A scheme of short-range interactions; residues for which potentials are derived are shown by filled circles. **(A)** Long-range interactions depending on the distance between remote residues α and β. **(B)** Short-range interactions depending on the distance between terminal residues α and β. **(C)** Short-range interactions depending on chain bending in the intervening residue α (or α and β), which affects the distance between terminal residues γ and γ. **(D)** Chiral energy depending on dihedral angle χ between two planes (*i – 1,i,i + 1*) and (*i,i + 1,i + 2*) (determined by the corresponding C_α atoms) and residues α and β, which affect the value of χ.

2.3. Long-Range Interactions

When residues are separated in the chain ($s > s_0 \gg 1$; *see* **Fig. 1A**) so that they can be at a distance where they do not interact, the precise value of *s* is not important. Moreover, the order of residues in a pair (αβ or βα) is not relevant. Our experience ***(12)*** shows that the potentials of the long-range interactions should be based on the distances between the C_β atoms (for the exception of Gly residues, where C_α is used, as C_β is absent).

Let us define $N_{\alpha\beta}(r)$ as the total number of cases where the αβ and βα pairs separated by more than s_0 chain residues occur at a distance, or rather in an interval (the value of the resolution interval D is discussed and optimized below):

$$N_{\alpha\beta}(r) = \sum_{p=1}^{P} \sum_{i=1}^{N_p - s_0} \sum_{j=i+s_0}^{N_p} (\delta_{q_i\alpha}\delta_{q_j\beta} + \delta_{q_i\beta}\delta_{q_j\alpha} - \delta_{q_i\alpha}\delta_{q_j\beta})\ \theta\ (\Delta/2 - |r_{ij} - r|) \tag{5}$$

where P is a number of proteins, N_p is a protein p sequence length; q_i is a residue i type; r_{ij} is a distance between residues i and j; $\delta_{\alpha\beta} = 1$, if $\alpha = \beta$ and $\delta_{\alpha\beta} = 0$, if $\alpha \neq \beta$; $\theta(x) = 1$, if $x \geq 0$ and $\theta(x) = 0$, if $x < 0$.

Let us also define $N^0_{\alpha\beta}(\geq R_{\alpha\beta})$ as the total number of cases, where residue pairs αβ and βα remote along a chain occur at noninteraction distances:

$$N^0_{\alpha\beta}(\geq R_{\alpha\beta}) = \sum_{p=1}^{P} \sum_{i=1}^{N_p - s_0} \sum_{j=i+s_0}^{N_p} (\delta_{q_i\alpha}\delta_{q_j\beta} + \delta_{q_i\beta}\delta_{q_j\alpha} - \delta_{q_i\alpha}\delta_{q_j\beta}\delta_{\alpha\beta})\ \theta\ (r_{ij} - R_{\alpha\beta}) \tag{6}$$

where $R_{\alpha\beta}$ is the minimal distance where direct interaction between α and β residues is absent (i.e., $\varepsilon_{\alpha\beta}(r) = 0$ for $r \geq R_{\alpha\beta}$); the values of $R_{\alpha\beta}$ are defined below.

Then the value of $\varepsilon_{\alpha\beta}(r)$ for the long-range interactions can be estimated as ***(11–12)***:

$$\varepsilon_{\alpha\beta}(r) = -RT_c \ln \left[\frac{N_{\alpha\beta}(r)}{N_{\alpha\beta} \cdot w(r)} \Big/ \frac{N^0_{\alpha\beta}(\geq R_{\alpha\beta})}{N_{\alpha\beta} \cdot w^0(\geq R_{\alpha\beta})} \right] - [\tilde{E}_{\alpha\beta}(r) - \tilde{E}_{\alpha\beta}(\geq R_{\alpha\beta})] \tag{7}$$

where $w(r)$ and $w^0(\geq R_{\alpha\beta})$ are the probabilities of finding the remote residue pairs at the distances r and $r \geq R_{\alpha\beta}$, respectively, in the total set of globular proteins.

The term $\tilde{E}_{\alpha\beta}(\geq R_{\alpha\beta})$ is the average energy of the indirect interactions at $r \geq R_{\alpha\beta}$; because of the averaging of indirect interactions over all the distances $r \geq R_{\alpha\beta}$: this term is small and can be neglected. The term $\tilde{E}_{\alpha\beta}(r)$ can be neglected at small distances $r < R_{\alpha\beta}$ where a direct interaction of two residues is strong.

Thus, one can $\varepsilon_{\alpha\beta}(r)$ estimate as:

$$\varepsilon_{\alpha\beta}(r) = -RT_c \ln [N_{\alpha\beta}(r) / N^*_{\alpha\beta}(r)] \tag{8}$$

where

$$N^*_{\alpha\beta}(r) = N^0_{\alpha\beta}(\geq R_{\alpha\beta}) \frac{w(r)}{w^0(\geq R_{\alpha\beta})} = N^0_{\alpha\beta}(\geq R_{\alpha\beta}) \frac{\sum\limits_{a \geq b} \sum N_{\alpha\beta}(r)}{\sum\limits_{a \geq b} \sum N_{\alpha\beta}(\geq R_{\alpha\beta})} \tag{9}$$

In **Eq. 9**, the ratio of probabilities $w(r)/w^0(\geq R_{\alpha\beta})$ is approximated by the ratio of the total number of all the remote residue pairs found at a distance r, to the

Table 1
Effective Residue Radii (in Å) Used in Derivation of Long-Range Potentials[a]

Gly	Ala	Pro	Asn	Leu	Val	Ser	Thr	Cys	Asp	Ile	His	Gln	Glu	Met	Phe	Lys	Trp	Tyr	Arg
3.9	4.9	4.9	4.9	4.9	5.0	5.0	5.0	5.0	5.0	5.0	5.1	5.1	5.2	5.6	5.7	6.7	6.8	7.1	7.6

[a]Covalent residue radii (*see* **ref. *12***) are adjusted by efecective van der Waals radius d/2 = 1.2 Å (*see* **Eq. 10**).

total number of all the residue pairs at all the distances $r \geq R_{\alpha\beta}$; (sums are taken over all $20 \cdot (20 + 1)/2 = 210$ different residue pairs; the pairs $\alpha\beta$, where $\alpha < \beta$, are taken into account in $\beta\alpha$ pairs).

Equations 8 and **9** show that the value of $\varepsilon_{\alpha\beta}$ does not change with simultaneous multiplication of all the $N_{\alpha\beta}(r)$ terms with a function depending on r (when $r \geq R_{\alpha\beta}$), but does with a function of α and β. This once again shows that the foregoing definition of $\varepsilon_{\alpha\beta}(r)$ counts the interaction energy from the interaction energy $\varepsilon_0(r)$ for some "average" residue pair, and the function $\varepsilon_0(r)$ cannot be found from protein statistics directly. In this study the simple assumption that

$$\varepsilon_0(r) = \begin{cases} 0, \text{ when } r > R_{\min} \\ +\infty, \text{ when } r \leq R_{\min} \end{cases} \tag{8a}$$

where $R_{\min}$ is an adjustable radius ($R_{\min} \approx 2.5 - 3.0$ Å, *see* below) works well enough.

To calculate potentials using formulae **Eqs. 8** and **9**, one needs to determine the threshold distances $R_{\alpha\beta}$ (*see* **Table 1**). We used the estimate:

$$R_{\alpha\beta} = R_\alpha' + R_\beta' \tag{10}$$

where $R_\alpha' = R_\alpha + \delta/2$ and $R_\beta' = R_\beta + \delta/2$ are "effective" radii of residues α and β, respectively. For a "covalent" residue radius R_α, we simply took the maximal (overall residues of a given type α in a database) distance between the C_β (or C_α for Gly) atom and any other heavy atoms of the residue. To convert a "covalent radius" into something like van der Waals radius R' of a residue, we add ($\delta/2$) ~ 1.2 Å.

2.4. Short-Range Interactions Depending on Distance Between Residues

In this study, short-range interactions are defined as the ones between residues occupying positions $i,i + 2$, $i,i + 3$, and $i,i + 4$ along a chain. This corresponds to (*see* **Fig. 1B**).

To estimate these interactions, we neglect the nonimportant distance-independent term ln A, and also the energy of indirect interactions, $\tilde{E^{\sigma}}_{\alpha\beta}(r)$ (which

is of a secondary importance, as the residues close in a chain are also close in space) in **Eq. 3**, and approximate the probability of finding a pair $i,i+s$ at a given distance r by the ratio of the total number of all $i,i+s$ residues pairs found at a distance r, to the total number of $i,i+s$ residue pairs found at all the distances. We found ***(12)*** that potentials based on distances between C_β atoms (C_α atoms for Gly) are more accurate than the ones based on C_α atoms only.

Thus, for $s = 2,3,4$ we have

$$\varepsilon^s_{\alpha\beta}(r) = -RT_c \ln [N^s_{\alpha\beta}(r) / N^{*s}_{\alpha\beta}(r)] \tag{11}$$

where

$$^s_{\alpha\beta}(r) = \sum_{p=1}^{P} \sum_{i=1}^{N_p - s} (\delta_{q_i\alpha}\delta_{q_{i+s}\beta} + \delta_{q_i\beta}\delta_{q_{i+s}\alpha} - \delta_{q_i\alpha}\delta_{q_{i+s}\beta}\delta_{\alpha\beta})\, \theta\, (|r_{i,i+s} - r|) \tag{12}$$

and

$$N^{*s}_{\alpha\beta}(r) = \sum_r N^s_{\alpha\beta}(r) \frac{\sum_\alpha \sum_\beta N^s_{\alpha\beta}(r)}{\sum_\alpha \sum_\beta \sum_r N^s_{\alpha\beta}(r)} \tag{13}$$

For short-range interactions we distinguish between pairs $\alpha\beta$ and $\beta\alpha$.

2.5. Short-Range Interactions Depending on Chain Bending

The distance between two residues in positions also depends on residues that occupy intervening positions (*see* **Fig. 1C**): these residues determine the local chain stiffness.

To take into account these interactions we follow the above approach and introduce two "bending-energy" terms:

$$u^{(2)}_\alpha(r) = -RT_c \ln [\tilde{N}^{(2)}_\alpha(r) / \tilde{N}^{*(2)}_\alpha(r)] \tag{14}$$

and

$$u^{(3)}_{\alpha\beta}(r) = -RT_c \ln [\tilde{N}^{(3)}_{\alpha\beta}(r) / \tilde{N}^{*(3)}_{\alpha\beta}(r)] \tag{15}$$

where

$$\tilde{N}^{(2)}_\alpha(r) = \sum_{p=1}^{P} \sum_{i=1}^{N_p - 2} \delta_{q_{i+1}\alpha}\, \theta\, (\Delta/2 - |r_{i,i+s} - r|) \tag{16}$$

$$\tilde{N}^{*(2)}_\alpha(r) = \sum_r \tilde{N}^{(2)}_\alpha(r) \frac{\sum_\alpha \tilde{N}^{(2)}_\alpha(r)}{\sum_r \sum_\alpha \tilde{N}^{(2)}_\alpha(r)} \tag{17}$$

and

$$\tilde{N}^{(3)}_{\alpha\beta}(r) = \sum_{p=1}^{P} \sum_{i=1}^{N_p - 3} \delta_{q_{i+1}\alpha}\delta_{q_{i+2}\beta}\, \theta\, (\Delta/2 - |r_{i,i+3} - r|) \tag{18}$$

$$\tilde{N}^{*(3)}_{\alpha\beta}(r) = \sum_r \tilde{N}^{(3)}_{\alpha\beta}(r) \frac{\sum_\alpha \sum_\beta \tilde{N}^{(3)}_{\alpha\beta}(r)}{\sum_\alpha \sum_\beta \sum_r \tilde{N}^{(3)}_{\alpha\beta}(r)} \tag{19}$$

Here, $\tilde{N}^{(2)}_{\alpha}(r)$ is the number of pairs $i,i+2$ with a distance r between i and $i+2$ residues and the residue α in the $i+1$ position; $\tilde{N}^{(3)}_{\alpha\beta}(r)$ is the number of $i,i+3$ pairs with a distance r between i and $i+3$ residues and residues α in $i+1$ and β in $i+2$ positions (*see* **Fig. 1C**). We derive bending potentials using distances between C_α atoms ***(12)***, as they were found to be more accurate than the ones based on C_β atoms.

2.6 Chiral Energy Term

We define local chirality of a protein chain as a dihedral angle χ between two planes determined by C_α atoms of residues occupying positions $(i-1,i,i+1)$ and $(i,i+1,i+2)$ along a chain (*see* **Fig. 1D**). As usually, the angle χ is counted off the $(i-1,i)$ vector to the $(i+1,i+2)$ vector in a counterclockwise (if looking from $i+1$ to i) direction. Because secondary-structure elements are well distinguished by local chirality (α-helices have the angle close to 120°, whereas the beta structure has the angle close to zero), the residue preferences in choosing a secondary-structure type are virtually taken into account by introducing the chirality term. This chirality potential is calculated as:

$$v_{\alpha\beta}(\chi) = -RT_c \ln [\hat{N}_{\alpha\beta}(\chi) / \hat{N}^*_{\alpha\beta}(\chi)] \tag{20}$$

where

$$\hat{N}_{\alpha\beta}(\chi) = \sum_{p=1}^{P} \sum_{i=1}^{N_p-2} \delta_{q\alpha}\delta_{q_{i+1}\beta}\, \theta\,(\Delta_\chi/2 - |\chi_{i-1,\,i,\,i+1,\,i+2} - \chi|) \tag{21}$$

and

$$\tilde{N}^{*(3)}_{\alpha\beta}(r) = \sum_r \tilde{N}^{(3)}_{\alpha\beta}(r) \frac{\sum_\alpha \sum_\beta \tilde{N}^{(3)}_{\alpha\beta}(r)}{\sum_\alpha \sum_\beta \sum_r \tilde{N}^{(3)}_{\alpha\beta}(r)} \tag{22}$$

Here, $\hat{N}_{\alpha\beta}(\chi)$ is the number of $\alpha\beta$ pairs occupying position $i,i+1$ along a chain when a dihedral angle χ formed by C_α atoms of residues $(i-1,i,i+1,i+2)$ falls into interval $(c - \Delta_\chi/2;\ c + \Delta_\chi/2)$.

2.7. Sparse Statistics

Above, all the potentials where obtained from equations having a general form

$$\varepsilon_x(q) = -RT_c \ln [N_x(q) / N^*_x(q)] \tag{23}$$

where $x = \alpha$ for u_α potential, and $x = \alpha\beta$ pair for all other $\varepsilon_{\alpha\beta}$, $u_{\alpha\beta}$, and $v_{\alpha\beta}$ potentials, whereas $N_x(q)$ and $N^*_x(q)$ are the observed and expected number of

residue pairs, respectively ($q = \chi$ for the chiral and $q = r$ for all other potentials). **Equation 23** is not applicable for the cases of sparse statistics when one or both of the terms, $N_x(q)$ and $N^*_x(q)$, are equal to zero. In these cases we define the potentials as follows:

$$\varepsilon_x(q) = +\infty \quad \text{if} \quad N^*_x(q) = 0 \tag{24}$$

$$\varepsilon_x(q) = RT_c \cdot N^*_x(q) \quad \text{if} \quad N_x(q) = 0 \quad \text{and} \quad N^*_x(q) \neq 0 \tag{25}$$

Equation 24 is obvious: in this way, we forbid interresidue distances which, for any physical reason, are not observed in any protein structures (*see* above).

Equation 25 is rather arbitrary; we use it to obtain some kind of high energy and, simultaneously, to avoid an infinity that can be caused by sparse statistics rather than by the physical impossibility of particles at a distance from each other.

The energy of a chain conformation is the sum of all the individual terms described.

2.8. Statistical Errors in Potential Estimates

The accuracy of phenomenological potentials depends on the size of the database used for their derivation. It is important for applications to have an estimate of the statistical error arising from the finite size of the database.

Such an estimate can be easily made in the following way: let us divide a database of protein structures into two approximately equal subdatabases, **A** and **B**, and let us derive two corresponding sets of potentials: sets **A** and **B**. Because of statistical fluctuations, potentials **A** and **B** will be slightly different. One can estimate the amplitude of statistical error for a potential $\varepsilon_x(r)$ as:

$$\Delta\varepsilon_x = |\varepsilon^A_x - \varepsilon^B_x| / 2 \tag{26}$$

where ε^A_x and ε^B_x and are potentials corresponding to the databases **A** and **B**.

In the case of sparse statistics, when $N^A_x(r) = 0$ and/or $N^B_x(r) = 0$, the values of $RT_c \cdot N^{*A}_x(r)/2$ and/or $RT_c \cdot N^{*B}_x(r)/2$ are added to the value of $\Delta\varepsilon_x$.

3. Testing of Potentials in Gapless Threading Test

The accuracy of potentials is estimated using the threading test suggested by Hendlich et al. ***(17)***. In this test, the energy of the native structure is compared with the energies of alternative structures obtained by threading the native sequence through all possible structural conformations provided by the backbones of a set of proteins. No gaps or insertions are allowed, thus, a chain of **N** residues long can be threaded through a host protein molecule of **M** residues long in **M – N + 1** different ways. Because glycine residues have no C_β atoms (which are necessary for threading with C_β atom-based potentials) we constructed virtual C_β atoms for all glycine residues of the threading database.

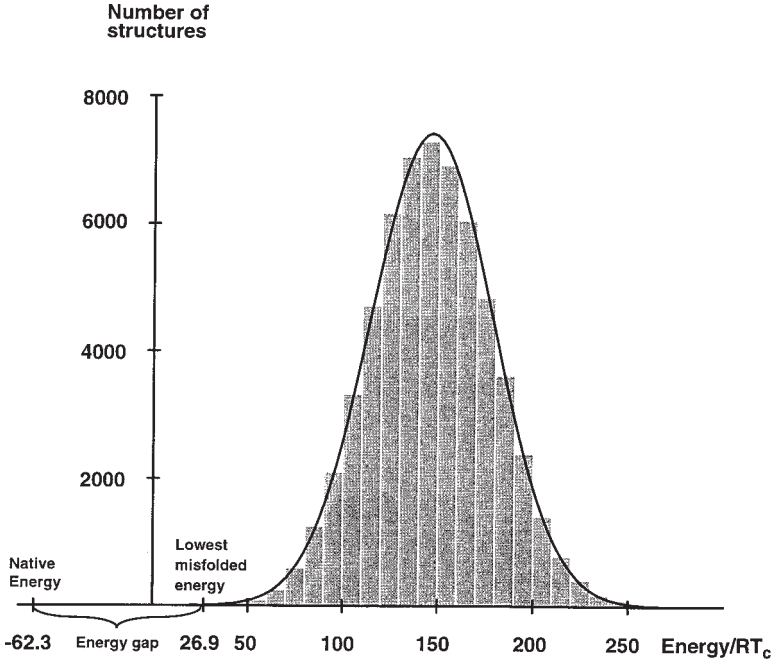


Fig. 2. The histogram and the corresponding normal distribution (thin line) of 59,786 threading energies of the rotamase molecule (1fkj). The normal distribution is built with an average energy of 146.4 and a standard deviation of 32.1. The difference between the average energy and the native structure energy is 208.7, which corresponds to $Z = 208.7/32.1 = 6.5$. The difference between the lowest energy of the misfolded structures and the native structure energy gives the value of the energy gap (89.2) separating the native structure from misfolded ones.

A typical example of the energy distribution for the rotamase molecule (1fkj) in such a threading experiment is shown in **Fig. 2**.

To analyze the contribution of different energy terms in the recognition of protein structure we compare in **Tables 2** and **3** the averaged characteriztics of threading tests for 50 structures given separately by each of the energy terms (Fifty testing structures were chosen as the shortest ones from the database of 359 proteins. For each of these 50 proteins the rest of the database was used both for derivation of potentials and as a source of alternative structures for threading.)

Tables 2 and **3** also show how the accuracy of the different energy terms depends on the size of the resolution interval. The table presents common measures of the fold-recognition quality, such as (1) the average ranking, (2) the average energy gap width, and (3) the average *Z*-score; *Z*-score is a relative

Table 2
Characteristics of Native Structure Recognition in Threading Tests by Different Distance-Dependent and Chain-Bending Energy Terms at Different Resolutions[a]

Resolution (Å)		0.25	0.5	1.0	2.0	3.0
	<p>[b]	2.11	2.02*	2.14	2.33	2.38
L	<Z>[c]	5.31	5.35	5.52*	5.45	5.47
	<G>[d]	25.08	25.62*	24.61	21.38	18.61
	<P>	224.81	87.04*	138.68	420.12	769.13
S2	<Z>	2.54	2.70*	2.49	2.46	1.81
	<G>	–8.99	–6.31	–5.94	–5.64*	–6.08
	<P>	497.85	270.34*	307.57	513.02	370.58
S3	<Z>	1.95	2.39*	2.29	2.14	2.35
	<G>	–12.92	–9.10	–8.07	–8.10	–5.80*
	<P>	2386.96	1244.44	817.63	785.51*	819.04
S4	<Z>	1.64	1.95	2.10	2.08	2.12*
	<G>	–18.22	–12.60	–9.01	–7.63	–6.98*
	<P>	34.32*	61.50	669.22	2228.96	2153.32
B2	<Z>	1.18	2.95*	2.31	1.88	1.81
	<G>	–3.11*	–3.40	–4.30	–4.70	–3.62
	<P>	6.52	5.10*	5.47	21.01	39.70
B3	<Z>	3.54	3.89	3.92*	3.21	3.19
	<G>	–0.91	0.81*	–0.06	–3.03	–4.29

[a]Long-range (L), short-range (S2, S3, S4), and bending (B2, B3) interactions s hown, respectively, in **Fig. 1**; C_β atoms are used as force centers in L, S2, S3, and S4 energy terms and C_α atoms in B2 and B3 terms.

[b]Average position is defined as the geometrical mean: $\langle P\rangle = [\prod_{i=1}^{N} P_i]^{1/N}$, where P_i is the position of the native fold energy for chain i in the energy-sorted list for this chain, and N is the number of proteins.

[c]Average Z-score; defined as a root-mean-square: $\langle P\rangle = \left[(1/N)\sum_{i=1}^{N} Z^2_i\right]^{1/2}$ where Z_i is Z-score corresponding to the native fold energy of protein i.

[d]Average (arithmetic mean) energy gap (in RT_c units) between the lowest energy of an alternative structure and the native structure.

*The best value.

Table 3
Characteristics of the Native Structure Recognition in Threading Tests by Chiral Energy Term at Different Resolutions

Resolution (degrees)	36	30	24	20	18	15	12	10
<P>[b]	3.76	2.94	4.06	4.15	3.36	4.90	6.21	9.09
<Z>[c]	3.61	3.87	3.83	3.94	4.03	4.10	3.98	3.87
<G>[d]	2.82	3.56	2.41	2.95	3.39	3.36	1.55	–0.21

[b,c,d,*]See the corresponding footnotes to **Table 2**.

Table 4
Characteristics of Native Structure Recognition in Threading Tests by Different Combinations of Energy Terms at Different Resolutions[a]

Resolution (Å)		0.25	0.5	1.0	2.0	3.0
	<P>[b]	2.11	2.02	2.14	2.33	2.38
L	<Z>[c]	5.31	5.35	5.52	5.45	5.47
	<G>[d]	25.08	25.62	24.61	21.38	18.61
	<P>	2.87	1.90	2.71	6.00	9.72
SB	<Z>	3.82	4.66	4.30	3.73	3.51
	<G>	6.61	11.96	5.13	–1.09	–2.43
	<P>	1.59	1.35	1.58	1.85	2.43
SBC	<Z>	4.54	5.23	4.98	4.65	4.54
	<G>	23.54	27.31	20.31	13.18	10.96
	<P>	1.18	1.09	1.09	1.22	1.45
LSB	<Z>	5.79	6.53	6.43	6.19	6.06
	<G>	68.93	71.02	61.88	47.47	40.33
	<P>	1.16	1.06	1.07	1.10	1.20
LSBC	<Z>	6.16	6.79	6.72	6.55	6.47
	<G>	87.50	89.32	81.56	68.21	60.80

[a]L, SB, SBC, LSB, and LSBC correspond to different combinations of local and long-range terms: long-range (L); short-range and bending terms (SB); short-range, bending, and chiral terms (SBC); long-range, short-range, and bending terms (LSB); long-range, short-range, bending, and chiral terms (LSBC); chiral potental was used at resolution 18° in all the cases.
[b,c,d,]*See the corresponding footnotes to **Table 2**.

deviation of the native structure energy $\boldsymbol{E_N}$ from the mean energy of alternative structures $\boldsymbol{\langle E \rangle}$ expressed in the number of the root mean deviations of the threading energies from the mean, σ:

$$Z = (\langle \mathrm{E} \rangle - E_N) / \sigma \tag{27}$$

Table 2 shows that the fold-recognition quality is optimal at the resolution intervals of approx 0.5–1.0 Å both for long- and short-range potentials (except of sort-range *i,i + 4* interaction (S4), which shows the best accuracy at spacing approx 2–3 Å). One can see that the short-range potentials are more sensitive to the optimization of the resolution than the long-range ones.

Table 3 shows that the chiral energy term achieves its highest accuracy at the resolution range of 30–15°. (For further testing, the resolution of 18° was chosen as a compromise in average positioning, *Z*-score, and energy-gap characteriztics; it corresponds to 0.5–0.8 Å resolution interval for space coordinates.)

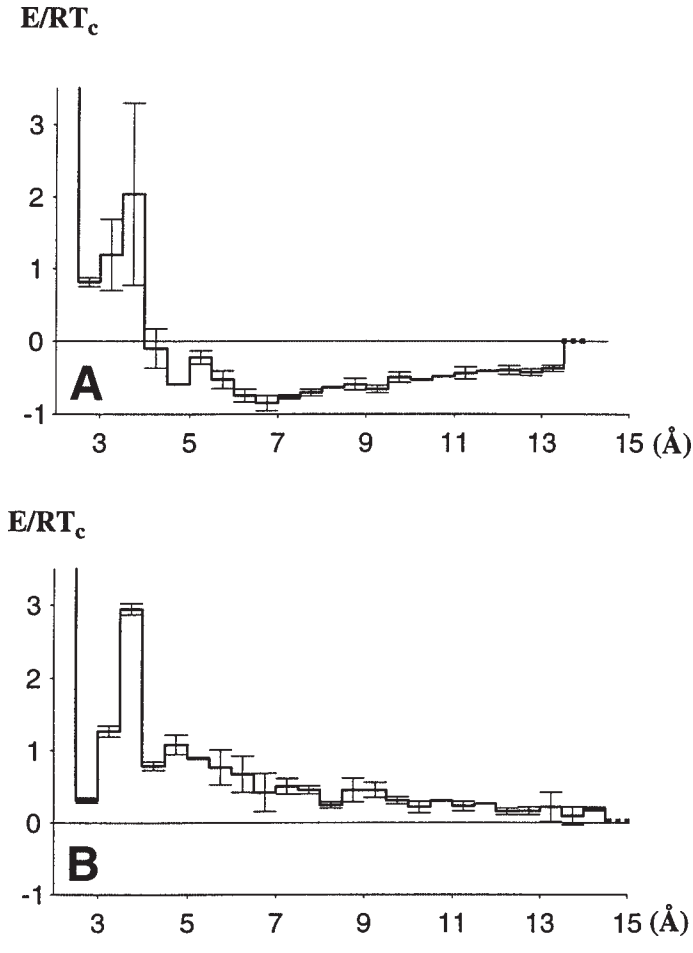


Fig. 3. Long-range potentials for **(A)** Phe–Leu and **(B)** Arg–Arg residue pairs derived from the database of 359 proteins at a resolution of 0.5 Å; inaccuracies of the potentials caused by limited statistics are shown by thin error bars; the estimates are done by **Eq. 26**; potential sets A and B were derived from the approximately equal databases of 180 and 179 proteins obtained by division of the original database of 359 proteins. Errors of the amplitude less than 0.05RTc are not shown. Long-range potentials are infinitely high at distances below 2.5 Å. The dots show that part of potential that is taken as zero at $r \geq R_\alpha + R_\beta$.

The results in **Tables 2** and **3** show that the main contribution to protein–structure recognition arises primarily from long-range interactions and chiral and bending energies. Recognition accuracy with different combinations of the local and long-range energies is presented in **Table 4**.

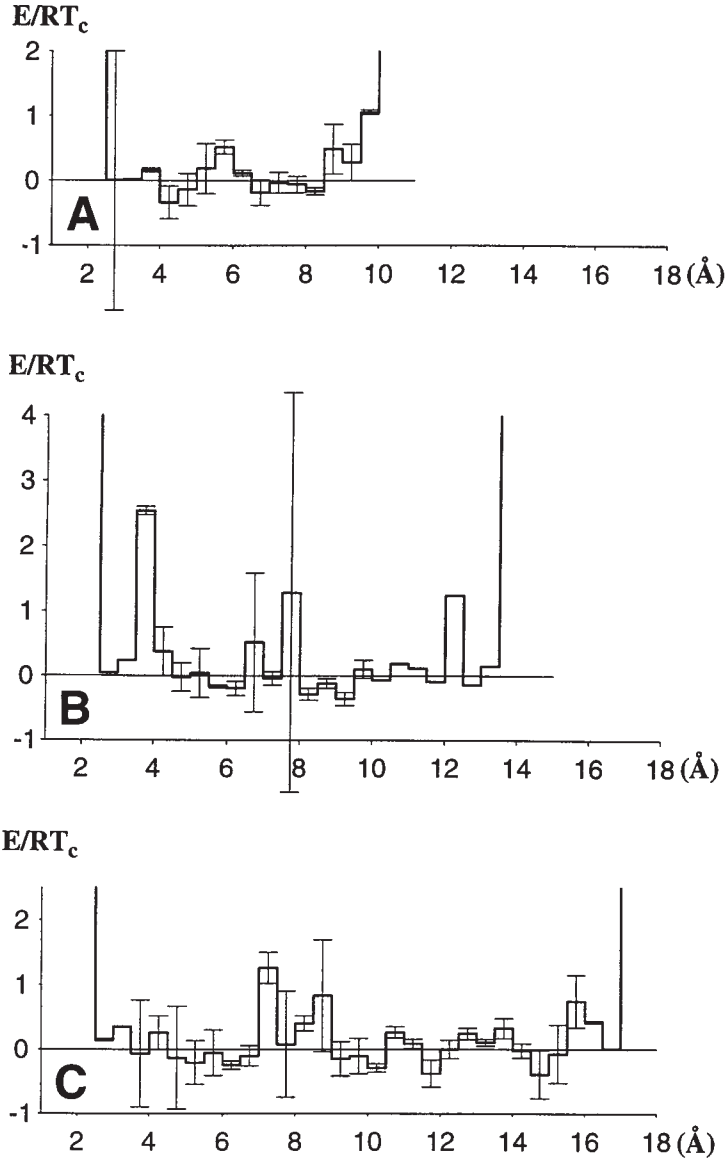


Fig. 4. Short-range distance dependent pairwise potentials. The potentials are given for Ala–Ser residue pair; they are derived from the database of 214 proteins at a resolution of 1 Å: (a), (b), and (c) correspond, respectively, to the *i,i + 2, i,i + 3, i,i + 4* types of short-range interactions (*see* **Fig. 1B**); statistical inaccuracy of the potentials is shown by error bars; errors of amplitude less than $0.05RT_c$ are not shown. The potentials are infinitely high at $r \leq R_{min} = 2.5$ Å and $r \geq R_{max} = 10.5$ Å, 13.5 Å, and 17 Å for, correspondingly, *i,i + 2, i,i + 3, i,i + 4* types of short-range interactions.

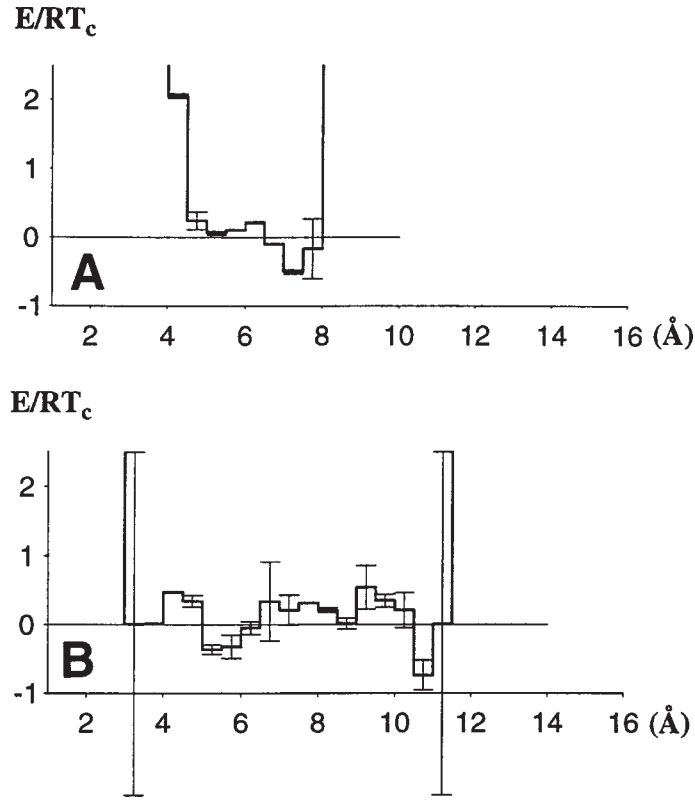


Fig. 5. Short-range bending potentials derived from the database of 359 proteins at a resolution of 0.5 Å: (a) and (b) corresponds, respectively, to Ser residue and to Ala-Ser residue pair occupying an intervening position (*see* **Fig. 1C**); error-bars show statistical inaccuracy of the potentials; errors of the amplitude less than $0.05RT_c$ are not shown. The potentials are infinitely high at $r \leq R_{min}$ and $r \geq R_{max}$; R_{min} = 4 Å and 3 Å; R_{max} = 8 Å and 11.5 Å for, correspondingly, $i,i + 2$ and $i,i + 3$ types of short-range bending interactions.

One can see that the total contribution of the local energy terms to the overall accuracy at the optimal resolution is slightly more than that of long-range potential.

Plots of typical potentials derived from the data set of 359 proteins at resolution of 0.5 Å is given in **Figs. 3–6**. One cannote a significant difference between long-range (**Fig. 3**) and short-range (**Figs. 4–6**) potentials. Long-range potentials change relatively smoothly with distance and, in essence, have one energy minimum for attractive (usually hydrophobic) residue pairs and no minimum for repelling pairs.

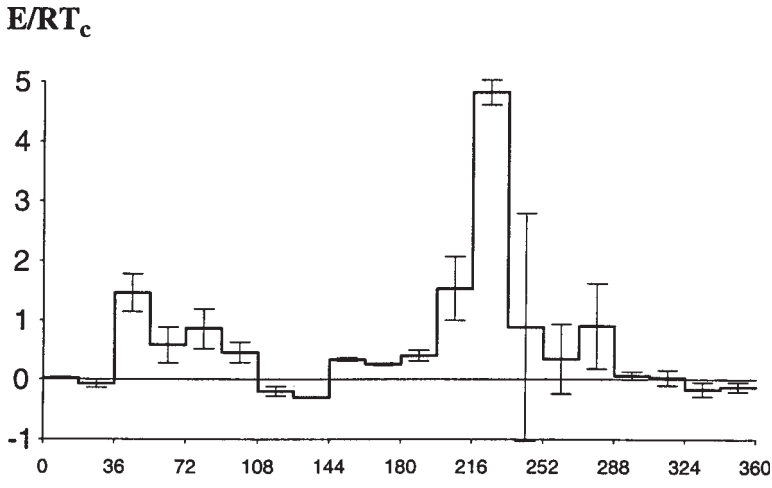


Fig. 6. Chiral potential for Val–Ala residue pair at resolution of 18°; error bars show statistical inaccuracy of the potential; errors of the amplitude less than $0.05RT_c$ are not shown.

Short-range potentials are characterized by more abrupt changes; they can have more than one local minimum, separated by barriers. Also, it is worth noting that because of the hard-core interatom repulsion, both long-range and short-range potential wells are bounded at R_{min} = 2.5 Å for C_β-based potentials and 3 Å for C_α ones; a prohibition of chain disruption restricts the maximal distance where the short-range potentials act.

The statistical error estimates, calculated by **Eq. 26** are shown in **Figs. 3–6** by the corresponding error bars. One can see differences in the amplitudes of the statistical errors, which are relatively small for the long-range interactions and sometimes rather significant for the short-range ones.

The detailed results of the threading experiment for 50 proteins with C_β–C_α combined potentials, derived at the resolution interval of 0.5 Å, are given in **Table 5**.

The potentials successfully recognize the native structure: 49 proteins were evaluated with the lowest energy for their native structures. Because all of the foregoing energy estimates are done with approximate energy functions, there is always a chance of finding a structure with lower energy than a given native one, considering more extensive ensembles of structures.

Table 5 shows large energy gaps between the native and competing folds for practically all the tested protein chains. However, it is important to stress that these gaps should depend on the number of tested alternatives: the more the alternatives, the less the gap. Because the energies of alternative structures have

virtually Gaussian distributions (**Fig. 2**), one can estimate the probability to find a structure with energy less than a given native one as

$$p(Z) = (1/\sqrt{2\pi}) \cdot \int_{Z}^{+\infty} e^{(t^2/2)}\, dt \tag{28}$$

where Z is a "Z-score" value. Thus, to find an energy lower than the energy of a given native structure, one needs to look through the following number of random structures:

$$N_Z = 1 / p(Z) \tag{29}$$

Having Z-score values obtained with C_β–C_α-based potentials, and assuming that structures obtained in threading give representative ensembles of misfolded protein-like structures, we found N_Z values for each of the 50 proteins tested. The geometric averaging of the N_Z values gives $\langle N_Z \rangle \sim 1.7 * 10^{11}$. Given an average chain length of 111 residues for 50 tested proteins, these numbers show that one can predict a protein fold only if the average number of possible backbone conformation per residue does not exceed $10^{11.23/111} = 1.26$. For globular folds where backbone conformations are not independent, this crucial number is not yet known (for a coil, where backbone conformations are independent, there are at least three conformations per residue: α_R, α_L, and β). Because the backbone conformations used for threading represent only a portion of the globular folds, and because they are not necessarily compact, the foregoing estimates indicate that our potentials are adequate for recognition of the native fold within some restricted set of folds, rather than for distinguishing the native fold from all possible folds.

4. Notes

1. To reduce biased errors in potentials derived from protein statistics, it is necessary to avoid similar proteins in the database of protein structure. Our experience shows that tests on sequence similarity alone are not absolutely sufficient for excluding all remote homologs. We found that pairs of proteins with RMSD (<10 Å) are good candidates for belonging to the same protein family.
2. To achieve the highest accuracy with potentials derived from a given database, it is necessary to optimize the size of the resolution interval in distance- (angle-) dependent energy functions to preserve as much detail as possible without introducing excessive error due to limited statistics.
3. The approach we use here for the derivation of energy functions differs from the other ones as it is directly based on the theory that explains the origin of Boltzmann statists in protein structure. However, the derived potentials have the same peculiarities ***(10)*** as correlation functions observed in liquids. They result from true direct interactions of residues in question and the indirect interactions of these residues via the surrounding liquids. One can neglect the indirect interactions when (1) direct interactions are strong (i.e., when distances are rather short),

Table 5
Characteristics of the Native Conformation Position in the Energy-Sorted List for 50 Proteins Obtained with C_β–C_α-Based Potentials Derived at a Resolution Interval of 0.5 Å

PDB code	Length	Threadings	Position[b]	Energy gap[c] in RT_c	Energy gap[c] in δ units	Z-score[d]	N_z[e]
1tgx.A	61	76258	1	2.5	0.1	4.2	$0.82*10^5$
1ptx	64	74841	1	42.2	2.2	6.9	$0.36*10^{12}$
1kve.B	77	70251	1	22.4	0.9	4.8	$0.10*10^7$
1cks.B	78	69898	1	67.8	2.4	5.9	$0.61*10^9$
1aav.A	85	67440	18	–17.2	–0.7	3.3	$0.17*10^4$
1ihf.B	94	64289	1	63.0	1.9	6.3	$0.54*10^{10}$
1who	94	64289	1	107.7	3.7	7.3	$0.98*10^{13}$
2hpe.A	99	62547	1	94.5	3.6	7.6	$0.76*10^{14}$
1lts.D	103	61158	1	71.2	2.2	6.1	$0.14*10^{10}$
1cmb.A	104	60812	1	42.9	1.1	4.7	$0.96*10^6$
1mhl.A	104	60812	1	38.8	1.2	5.0	$0.30*10^7$
1onc	103	61158	1	65.8	2.3	6.3	$0.85*10^{10}$
1kpt.A	105	60468	1	94.2	3.8	8.0	$0.14*10^{16}$
2psp.A	105	60468	1	149.9	4.7	8.7	$0.82*10^{18}$
1bri.C	107	59786	1	68.5	2.5	6.6	$0.49*10^{11}$
1fkj	107	59786	1	89.2	2.8	6.5	$0.24*10^{11}$
1cew.I	108	59446	1	62.9	1.8	5.8	$0.36*10^9$
1jpc	108	59446	1	138.5	4.8	9.0	$0.94*10^{19}$
1thx	108	59446	1	139.7	4.6	8.4	$0.61*10^{17}$
1bnd.A	109	59108	1	70.0	2.5	6.9	$0.32*10^{12}$
1jer	110	58773	1	67.1	2.4	6.8	$0.17*10^{12}$
1ccr	111	58439	1	76.3	2.2	6.1	$0.24*10^{10}$
1wad	111	58439	1	78.6	2.3	6.0	$0.12*10^{10}$
2tgi	112	58106	1	56.1	2.0	6.5	$0.34*10^{11}$
1dyn.A	113	57775	1	37.6	1.0	4.4	$0.19*10^6$
4rhn	115	57116	1	96.0	2.9	6.7	$0.84*10^{11}$

(*continued*)

and (2) when a number of residue types is big ***(10,18)***; the latter means that proteins consisting of 20 amino acid types should give more precise potentials than chain models of two to three residue types ***(3)***.

4. In estimating the role of simplified pairwise potentials for the protein-folding problem, one should not presume to explain with them all of the details of protein structure. However, these potentials can be useful for efficient discrimination of

Table 5 (continued)
Characteristics of the Native Conformation Position in the Energy-Sorted List for 50 Proteins Obtained with C_β–C_α-Based Potentials Derived at a Resolution Interval of 0.5 Å

PDB code	Length	Thread-ings	Posi-tion[b]	Energy gap[c] in RT_c	Energy gap[c] in δ units	Z-score[d]	N_z[e]
2rsl.B	120	55475	1	141.7	3.7	7.1	$0.13*10^{13}$
2pfl	121	55147	1	59.7	1.9	6.1	$0.23*10^{10}$
1reg.X	122	54820	1	132.5	3.4	7.2	$0.27*10^{13}$
1whi	122	54820	1	139.9	4.0	7.7	$0.17*10^{15}$
1bp2	123	54494	1	109.0	3.3	7.5	$0.27*10^{14}$
1bur.T	123	54494	1	137.7	4.3	8.3	$0.21*10^{17}$
1msp.A	124	54171	1	77.0	2.1	6.3	$0.86*10^{10}$
1zia	124	54171	1	128.6	3.5	7.9	$0.54*10^{15}$
4fgf	124	54171	1	58.0	1.7	5.7	$0.20*10^{9}$
7rsa	124	54171	1	94.8	2.9	6.8	$0.25*10^{12}$
1otg.A	125	53850	1	115.7	3.3	6.9	$0.29*10^{12}$
1oun.A	125	53850	1	122.0	3.5	7.3	$0.92*10^{13}$
2phy	125	53850	1	134.9	4.1	8.0	$0.15*10^{16}$
1rie	127	53219	1	130.5	3.6	7.6	$0.83*10^{14}$
1ttb.A	127	53219	1	116.9	3.5	7.2	$0.40*10^{13}$
1doi	128	52905	1	126.1	3.2	6.6	$0.58*10^{11}$
3chy	128	52905	1	171.9	4.0	7.5	$0.28*10^{14}$
1cpq	129	52593	1	80.0	1.7	5.3	$0.15*10^{8}$
1msc	129	52593	1	22.4	0.7	5.6	$0.97*10^{8}$
2aza.A	129	52593	1	92.0	2.8	6.7	$0.11*10^{12}$
1lzr	130	52283	1	140.6	4.3	8.5	$0.14*10^{18}$
1lid	131	51976	1	93.5	2.9	6.6	$0.36*10^{11}$
1lis	131	51976	1	88.8	2.7	7.0	$0.70*10^{12}$
1lit	131	51976	1	125.6	3.9	8.1	$0.25*10^{16}$
Avg.	111	57736	1.06	89.3	2.7	6.8	$0.17*10^{12}$

[b,c,d]See footnotes to **Table 2**.
[e]N_z values are defined in **Eqs. 29** and **30**; the average N_z is defined as the geometrical mean.

the tiny fraction of most favorable conformations from the vast majority of the other ones.

References

1. Sippl, M. (1990) Calculation of conformational ensembles from potentials of mean force. An approach to the knowledge-based prediction of local structures in globular proteins. *J. Mol. Biol.* **213,** 859–883.

2. Vajda, S., Sippl, M., and Novotny, J. (1997) Empirical potentials and functions for protein folding and binding. *Curr. Opin. Struct. Biol.* **7(2),** 222–228.
3. Thomas, P. D. and Dill, K. A. (1996) Statistical potentials extracted from protein structures: how accurate are they? *J. Mol. Biol.* **257,** 457–469.
4. Godzik, A., Kolinski, A., and Skolnick, J. (1995) Are proteins ideal mixtures of amino acids? Analysis of energy parameter sets. *Prot. Sci.* **4,** 2107–2117.
5. Jernigan, R. and Bahar, I. (1996) Structure-derived potentials and protein simulations. *Curr. Opin. Struct. Biol.* **6,** 195–209.
6. Rooman, J. and Wodak, S. (1995) Are database-derived potentials valid for scoring both forward and inverted protein folding? *Protein Eng.* **8,** 849–858.
7. Skolnick, J., Jaroszewski, L., Kolinski, A., and Godzik, A. (1997) Derivation and testing of pair potentials for protein folding. When is the quasichemical approximation correct? *Prot. Sci.* **6(3),** 676–688.
8. Kocher, J. P., Rooman M. J., and Wodak S. J. (1994) Factors influencing the ability of knowledge-based potentials to identify native sequence–structure matches. *J. Mol. Biol.* **235(5),** 1598–1613.
9. Pohl, F. M. (1971) Empirical protein energy maps. *Nat. New Biol.* **234,** 277–279.
10. Finkelstein, A., Badretdinov, A., and Gutin, A. (1995) Why do protein architectures have Boltzmann-like statistics? *Proteins* **23,** 142–150.
11. Reva, B. A., Finkelstein, A. V., Sanner, M. F., and Olson, A. J. (1997) Accurate mean-force pairwise-residue potentials for discrimination of protein folds, in *Proceedings of Pacific Symposium on Biomolecular Computations*, 373–384.
12. Reva, B. A., Finkelstein, A. V., Sanner, M. F., and Olson, A. J. (1997) Residue-residue mean-force potentials for protein structure recognition. *Protein Eng*. **10(5),** 865–876.
13. Hobohm, U., Scharf, M., Schneider, R., and Sander, C. (1992) Selection of representative protein data sets. *Prot. Sci.* **1,** 409–417.
14. Smith, T. and Waterman, M. (1981) Identification of common molecular subsequences. *J. Mol. Biol.* **147,** 195–197.
15. Murzin, A., Brenner, S., Hubbard, T., and Chothia, C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247,** 536–540.
16. Ben-Naim, A. (1997) Statistical potentials extracted from protein structures: are these meaningful potentials? *J. Chem. Phys.* **107(9),** 3698–3706.
17. Hendlich, M., Lackner, P., Weitckus, S., Floeckner, H., Froschauer, R., Gottsbacher, K., Casari, G., and Sippl., M. (1990) Identification of native protein folds amongst a large number of incorrect models. The calculation of low energy conformations from potentials of mean force. *J. Mol. Biol.* **216,** 167–180.
18. Shakhnovich, E. I. and Gutin, A. M. (1989) Formation of unique structure in polypeptide chain. Theoretical investigation with the aid of a replica approach. *Biophys. Chem.* **34,** 187–199.

9

Genetic Algorithms and Protein Folding

Steffen Schulze-Kremer

1. Introduction

Genetic algorithms are, like neural networks, an example *par excellence* of an information-processing paradigm that was originally developed and exhibited by nature and later discovered by humans, who subsequently transformed the general principle into computational algorithms to be put to work in computers. Nature uses the principle of genetic heritage and evolution in an impressive way. Application of the simple concept of performance based reproduction of individuals ("survival of the fittest") led to the rise of well-adapted organisms that can endure in a potentially adverse environment. Mutually beneficial interdependencies, cooperation, and even apparently altruistic behavior can emerge solely by evolution. The investigation of those phenomena is part of research in artificial life but is not dealt with here.

Evolutionary computation comprises the four main areas of genetic algorithms ***(1)***, evolution strategies ***(2)***, genetic programming ***(3)***, and simulated annealing ***(4)***. Genetic algorithms and evolution strategies emerged at about the same time in the United States and Germany. Both techniques model the natural evolution process in order to optimize either a fitness function (evolution strategies) or the effort of generating subsequent, well-adapted individuals in successive generations (genetic algorithms). Evolution strategies in their original form were basically stochastic hill-climbing algorithms and were used for optimizing complex, multiparameter objective functions that, in practice, cannot be treated analytically. Genetic algorithms in their original form were not primarily designed for function optimization but rather to demonstrate the efficiency of genetic crossover in assembling successful candidates over complicated search spaces. Genetic programming takes the idea of solving an optimization problem by evolution of potential candidates one step further in that

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

not only the parameters of a problem but also the structure of a solution is subject to evolutionary change. Simulated annealing is mathematically similar to evolution strategies. It was originally derived from a physical model of crystallization. Only two individuals compete for the highest rank according to a fitness function and the decision about accepting suboptimal candidates is controlled stochastically.

The methods presented in this chapter are heuristic, i.e., they contain a random component. As a consequence (and in contrast to deterministic methods), it can never be guaranteed that the algorithm will find an optimal solution or even any solution at all. Evolutionary algorithms are therefore used preferably for applications where deterministic or analytic methods fail, e.g., because the underlying mathematical model is not well defined or the search space is too large for systematic, complete search (*np* completeness). Another application area for evolutionary algorithms that is rapidly growing is the simulation of living systems starting with single cells and proceeding to organisms, societies, or even whole economic systems ***(5,6)***.

Work with evolutionary algorithms bears the potential for a philosophically and epistemologically interesting recursion. At the beginning, evolution emerged spontaneously in nature. Next, humans discovered the principle of evolution and acquires knowledge of its mathematical properties. He ("re-") defines genetic algorithms for computers. To complete the recursive cycle, computational genetic algorithms are applied to the very objects (DNA, proteins) of which they had been derived in the beginning. A practical example of such a meta-recursive application is given in the sections on protein folding. **Figure 1** illustrates this interplay of natural and simulated evolution.

2. Genetic Algorithms

The so-called genetic algorithm ***(7)*** is a heuristic method that operates on pieces of information like nature does on genes in the course of evolution. Individuals are represented by a linear string of letters of an alphabet (in nature, nucleotides, in genetic algorithms, bits, characters, strings, numbers, or other data structures), and they are allowed to *mutate*, *crossover*, and *reproduce*. All individuals of one generation are evaluated by a *fitness function*. Depending on the generation replacement mode, a subset of parents and offspring enters the next reproduction cycle. After a number of iterations the population consists of individuals that are well adapted in terms of the fitness function. Although this setting is reminiscent of a classical function optimization problem, genetic algorithms were originally designed to demonstrate the benefit of genetic crossover in an evolutionary scenario, not for function optimization. It cannot be proven that the individuals of a final generation contain an optimal solution for the objective encoded in the fitness function but it can be shown

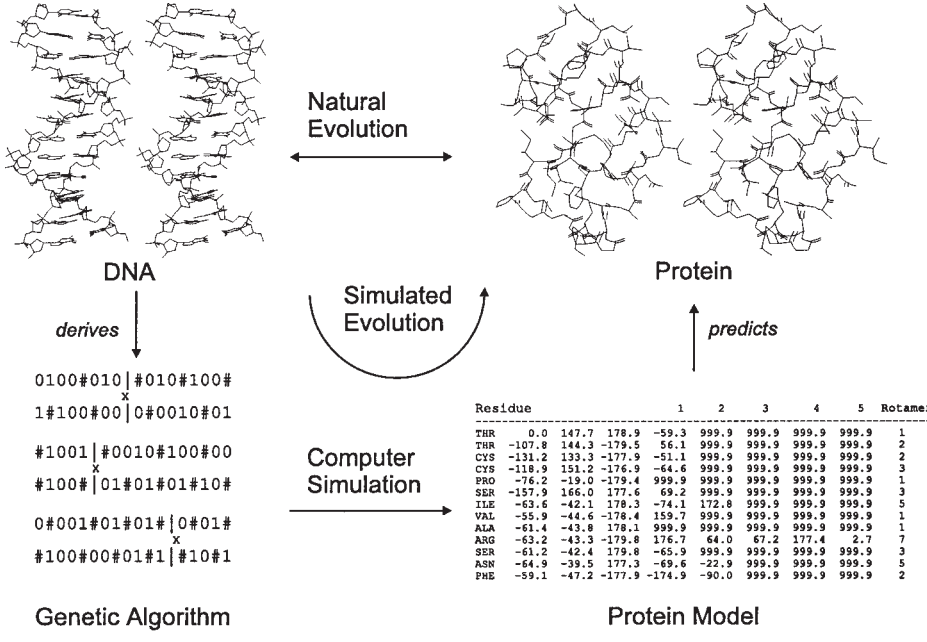


Fig. 1. Interplay of natural and simulated evolution. Responding to nature's challenges by her own means. The principle of evolution can be used to model structures and to stimulate biomolecular reactions.

mathematically that the genetic algorithm optimizes the effort of testing and producing new individuals if their representation permits development of building blocks (also called schemata). In that case, the genetic algorithm is driven by an implicit parallelism and generates significantly more successful progeny than random search. In a number of applications where the search space was too large for other heuristic methods, or too complex for analytic treatment, genetic algorithms produced favorable results *(8)*.

2.1. Basic Algorithm

The basic outline of a genetic algorithm is as follows:

1. *Initialize a population of individuals.* This can be done either randomly or with domain-specific background knowledge to start the search with promising seed individuals. Where available the latter is always recommended.
   a. Individuals are represented as a string of bits. This is not a restriction for the type of problem because other data types (numbers, strings, structures) can also be encoded as bit strings.

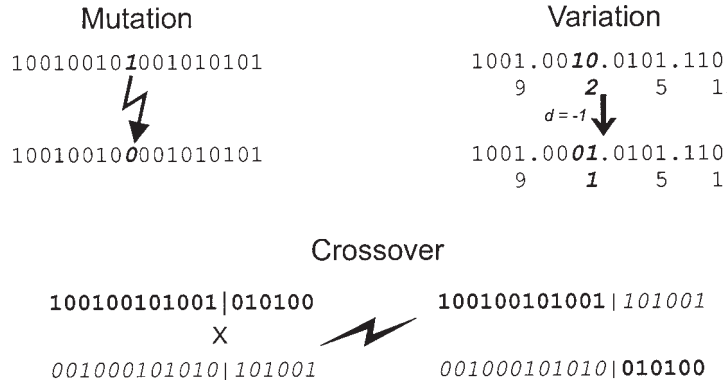


Fig. 2. Genetic operators for the genetic algorithm. Mutation exchanges one single bit. Variation modifies the encoded value by a small increment (or decrement). Crossover (single-point) exchanges a continuous fragment of an individual. Analogously, more than one crossover point can be selected and only the fragments between those positions are then exchanged (two-point crossover for two crossover points; uniform crossover for as many crossover sites as positions in the individual).

   b. A fitness function must be defined that takes as input an individual and returns a number (or a vector) that can be used as a measure of the quality (fitness) of that individual.
   c. The application should be formulated such that the desired solution to the problem coincides with the most successful individual according to the fitness function.
2. *Evaluate all individuals* of the arrent population.
3. *Generate new individuals.* The reproduction probability for an individual is proportional to its relative fitness within the current generation. Reproduction involves domain-specific genetic operators (*see* **Fig. 2**). Operations to produce new individuals are:
   a. *Mutation.* Substitute one or more bits of an individual randomly by a new value (0 or 1).
   b. *Variation.* Change the bits in a way that the number encoded by them is slightly incremented or decremented.
   c. *Crossover.* Exchange parts (single bits or strings of bits) of one individual with the corresponding parts of another individual. Originally, only one-point crossover was performed, but theoretically one can process up to $L - 1$ different crossover sites (with L as the length of the individual). For one-point crossover, two individuals are aligned and one location on their strings is randomly chosen as the crossover site. Now the parts from the beginning of the individuals to the crossover site are exchanged between them. The resulting hybrid individuals are taken as the new offspring individuals.
4. *Select individuals* for the new parent generation.

a. In the original genetic algorithm, all offspring were selected and all parents were discarded. This is motivated by the biological model and is called total generation replacement.
b. More recent variations of generation replacement compare the original parent individuals and the offspring which are then ranked by their fitness values. Only the n best individuals (n is the population size, i.e., the number of individuals in one generation) are taken into the next generation. This method is called elitist generation replacement. It guarantees that good individuals are not lost during a run. With total generation replacement good individuals may "die out" because they produce only offspring inferior in terms of the fitness function. Another variant is steady-state replacement. There, two individuals are randomly selected from the current population. The genetic operators are applied and the offspring replace the parents in the population. Steady-state replacement often converges sooner because on average it requires fewer fitness evaluations than elitist or total generation replacement.

5. Go back to **step 2** until either a desired fitness value is reached or until a predefined number of iterations is performed.

2.2. Schemata Theorem

The mathematical foundation of genetic algorithms is the schemata theorem of J. H. Holland ***(1)***. It makes a statement about the propagation of schemata (or building blocks) within all individuals of one generation. A schema is implicitly contained in an individual. Like individuals, schemata consist of bit strings (1, 0) and can be as long as the individual itself. In addition, schemata may contain "don't care" positions where it is not specified whether the bit is 1 or 0, i.e., schemata H_i are made from the alphabet {1, 0, #}. In other words, a schema is a generalization of (parts of) an individual. For example, the individuals:

```
10010010010000011110100 0101
```

and

```
01001001010000011111101 0101
```

can be summarized by the schema:

```
##0##0##01000001111#10#0101
```

where all identical positions are retained and differing positions marked with a "#," which stands for "don't care." The length $\delta(H)$ of the above schema is 25, which is the distance from the first to the last fixed symbol (1 or 0 but not #). The total number of different schemata of length l over an alphabet of cardinality k is $(k + 1)^l$. Because each string of length l contains 2^l schemata, with n individuals in one generation, there are between 2^l and $n2^l$ schemata in the population (depending on the similarity of the n individuals). The *order* of a schema $o(H)$ is the number of fixed positions (1 or 0 but not #).

Let $s(H, t)$ ***(9)*** be the number of occurrences of a particular schema H in a population of n individuals at time t. The bit string A_i of individual i then gets selected for reproduction with probability p_i :

$$p_i = f_i / \sum_{j=1}^{n} f_j$$

where f_i is the fitness value of the jth individual. The expected number of occurrences of schema H at time $t + 1$ is:

$$s(H, t + 1) = s(H,t) \cdot n \cdot \bar{f}(H) / \sum_{j=1}^{n} f_j$$

with $\bar{f}(H)$ as the average fitness of all individuals (strings A_i) that contain H. Crossover and mutation operators can destroy schemata during reproduction. The longer a single individual, the smaller the probability that a schema H will be involved in a crossover event. The longer a schema $\delta(H)$, the more likely is its destruction through recombination with another individual. Hence, for crossover the lower bound for the survival probability of a schema H is:

$$p_s \geq 1 - \delta(H) / (L - 1)$$

with L as the length of one whole individual. If we perform crossover stochastically at a frequency p_c the survival probability p_s becomes

$$p_s \geq 1 - p_c\, \delta(H) / (L - 1)$$

Summarizing the effects of independent crossover and reproduction, we arrive at the following equation for the expected occurrence of a schema H at time $t + 1$:

$$s(H, t + 1) = s(H,t) \cdot n \cdot \frac{\bar{f}(H)}{\sum_{j=1}^{n} f_j} \cdot \left(1 - p_c \frac{\delta(H)}{L - 1}\right)$$

This equation tells us that schemata increase over time proportional to their relative fitness and inversely proportional to their length. Mutation can effect a schema H at each of its $o(H)$ fixed positions with mutation probability p_m. Survival of a single constant position in a schema is then $p_s = 1 - p_m$ and survival of the entire schema:

$$p_s = (1 - p_m)^{o(H)}$$

which for small p_m, can be approximated by $p_s \approx 1 - o(H) \cdot p_m$. Summarizing the effects of independent mutation, crossover, and variation, we get the following equation for the expected count of a schema H:

$$s(H, t + 1) = s(H,t) \cdot n \cdot \frac{\bar{f}(H)}{\sum_{j=1}^{n} f_j} \cdot \left(1 - p_c \frac{\delta(H)}{L - 1} - o(H) \cdot p_m\right)$$

Table 1
Genetic Algorithm at Work (Part I)[a]

Individual	Bit String	Integer Value	Fitness $f(i) = i^2$	Reproduction Probability $f(i)/\sum f$	Expected Count $f_i/\bar{f}$	Actual Count (Roulette wheel)
1	01010	10	100	8.2%	0.33	1
2	10101	21	441	36.1%	1.45	1
3	00010	2	4	0.3%	0.01	0
4	11010	26	676	55.4%	2.22	2
Sum			1221	100.0%	4.01	4
Average			305.25			
Max			676			

Schemata	Pattern	In individual	Average schema fitness
H_1	`00###`	3	4
H_2	`1####`	2,4	558.5
H_3	`#1#1#`	1	100

[a]Continued in **Table 2**. See main text for explanation.

Assuming that a schema H could always outperform other schemata by a fraction of the total mean fitness, this equation can be rewritten as:

$$s(H, t+1) = s(H,t) \cdot n \cdot \frac{\frac{1}{n}\sum_{j=1}^{n} f_i + b\frac{1}{n}\sum_{j=1}^{n} f_j}{\frac{1}{n}\sum_{j=1}^{n} f_i} \cdot \left(1 - p_c \frac{\delta(H)}{L-1} - o(H) \cdot p_m\right)$$

$$= s(H,t) \cdot (1+b) \cdot \left(1 - p_c \frac{\delta(H)}{L-1} - o(H) \cdot p_m\right)$$

This equation is of the form $f_k = f_0 \cdot (1+b)^k \cdot g(p_c, p_m, L, \delta(H))$, which says that the number of schemata better than average will exponentially increase over time. Effectively, many different schemata are sampled implicitly in parallel and good schemata will persist and grow. This is the basic rationale behind the genetic algorithm. It is suggested that if the (linear) representation of a problem allows the formation of schemata then the genetic algorithm can efficiently produce individuals that continuously improve in terms of the fitness function.

2.3. Handworked Example

Let us examine the performance of the genetic algorithm on a simple application which is the search for the largest square product of a five bit integer. **Table 1** shows four initial individuals that were randomly generated.

Table 2
Genetic Algorithm at Work (Part II)

Mating pool after reproduction with crossover site	Mating partners	Crossover site	New population	Integer value	Fitness value $f(i) = i^2$
010|10	3	3	01010	10	100
10|101	4	2	10010	18	324
110|10	1	3	11010	26	676
11|010	2	2	11101	29	841
Sum					1941
Average					485.25
Max					841
Schemata	After reproduction		After crossover		
Expected Count $\sum f(H) / \bar{f}$	Actual count	In individual	New count expected	Actual count	In individual
0.01	0	—	0.00	0	—
1.83	3	2, 3, 4	2.54	3	2, 3, 4
0.33	3	1, 3, 4	1.60	2	1, 3

The bit strings of the individuals are decoded to unsigned integer values. The fitness function f $(i) = i^2$ is used to assign a fitness value to each individual. Depending on their relative fitness values, the reproduction probability (between 0% and 100%) for each individual is calculated and converted into the number of expected successors. Then the so-called roulette wheel algorithm is used to perform a stochastic selection based on the reproduction probability. Three particular schemata and their occurrence and distribution over the four individuals is monitored.

Table 2 shows the situation after reproduction. The individuals selected for reproduction have been replicated according to their relative fitness. Crossover sites and mating partners have been assigned randomly. To keep this example simple, mutation is not used here. After performing crossover, the new fitness values of the individuals in the new population are calculated. The performance of the three schemata H_1, H_2, and H_3 is also shown. Schema H_1 is of low fitness because it implies that the decoded integer is smaller than 8. Therefore, this schema gets only a small chance for reproduction. Actually, H_1 dies out as its only parent (the original individual no. 3) does not get selected for reproduction. Schemata H_2 and H_3 both have a reasonable chance for reproduction and are subsequently found in the new generation. Both average and best fitness values have significantly improved in the new generation.

In this example, we monitored only three schemata. There are, however, between $2^5 = 32$ and $4.2^5 = 128$ schemata in this small population that were all implicitly evaluated in the same manner in parallel only at the small computational cost of copying and exchanging a few bit strings. The implicit arithmetics of finding and promoting the best schemata do not actually have to be carried out by the computer. They are, so to speak, side effects of the genetic paradigm. This implicit parallelism is the basic reason for the efficiency of genetic algorithms.

As an addendum, the stochastic universal sampling algorithm for minimization of fitness values by J. E. Baker is implemented as follows in C programming language. This implementation is especially elegant because the source code is quite short and the generation of only one random number between 0 and 1 is needed to perform a random selection among all individuals in one generation according to their individual fitness values.

```
k = 0;     /* k is an integer index of next individual to be selected */
ptr = Rand();                /* spin the roulette wheel; 0 < ptr < 1 */
Scaling_Factor = 1.0 / (Prev_Worst_Fitness - Average_Current_Fitness);
for (Sum = i = 0; i < Popsize; i++)  /* Popsize is size of population */
{                 /* Fitness[i] is the fitness value of individual i */
if (Fitness[i] < Prev_Worst_Fitness)
 Expected_Count = (Prev_Worst_Fitness - Fitness[i]) * Scaling_Factor;
else Expected_Count = 0.0;
for (Sum + = Expected_Count; Sum > ptr; ptr++)
       sample[k++] = i; /* sample is an array of "Popsize" integers. */
}          /* The value of each array element defines an individual. */
```

These instructions fill the array `sample` with integer values in a way that each individual with a fitness better than the average fitness of the current generation *and* better than the worst fitness in the last generation gets a chance of replication proportional to its fitness. Note the concerted increments of `Sum` and `ptr` in the inner `for`-loop.

3. Protein-Folding Application

3.1. 2D Protein Model

R. Unger and J. Moult have used a 2D, simplified protein model to demonstrate the usefulness of a genetic algorithm in the search for minimal energy conformations ***(10)***. Their protein model has only two kinds of amino acid residues: hydrophobic (black circles) and hydrophilic ones (white circles) (*see* **Fig. 3**). The "protein" is a chain of these two types of residues on a 2D, orthogonal grid. Bond angles are restricted to the values 0°, 90°, 180°, and 270°. The "force field" to determine the inner energy of a fold is defined to be the sum of all hydrophobic interactions. A hydrophobic interaction contributes –1 energy units when two noncovalently bonded hydrophobic residues come to lie

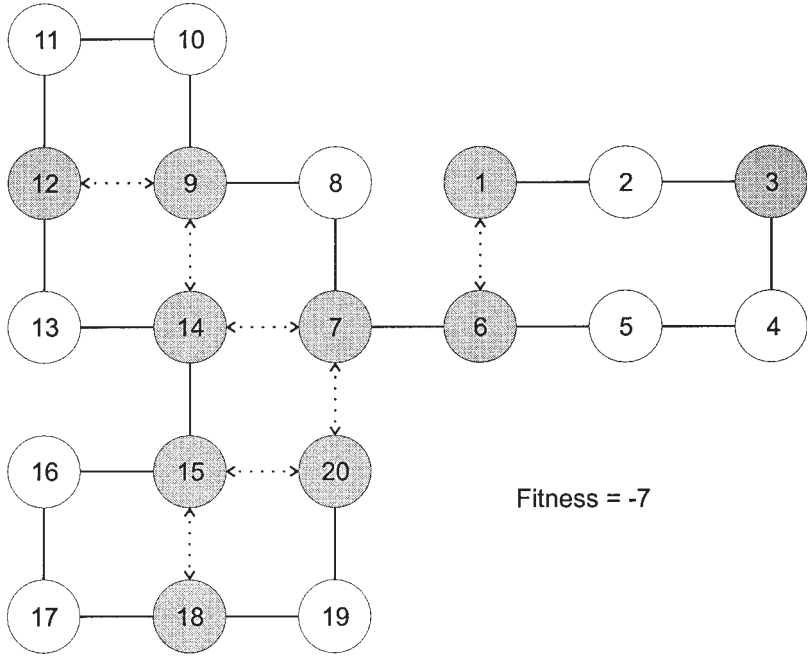


Fig. 3. Simplified 2D protein model. Each hydrophobic interaction between two adjacent hydrophobic residues (black circles) contributes –1 energy units. There are seven such interactions (indicated by dotted arrows) in this molecule.

orthogonally adjacent to each other. For example, the "molecule" in **Fig. 3** has seven hydrophobic interactions and hence an "energy" of –7 units.

This section presents the implementation of a simplified 2D protein model with a hydrophobic energy function for use with the public domain genetic algorithms software GENESIS ***(11)***. GENESIS version 5.0 was written to promote the study and application of genetic algorithms for function minimization. Because genetic algorithms are task-independent optimizers, the user must provide only a domain-specific crossover operator and an evaluation function that returns a fitness value for each individual. The GENESIS system was written by J. J. Grefenstette in the programming language C and is available on the Internet.

To represent the conformation of a 2D protein the following definitions are used. The primary structure is a list of bits 0 and 1, where 0 stands for a hydrophilic residue and 1 for a hydrophobic one (*see* the constant SEQUENCE in the example code below). The conformation is described by a bit string where every two bits in sequence define the bond angle between the current residue and the next one in sequence. **Figure 4** explains the meaning of the four directional codes 00, 01, 10, and 11 for a bond angle.

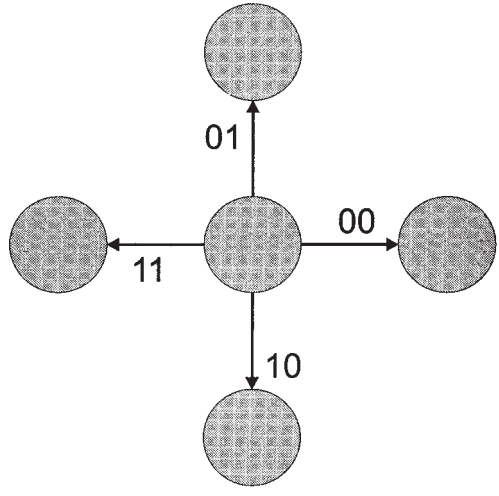


Fig. 4. Encoding conformation and direction. A positional code determines the location of the next residue, whether it is up (01), down (10), to the left (11), or to the right (00). Note: in analogy to Gray codes in this representation a change of one bit in the positional code implies an increment of 90° (or –90°) on the bond angle.

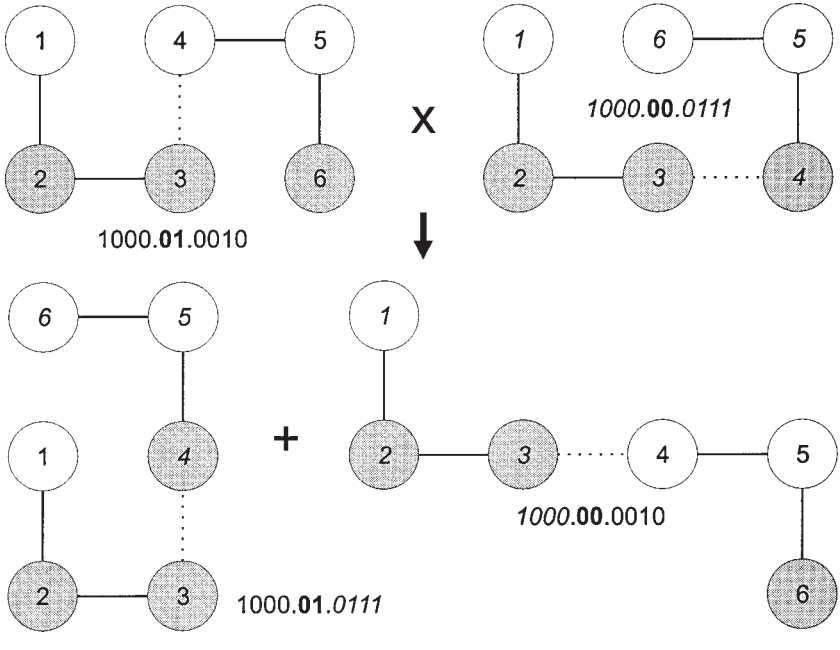


Fig. 5. Crossover for 2D proteins. This one-point crossover site is selected to minimize the overlaps between residues of the newly joined fragments.

Crossover is the prominent operator in genetic algorithms. For 2D proteins it can be implemented as follows (*see* **Fig. 5**). One bond (except the first in a molecule) is randomly selected. The fragments to the left and right of that bond (crossover site) are exchanged between two individuals. The new value for the bond at the crossover site is selected such that the overlap of the two fragments being joined becomes minimal. Depending on the structure of the fragments, some overlap in the newly created individuals cannot always be avoided. However, individuals with overlaps die out quickly because they get high (i.e., bad) fitness values (*see* function COUNT_OVERLAPS below). The crossover probability is set to 60% for every pair of individuals.

The following C-source code is the complete implementation for crossover of 2D proteins. The file with this code must be named CROSS.C and compiled and linked with the remaining Genesis files.

```
/* FILE CROSS.C */

#define SEQ_LENGTH 20                  /* 2D protein with 20 residues */
#define STR_LENGTH     ((SEQ_LENGTH - 1) * 2) /* 38 bits for 19 bonds */
#define X              ((SEQ_LENGTH + 1) * 2) /* max X grid extension */
#define Y              ((SEQ_LENGTH + 1) * 2) /* max Y grid extension */

#include "extern.h"

typedef struct choice {
    char piece[STR_LENGTH + 1];
    double overlaps;
}  choice;
```

The function SELECT_MIN_OVERLAP determines the one individual with the smallest number of overlapping residues out of n choices (n = UPPER − LOWER; UPPER and LOWER are indices of individuals in the current population). If two or more conformations have the same (smallest) number of overlaps, one of them is chosen at random.

```
int select_min_overlap (choice *choices, int lower, int upper)
{
  int i, j, best(4);
  double tmp = 10e10;

  for(i = lower; i < upper; i++)
    if (choices[i].overlaps < tmp)
      tmp = choices[i].overlaps;

  j = 0;
  for(i = lower; i < upper; i++)
    if (choices[i].overlaps = = tmp)
      best[j++] = i;
  return (best[Randint (0, j -1)]);
}
```

COUNT_OVERLAPS calculates the number of overlapping residues in one individual.

```
double count_overlaps (char *str)
{
  int i, j;
  int x = X / 2;
  int y = Y / 2;
  double sum = 0.0;
  static int matrix[X][Y];
  for (i = 0; i < X; i++)    /* initializing arrays */
    for (j = 0; j < Y; j++)
     matrix[i][j] = 0;
  for (i = 0; i < SEQ_LENGTH; i++) /* filling arrays */
  {
    matrix[x][y]++;
    if (str[2*i] == '0')
     if (str[2*i + 1] == '0')
            x += 1;
     else
            y += 1;
     else
            if (str[2*i + 1] == '0')
                   y -= 1;
            else
                   x -= 1;
  }
  for (i = 0; i < X; i++)    /* summation */
    for (j = 0; j < Y; j++)
     if (matrix[i][j] > 1)
            sum += matrix[i][j] - 1;
  return sum;
}
```

Next, the source code of the actual CROSSOVER function is presented. CROSSOVER selects a pair of individuals, finds a crossover site, and exchanges the fragments to the left and right between the two individuals. The length of individuals in this application is set to 40, although there are only 19 bonds and thus 38 bits required. The extra bond allows CROSSOVER to modify the last bond in the molecule.

```
Crossover()
{
  register int mom, dad;            /* participants in the crossover */
  register int xpoint;     /* first crossover point w.r.t. structure */
  register int i, j, k;                     /* loop control variable */
  static int last;              /* last element to undergo Crossover */
  int boy, gal;               /* set if parents differ from offspring */
  static int firstflag = 1;
```

```
static char i1[STR_LENGTH + 1], i2[STR_LENGTH + 1];
static char piece1a[STR_LENGTH + 1],piece1b[STR_LENGTH + 1];
static char piece2a[STR_LENGTH + 1],piece2b[STR_LENGTH + 1];
static choice choices(8);

Trace("Crossover entered");
Dtrace("crossover");

if (firstflag)
{
  last = (C_rate*Popsize*Gapsize) - 0.5 ;
  firstflag = 0;
}

for (mom = 0; mom < last ; mom += 2)
{
  dad = mom + 1; /* kids start as identical copies of parents */
  Unpack (New[mom].Gene, i1, Bytes);
  Unpack (New[dad].Gene, i2, Bytes);

  xpoint = 2 * Randint (1, SEQ_LENGTH - 1);/* crossover point */
  strncpy (piece1a, i1, xpoint);
  piece1a[xpoint] = '\0';
  strcpy (piece1b, i1 + xpoint);
  strncpy (piece2a, i2, xpoint);
  piece2a[xpoint] = '\0';
  strcpy (piece2b, i2 + xpoint);

  strcpy (choices[0].piece, piece1a);
  strcat (choices[0].piece, piece2b);
  for (j = 1; i < 4; i++)
   strcpy (choices[i].piece, choices[0].piece);
  strcpy (choices(4).piece, piece2a);
  strcat (choices(4).piece, piece1b);
  for (i = 5; i < 8; i++)
   strcpy (choices[i].piece, choices(4).piece);
```

The foregoing source code performs a crossover at the site XPOINT between the two individuals I1 and I2 and makes four copies of each of the two resulting hybrid individuals. In the following FOR-loop the four choices (00, 01, 10, 11) for the new crossover site are set up. Each bit pair corresponds to one of the four bond angles (0°, 90°, 270°, and 180°) between two fragments. Now the overlap of the hybrid fragments joined by each of the four bond angles can be calculated.

```
for (k = 0; k < 2; k++)
 for (i = 0; i < 2; i++)
        for (j = 0; j < 2; j++)
        {
                choices[k*4 + i*2 + j].piece[xpoint] = i + '0';
```

```
                    choices[k*4 + i*2 + j].piece[xpoint + 1] = j + '0';
               }
      for (i = 0; i < 8; i++)
       choices[i].overlaps = count_overlaps (choices[i].piece);
```

Finally, the conformations with the smallest overlap are selected to replace their parents.

```
      gal = select_min_overlap (choices, 0, 4);
      boy = select_min_overlap (choices, 4, 8);
      Pack (choices[gal].piece, New[mom].Gene, STR_LENGTH);
      Pack (choices[boy].piece, New[dad].Gene, STR_LENGTH);

      if   ((strcmp (choices[gal].piece, i1) ! = 0) ||
           (strcmp (choices[boy].piece, i2) ! = 0))
      {
           New[mom].Needs_evaluation = 1;
           New[dad].Needs_evaluation = 1;
      }
    }
    Trace("Crossover completed");
}
```

Mutation is applied with a probability of 0.1%. The remaining parameters for this application in GENESIS are listed as follows:

```
          Experiments = 1
         Total Trials = 10000
      Population Size = 50
     Structure Length = 40
       Crossover Rate = 0.6
        Mutation Rate = 0.001
       Generation Gap = 1.0
       Scaling Window = 5
      Report Interval = 10
     Structures Saved = 10
    Max Gens w/o Eval = 4
        Dump Interval = 10
          Dumps Saved = 0
              Options = celD
          Random Seed = 123456789
             Rank Min = 0.75
```

The fitness function is quite elementary. Only orthogonally neighboring hydrophobic residues that are not connected by a covalent bond contribute to the "energy" of a conformation. Each such interaction decrements the fitness value by one energy unit. (Remember, the lower the fitness value [energy], the better [more stable] the proton conformation.) If one residue is placed on a grid point that was already occupied by another residue each such overlap

increments the fitness value by a penalty sum of 100 energy units. Because the fitness function is to be minimized, there is a strong tendency to lose individuals with overlaps. The following source code must be saved in the file EVAL.C and compiled and linked with the remaining files of GENESIS.

```
/* FILE EVAL.C */
#include <stdio.h>
#include <stdlib.h>

#define SEQ_LENGTH  20
#define X           ((SEQ_LENGTH + 1) * 2)
#define Y           ((SEQ_LENGTH + 1) * 2)
#define SEQUENCE    {1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0,
                     0, 1, 0, 1}
```

The binary sequence in the constant SEQUENCE corresponds to the 2D primary structure B-W-B-W-W-B-B-W-B-W-W-B-W-B-B-W-W-B-W-B.

```
double eval (str, length, vect, genes)
char str[];             /* string representation          */
int length;             /* length of bit string           */
double vect[];          /* floating point representation  */
int genes;              /* number of elements in vect     */
{
  int i, j;
  int x = X / 2;
  int y = Y / 2;
  double sum = 0.0;
  static int sequence[SEQ_LENGTH] = SEQUENCE;
  static int matrix[X][Y](3);

  for (i = 0; i < X; i++)
    for (j = 0; j < Y; j++)
      matrix[i][j][0] = matrix[i][j][1] = matrix[i][j][2] = 0;
  for (i = 0; i < SEQ_LENGTH; i++)
{
  matrix[x][y][1] = i + 1;  /* holds the residue numbers */
  matrix[x][y][2] ++;       /* counts overlaps */
  if (sequence[i])
    matrix[x][y][0] = 1;    /* determines hydrophobic or ... */
  else
    matrix[x][y][0] = -1;   /* ... hydrophilic */
  if (str[2*i] = = "0")
    if (str[2*i + 1] == "0")
      x += 1;
    else
      y += 1;
    else
      if (str[2*i + 1] == "0")
```

```
            y -= 1;
      else
            x -= 1;
  }
  for (j = 1; i < X - 1; i++)
    for (j = 1; j < Y - 1; j++)
    {
      if (matrix[i][j][2] > 1)
       sum + = (matrix[i][j][2] - 1) * 200;
      if (matrix[i][j][0] = = 1)  /* look for neighboring  */
      {                           /* hydrophobic residue   */
         if   ((matrix[i-1][j][0] == 1) &&
              (abs (matrix[i][j][1] - matrix[i-1][j][1]) > 2))
              sum--;
         if   ((matrix[i][j-1][0] == 1) &&
              (abs (matrix[i][j][1] - matrix[i][j-1][1]) > 2))
              sum--;
         if   ((matrix[i + 1][j][0] == 1) &&
              (abs (matrix[i][j][1] - matrix[i + 1][j][1]) > 2))
              sum--;
         if   ((matrix[i][j + 1][0] == 1) &&
              (abs (matrix[i][j][1] - matrix[i][j + 1][1]) > 2))
              sum--;
       }

      }
    sum /= 2.0;       /* because each hydroph. interaction is counted 2x */
    if (sum == 0.0)
      sum = 1;        /* Genesis cannot cope with fitness zero */
    return sum;
  }
```

The following conformations were found by the genetic algorithm (*see* **Figs. 6** and **7**). One of them has the optimal fitness value for this application with –9 energy units.

Figure 8 shows the performance of a typical run of the genetic algorithm on the 2D protein model. As the best individual is always propagated into the next generation (elitist option "e" in GENESIS, the fitness value for the best individual of each generation decreases monotonously. The average fitness, however, fluctuates considerably because the genetic algorithm produces worse individuals all the time.

Table 3 shows some data on the performance of the GA. We can see that the number of trials becomes much smaller than the product of population size and number of generations. This means that some individuals are propagated without any changes or identical individuals are rediscovered that did not have to be evaluated again. Ten thousand evaluations divided by 50 individuals in one

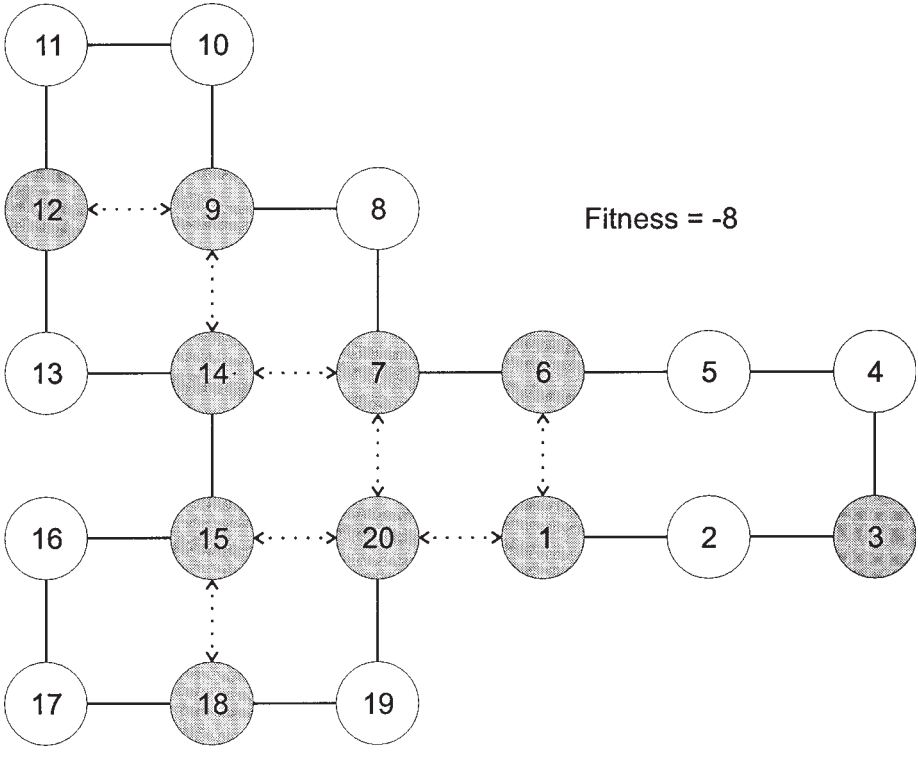


Fig. 6. Near-optimal solution with fitness -8 energy units.

generation gives 200 generations if every individual would have to be evaluated. In this run, however, about 300 generations were performed. Less than 3 (or 8) bit positions remained 100% (or 95%) constant during the run. This defies any premature convergence and shows that the genetic algorithm is still improving on its individuals. A maximum average bias of 75% for any bit position at the end of the run substantiates this finding (i.e., on the average not more than 75% of all individuals in one generation have the same value at any bit position). As expected, online (calculated over all evaluations) and offline performance (calculated only for those better than the average fitness of the last generation) both decrease steadily along with the best fitness. The average fitness of the current population tends to fluctuate considerably because there are always a few individuals created with much worse fitness values.

R. Unger and J. Moult ***(10)*** continue to show that the performance of the genetic algorithm is much more efficient than various Monte Carlo strategies. The genetic algorithm arrives faster and with less computational effort at better fitness values than Monte Carlo search. These results show that the genetic

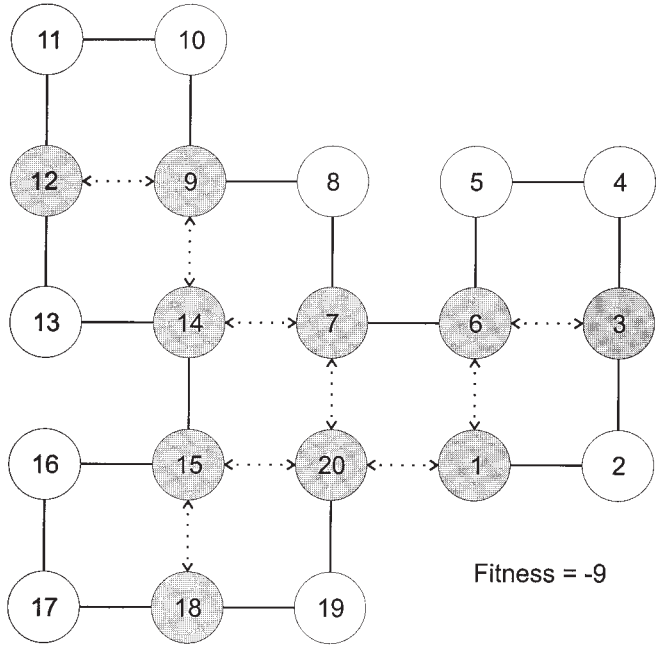


Fig. 7. Optimal solution with fitness -9 energy units.

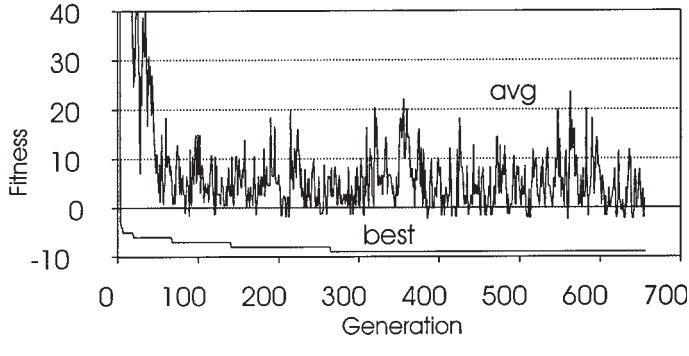


Fig. 8. Performance of genetic algorithm on the 2D protein model. The best individual was always passed on in this run, hence the monotonously decreasing fitness. The average fitness of all individuals also decreases but fluctuates considerably.

algorithm is certainly a useful search tool for the simplified protein-folding model. The next step is now to extend the protein model to three dimensions and to make the fitness function more realistic. This approach is discussed in the following sections.

Table 3
Performance Criteria for the Genetic Algorithm

Gens	Trials	Lost	Conv	Bias	Online	Offline	Best	Avg
0	50	0	0	0.55	730.24	138.00	98	730.24
1	80	0	0	0.56	683.31	123.00	98	602.04
2	111	0	0	0.57	628.90	115.89	97	513.98
3	141	0	1	0.59	585.48	111.87	97	416.00
4	171	1	1	0.60	547.30	106.92	–3	353.90
5	201	1	1	0.61	505.15	90.44	–4	298.10
6	232	1	1	0.61	474.90	77.82	–4	270.04
7	262	1	1	0.62	443.59	68.35	–5	212.02
8	292	1	1	0.62	414.65	60.81	–5	174.20
...								
17	564	0	1	0.67	263.06	29.07	–5	70.42
18	594	1	1	0.67	251.91	27.35	–5	46.66
19	624	1	2	0.69	241.33	25.80	–5	32.70
20	654	1	2	0.68	231.43	24.35	–6	24.68
21	684	1	2	0.68	223.12	23.02	–6	32.66
...								
65	2020	1	2	0.76	92.65	3.83	–6	6.70
66	2050	1	2	0.76	91.47	3.68	–6	8.70
67	2080	1	2	0.76	90.28	3.54	–6	6.66
68	2111	1	2	0.76	89.02	3.40	–7	2.74
69	2141	1	3	0.76	87.77	3.25	–7	0.84
70	2171	1	3	0.76	86.54	3.11	–7	0.94
71	2201	1	2	0.76	85.44	2.97	–7	4.82
...								
138	4243	1	6	0.77	48.17	–1.83	–7	8.34
139	4273	1	6	0.78	47.87	–1.86	–7	2.54
140	4303	1	3	0.77	47.55	–1.90	–7	4.42
141	4334	1	4	0.77	47.25	–1.94	–8	2.28
142	4365	1	4	0.77	47.04	–1.98	–8	10.44
143	4395	1	4	0.77	46.75	–2.03	–8	2.50
...								
262	8012	1	5	0.77	29.05	–4.72	–8	6.46
263	8043	2	6	0.77	28.98	–4.74	–8	6.38
264	8074	2	6	0.77	28.91	–4.75	–8	6.30
265	8105	2	6	0.77	28.81	–4.76	–9	2.34
266	8135	2	6	0.77	28.74	–4.78	–9	8.20
267	8165	2	7	0.76	28.63	–4.80	–9	2.26
268	8195	2	6	0.76	28.53	–4.81	–9	2.30
269	8225	2	6	0.76	28.46	–4.83	–9	4.28
270	8256	2	7	0.76	28.40	–4.84	–9	8.38

"Gens" denotes the number of generations calculated; "Trials" is the number of invocations of the fitness function; "Lost" refers to the number of bit positions that are 100% identical over the whole population; "Conv" refers to those that are only 95% identical; "Bias" indicates the average convergence of all positions (theoretical minimum is 50%); "Online" is the mean of all fitness evaluations so far; "Offline" is the mean of the current best evaluations, i.e., those that are improvements over the average of the previous generation; "Best" is the best fitness detected since the beginning of the run; "Avg" is the average fitness of the current population. Some of the data of this run were removed for brevity.

3.2. 3D Protein Model

Here we describe the application of a genetic algorithm to the problem of 3D protein structure prediction ***(13–14)*** with a simple force field as the fitness function. It is a continuation of work presented earlier ***(15)***. Similar research on genetic algorithms and protein folding was done independently by several groups worldwide. (For more information, or to get in touch with researchers using genetic algorithms, send an email to one of the following mailing lists: ga-molecule@interval.com, ga-list-request@aic.nrl.navy.mil or to Melanie Mitchell at mm@santafe.edu who keeps an extensive bibliography on applications of genetic algorithms in chemistry. Alternatively, try a search for "genetic algorithm" in gopher space or the Web, e.g., at gopher://veronica.sunet.se or http://altavista.digital.com.) Genetic algorithms have been used to predict optimal sequences to fit structural constraints ***(16)***, to fold Crambin in the Amber force field ***(17)*** and Mellitin in an empirical, statistical potential ***(18)***, and to predict main chain-folding patterns of small proteins based on secondary-structure predictions ***(19)***.

In this section, the individuals of the genetic algorithm are conformations of a protein and the fitness function is a simple force field. In the following, the representation formalism, the fitness function and the genetic operators are described. Then, the results of an *ab initio* prediction run and of an experiment for side-chain placement for the protein Crambin is discussed.

3.2.1. Representation Formalism

For every application of a genetic algorithm, one has to decide on a representation formalism for the "genes." In this application, the so-called hybrid approach is taken ***(8)***. This means that the genetic algorithm is configured to operate on numbers, not bit strings as in the original genetic algorithm. A hybrid representation is usually easier to implement and also facilitates the use of domain specific operators. However, three potential disadvantages are encountered:

1. Strictly speaking, the mathematical foundation of genetic algorithms holds only for binary representations, although some of the mathematical properties are also valid for a floating-point representation.
2. Binary representations run faster in many applications.
3. An additional encoding/decoding process may be required to map numbers onto bit strings.

It is not the principal goal of this application to find the single optimal conformation of a protein based on a force field, but to generate a small set of nativelike conformations. For this task, the genetic algorithm is an appropriate tool. For a hybrid representation of proteins, one can use Cartesian coordinates, torsion angles, rotamers, or an otherwise simplified model of residues.

For a representation in Cartesian coordinates the 3D coordinates of all atoms in a protein are recorded. This representation has the advantage of being easily converted to and from the 3D conformation of a protein. However, it has the disadvantage that a mutation operator would, in most instances, create invalid protein conformations where some atoms lie too far apart or collide; therefore, a filter is needed that eliminates invalid individuals. Because such a filter would consume a disproportionate large amount of CPU time a Cartesian coordinate representation considerably slows down the search process of a genetic algorithm.

Another representation model is by torsion angles. Here, a protein is described by a set of torsion angles under the assumption of constant standard binding geometries. Bond lengths and bond angles are taken to be constant and cannot be changed by the genetic algorithm. This assumption is certainly a simplification of the real situation where bond length and bond angle to some extent depend on the environment of an atom. However, torsion angles provide enough degrees of freedom to represent any native conformation with only small root-mean-squares (RMS) deviations. (RMS = root-mean-square deviation; two conformations are superimposed and the square root is calculated from the sum of the squares of the distances between corresponding atoms.)

Special to the torsion angle representation is the fact that even small changes in the φ (phi)/ψ (psi) angles can induce large changes in the overall conformation. This is useful when creating variability within a population at the beginning of a run. **Figure 9** explains the definition of the torsion angles φ, ψ, ω (omega), χ_1 (chi1), and χ_2 (chi2). A small fragment taken from a hypothetical protein is shown. Two basic building blocks — the amino acids phenylalanine (Phe) and glycine (Gly) — are drawn as wire frame models. Atoms are labeled with their chemical symbols. Bonds in bold print indicate the backbone. The labels of torsion angles are placed next to their rotatable bonds.

In this report, the torsion angle representation is used. Torsion angles of 129 proteins from the Brookhaven database ***(20)*** (PDB) were statistically analyzed for the definition of the MUTATE operator. The frequency of each torsion angle in intervals of 10° was determined and the 10 most frequently occurring intervals are made available for substitution of individual torsion angles by the MUTATE operator. At the beginning of the run, individuals were initialized with either a completely extended conformation where all torsion angles are 180°, or by a random selection from the ten most frequently occurring intervals of each torsion angle. For the ω torsion angle the constant value of 180° was used because of the rigidity of the peptide bond between the atoms C_i and N_{i+1}. A statistical analysis of ω angles shows that, with the exception of proline average deviations from the mean of 180° occur rather frequently up to 5°, and only in rare cases up to 15°.

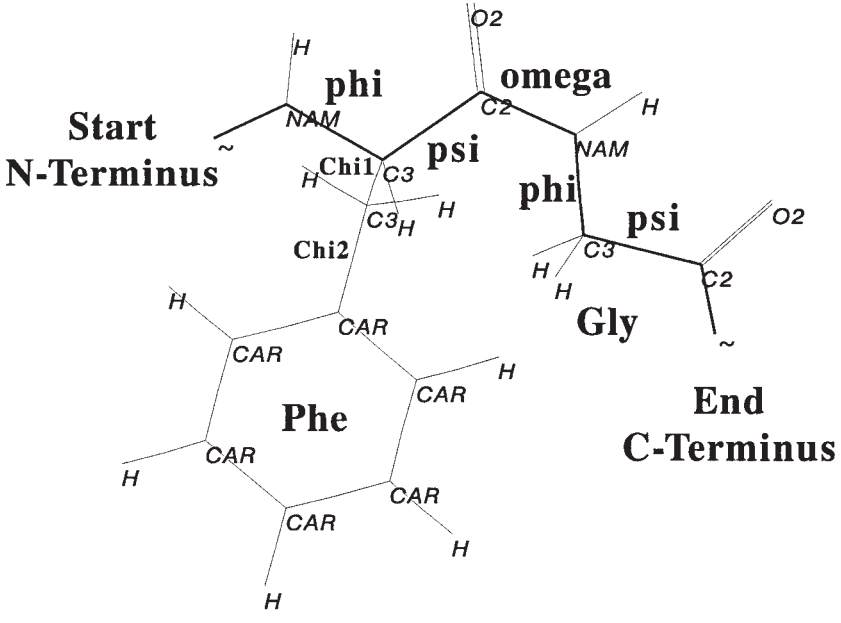


Fig. 9. Torsion angles φ, ψ, ω, χ_1, and χ_2.

The genetic operators in this application operate on the torsion angle representation, but the fitness function requires a protein conformation to be expressed in Cartesian coordinates. For the implementation of a conversion program, bond angles were taken from the molecular modeling software Alchemy ***(21)*** and bond lengths from the program CHARMM ***(22)***. Either a complete form with explicit hydrogen atoms, or the so-called extended atom representation with small groups of atoms represented as "superatoms," can be calculated. One conformation of a protein is encoded as an array of structures of the C programming language. The number of structures equals the number of residues in the protein. Each structure includes a three-letter identifier of the residue type and 10 floating-point numbers for the torsion angles φ, ψ, ω, χ_1, χ_2, χ_3, χ_4, χ_5, χ_6, and χ_7. For residues with less than seven side-chain torsion angles, the extra fields are filled with a default value. The main chain torsion angle ω was kept constant at 180°.

3.2.2. Fitness Function

In this application, a simple steric potential energy function was chosen as the fitness function (i.e., the objective function to be minimized). It is very difficult to find the global optimum of a potential energy function because of the large number of degrees of freedom even for a protein of average size. In

general, molecules with n atoms have $3n - 6$ degrees of freedom. For the case of a medium-sized protein of 100 residues this amounts to:

$$[(100 \text{ residues} \cdot \text{approximately } 20 \text{ atoms per residue}) \cdot 3] - 6 = 5994$$

degrees of freedom. Systems of equations with this number of variables are analytically intractable today. Empirical efforts to heuristically find the optimum are almost as difficult ***(23)***. If there are no constraints for the conformation of a protein, and only its primary structure is given, the number of conformations for a protein of medium size (100 residues) can be approximated to:

$$(5 \text{ torsion angles per residue} \cdot 5 \text{ likely values per torsion angle})^{100} = 25^{100}$$

This means that in the worst case 25^{100} conformations would have to be evaluated to find the global optimum. This is clearly beyond the capacity of today's and tomorrow's supercomputers. As can be seen from a number of previous applications, genetic algorithms were able to find suboptimal solutions to problems with an equally large search space ***(24–26)***. Suboptimal in this context means that it cannot be proven that the solutions generated by the genetic algorithm do, in fact, include an optimal solution, but that some of the results generated by the genetic algorithm practically surpassed any previously known solution. This can be of much help in nonpolynomial complete problems where no analytical solution of the problem is available.

3.2.3. Conformational Energy

The steric potential energy function was adapted from the program CHARMM. The total energy of a protein in solution is the sum of the expressions for E_{bond} (bond length potential), E_{phi} (bond angle potential), E_{tor} (torsion angle potential), E_{impr} (improper torsion angle potential), E_{vdW} (van der Waals pair interactions), E_{el} (electrostatic potential), E_{H} (hydrogen bonds), and of two expressions for interaction with the solvent, E_{cr} and E_{cphi}:

$$E = E_{\text{bond}} + E_{\text{phi}} + E_{\text{tor}} + E_{\text{impr}} + E_{\text{vdW}} + E_{\text{el}} + E_{\text{cr}} + E_{\text{cphi}}$$

Here we assume constant bond lengths and bond angles. The expressions for E_{bond}, E_{phi}, and E_{impr} are therefore constant for different conformations of the same protein. The expression E_{H} was omitted because it would have required the exclusion of the effect of hydrogen bonds from the expressions for E_{vdW} and E_{el}. This, however, was not done by the authors of CHARMM in their version v.21 of the program. In all runs, folding was simulated in vacuum with no ligands or solvent, i.e., E_{cr} and E_{cphi} are constant. This is certainly a crude simplification of the real situation, but is, nevertheless, more detailed

than the 2D protein model in **Subheading 3.1.** Thus, the potential energy function simplifies to:

$$E = E_{tor} + E_{impr} + E_{vdW} + E_{el}$$

Test runs showed that if only the three expressions E_{tor}, E_{vdW}, and E_{el} are used, there would not be enough force to drive the protein to a compact folded state. An exact solution to this problem requires the consideration of entropy. The calculation of the entropy difference between a folded and unfolded state is based on the interactions between protein and solvent. Unfortunately, it is not yet possible to routinely calculate an accurate model of those interactions. It was therefore decided to introduce an *ad hoc* pseudoentropic term E_{pe} that drives the protein to a globular state. The analysis of a number of globular proteins reveals the following empirical relation between the number of residues (length) and the diameter:

$$\text{expected diameter} = 8 \cdot \sqrt[3]{\text{length}}$$

The pseudoentropic term E_{pe} for a conformation is a function of its actual diameter. The diameter is defined to be the largest distance between any C_α atoms in one conformation. An exponential of the difference between actual and expected diameter is added to the potential energy if that difference is less than 15 Å. If the difference is greater than 15 Å, a fixed amount of energy is added (10^{10} kcal/mol) to avoid exponential overflow. If the actual diameter of an individual is smaller than the expected diameter, E_{pe} is set to zero. The net result is that extended conformations have larger energy values and are therefore less fit for reproduction than globular conformations.

$$E_{pe} = 4^{(\text{actual diameter} - \text{expected diameter})} \text{ [kcal/mol]}$$

Occasionally, if two atoms are very close, the E_{vdW} term can become very large. The maximum value for E_{vdW} in this case is 10^{10} kcal/mol and the expressions for E_{el} and E_{tor} are not calculated. Runs were performed with the potential energy function E as described earlier where lower fitness values mean fitter individuals and with a variant, where the four expressions E_{tor}, E_{vdW}, E_{el}, and E_{pe} were given individual weights. The results were similar in all cases. Especially, scaling down the dominant effect of electrostatic interactions did not change the results.

3.2.4. Genetic Operators

In order to combine individuals of one generation to produce new offspring, nature as well as genetic algorithms apply several genetic operators. In this volume, individuals are protein conformations represented by a set of torsion angles under the assumption of constant standard binding geometries. Three

operators are invented to modify these individuals: MUTATE, VARIATE, and CROSSOVER. The decision about the application of an operator is made during run time and can be controlled by various parameters.

3.2.4.1. MUTATE

The first operator is the MUTATE operator. If MUTATE gets activated for a particular torsion angle, this angle will be replaced by a random choice of one of the 10 most frequently occurring values for that type of residue. The decision whether a torsion angle will be modified by MUTATE is made independently for each torsion angle in a protein. A random number between 0 and 1 is generated, and if this number is greater than the MUTATE parameter at that time, MUTATE is applied. The MUTATE parameter can change dynamically during a run. The values that MUTATE can choose from come from a statistical analysis of 129 proteins from PDB. The number of instances in each of the 36 10° intervals was counted for each torsion angle. The 10 most frequent intervals, each represented by its left boundary, are available for substitution.

3.2.4.2. VARIATE

The VARIATE operator consists of three components: the 1°, 5°, and 10° operator. Independently and after application of the MUTATE operator for each torsion angle in a protein two decisions are made: first, whether the VARIATE operator will be applied and, second, if so which of the three components shall be selected. The VARIATE operator increments or decrements (always an independent random chance of 1:2) the torsion angle by 1°, 5°, or 10°. Care is taken that the range of torsion angles does not exceed the [–180°, 180°] interval. The probability of applying this operator is controlled by the VARIATE parameter, which can change dynamically during run time. Similarly, three additional parameters control the probability for choosing among the three components. Alternatively, instead of three discrete increments, a Gaussian uniformly distributed increment between –10° and +10° can be used.

3.2.4.3. CROSSOVER

The CROSSOVER operator has two components: the two-point crossover and the uniform crossover. CROSSOVER is applied to two individuals independently of the MUTATE and VARIATE operators. First, individuals of the parent generation, possibly modified by MUTATE and VARIATE, are randomly grouped pairwise. For each pair, an independent decision is made whether or not to apply the CROSSOVER operator. The probability of this is controlled by a CROSSOVER parameter, which can change dynamically during run time. If the decision is "no," the two individuals are not further modified and added to the list of offspring. If the decision is "yes," a choice between the two-point crossover and

Table 4
Run-Time Parameters

Parameter	Value
ω Angle constant 180°	on
Initialize start generation	random
Number of individuals	10
Number of generations	1000
MUTATE (start)	80%
MUTATE (end)	20%
MUTATE (start)	20%
VARIATE (end)	70%
VARIATE (start 10°)	60%
VARIATE (end 10°)	0%
VARIATE (start 5°)	30%
VARIATE (end 5°)	20%
VARIATE (start 1°)	10%
VARIATE (end 1°)	80%
CROSSOVER (start)	70%
CROSSOVER (end)	10%
CROSSOVER (start uniform)	90%
CROSSOVER (end uniform)	10%
CROSSOVER (start two point)	10%
CROSSOVER (end two point)	90%

the uniform crossover must be made. This decision is controlled by two other parameters that can also be changed during run time. The two-point crossover randomly selects two residues on one of the individuals. Then the fragment between the two residues is exchanged with the corresponding fragment of the second individual. Alternatively, uniform crossover decides independently for each residue whether or not to exchange the torsion angles of that residue. The probability for an exchange is then always 50%.

3.2.4.4. Parameterization

As mentioned in the previous paragraphs, there are a number of parameters that control the run time behavior of a genetic algorithm. The parameter values used for the experiments that will be presented in the **Subheading 3.2.5.** are summarized in **Table 4**. The main chain torsion angle ω was kept constant at 180°. The initial generation was created by a random selection of torsion angles from a list of the 10 most frequently occurring values for each angle. Ten individuals are in one generation. The genetic algorithm was halted after 1000 generations. At the start of the run, the probability for a torsion angle to be

modified by the MUTATE operator is 80%; at the end of the run it becomes 20%. In between, the probability decreases linearly with the number of generations. In contrast, the probability of applying the VARIATE operator increases from 20% at the beginning to 70% at the end of the run. The 10° component of the VARIATE operator is dominant at the start of the run (60%), whereas it is the 1° component at the end (80%). Similarly, the chance of performing a CROSSOVER rises from 10% to 70%. At the beginning of the run mainly uniform CROSSOVER is applied (90%), at the end it is mainly two-point CROSSOVER (90%). This parameter setting uses a small number of individuals but runs over a large number of generations. This keeps computation time low while allowing a maximum number of crossover events. At the beginning of the run, MUTATE and uniform CROSSOVER are applied most of the time to create some variety in the population so that many different regions of the search space are covered. At the end of the run, the 1° component of the VARIATE operator dominates the scene. This is intended for fine tuning those conformations that have survived the selection pressure of evolution so far.

3.2.4.5. Generation Replacement

There are different ways of selecting the individuals for the next generation. Given the constraint that the number of individuals should remain constant, some individuals have to be discarded. Transition between generations can be done by total replacement, elitist replacement, or steady-state replacement. For total replacement, only the newly created offspring enter the next generation and the parents of the previous generation are completely discarded. This has the disadvantage that a fit parent can be lost even if it only produces bad offspring once. With elitist replacement, all parents and offspring of one generation are sorted according to their fitness. If the size of the population is n, then the n fittest individuals are selected as parents for the following generation. This mode has been used here. Another variant is steady-state replacement, where two individuals are selected from the population based on their fitness and then modified by mutation and crossover. They are then used to replace their parents.

3.2.5. Ab initio Prediction

A prototype of a genetic algorithm with the representation, fitness function, and operators as described earlier has been implemented. To evaluate the *ab initio* prediction performance of the genetic algorithm the sequence of Crambin was given to the program. Crambin is a plant seed protein from the cabbage *Crambe abyssinica*. Its structure was determined by W. A. Hendrickson and M. M. Teeter ***(27*** to a resolution of 1.5 Å (*see* **Figs. 10** and **11**). Crambin has a strong amphiphilic character that makes its conformation especially difficult to

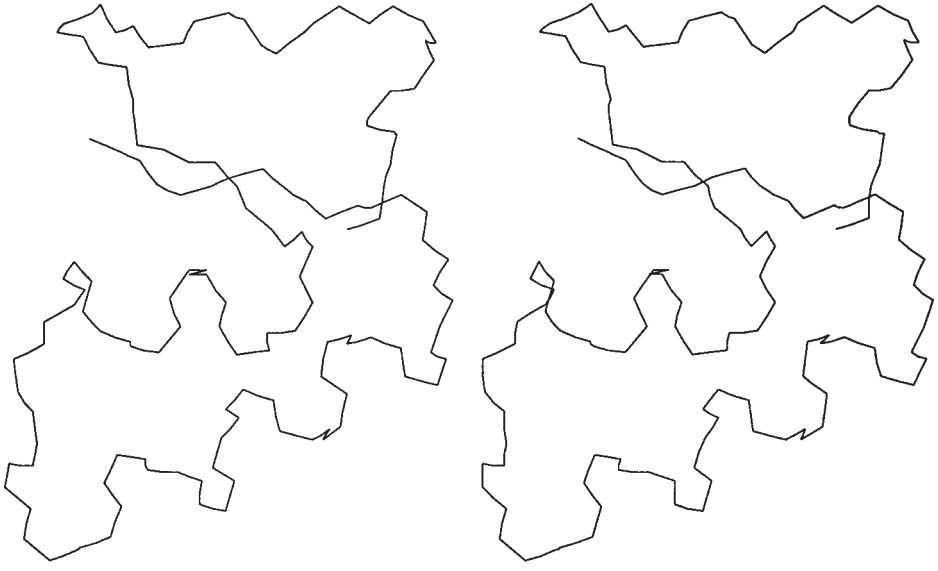


Fig. 10. Stereoprojection of Crambin without side chains.

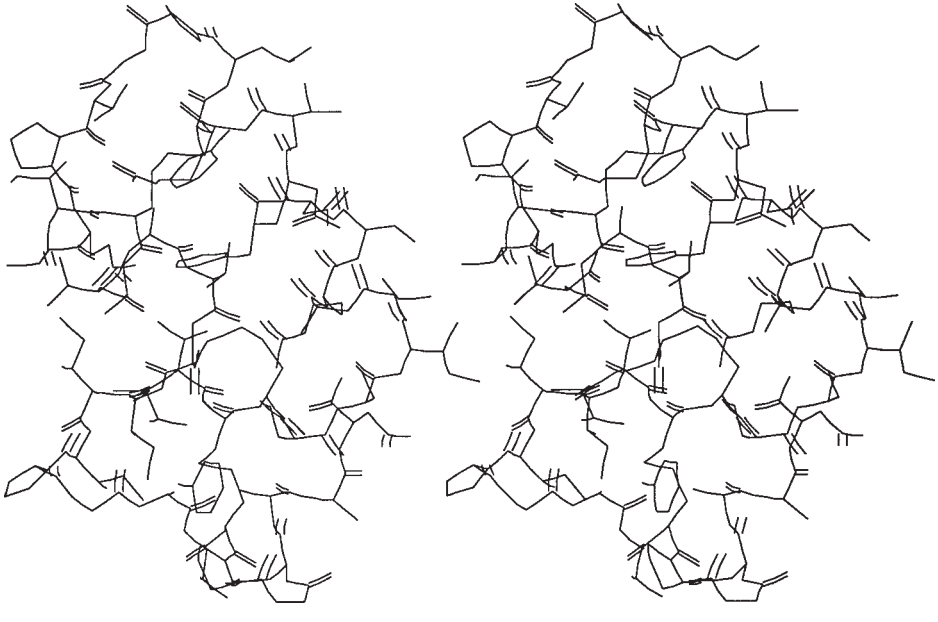


Fig. 11. Stereoprojection of Crambin with side chains.

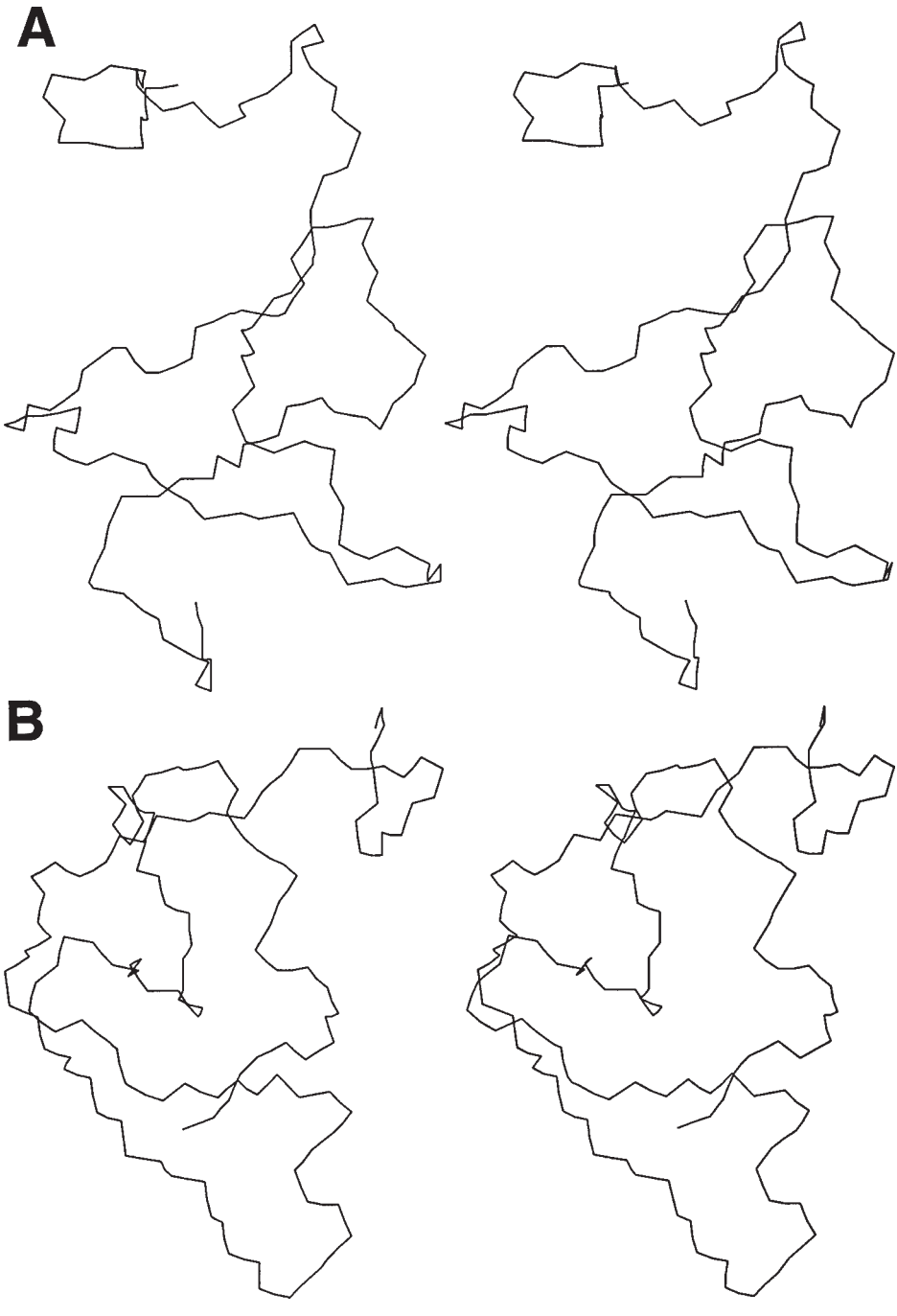


Fig. 12. Two conformations generated by the genetic algorithm.

Table 5
RMS Deviations to Native Crambin

Individual	RMS	Individual	RMS
P1	10.07 Å	P6	10.31 Å
P2	9.74 Å	P7	9.45 Å
P3	9.15 Å	P8	10.18 Å
P4	10.14 Å	P9	9.37 Å
P5	9.95 Å	P10	8.84 Å

The 10 individuals of the last generation were measured against the native conformation of Crambin. RMS values of around 9 Å for a small protein as Crambin exclude any significant structural similarity.

predict. However, because of its good resolution and small size of 46 residues, it was decided to use Crambin as a first candidate for prediction. The following structures are again displayed in stereo projection. If the observer manages to look cross eyed at the diagram in a way that superimposes both halves a 3D image can be perceived.

Figure 12 shows two of the 10 individuals in the last generation of the genetic algorithm. None of the 10 individuals shows significant structural similarity to the native Crambin conformation. This can be confirmed by superimposing the generated structures with the native conformation. **Table 5** shows the RMS differences between all ten individuals and the native conformation. All values are in the range of 9 Å, which rejects any significant structural similarity.

Although the genetic algorithm did not produce nativelike conformations of Crambin, the generated backbone conformations could be those of a protein, i.e., they have no knots or unreasonably protruding extensions. The conformational results alone would indicate a complete failure of the genetic algorithm approach to conformational search, but let us have a look at the energies in the final generation (*see* **Table 6**). All individuals have a much lower energy than native Crambin in the same force field. That means that the genetic algorithm actually achieved a substantial optimization but that the current fitness function was not a good indicator of "nativeness" of a conformation.

It is obvious that all individuals generated by the genetic algorithm have a much higher electrostatic potential than native Crambin. There are three reasons for this.

1. Electrostatic interactions are able to contribute larger amounts of stabilizing energy than any of the other fitness components.
2. Crambin has six partially charged residues that were not neutralized in this experiment.

Table 6
Steric Energies in the Last Generation

Individual	E_{vdw}	E_{el}	E_{tor}	E_{pe}	E_{total}
P1	–14.9	–2434.5	74.1	75.2	–2336.5
P2	–2.9	–2431.6	76.3	77.4	–2320.8
P3	78.5	–2447.4	79.6	80.7	–2316.1
P4	–11.1	–2409.7	81.8	82.9	–2313.7
P5	83.0	–2440.6	84.1	85.2	–2308.5
P6	–12.3	–2403.8	86.1	87.2	–2303.7
P7	88.3	–2470.8	89.4	90.5	–2297.6
P8	–12.2	–2401.0	91.6	92.7	–2293.7
P9	93.7	–2404.5	94.8	95.9	–2289.1
P10	96.0	–2462.8	97.1	98.2	–2287.5
Crambin	–12.8	11.4	60.9	1.7	61.2

For each individual, the van der Waals energy (E_{vdW}), electrostatic energy (E_{el}), torsion energy (E_{tor}), psuedoentropic energy (E_{pe}), and the sum of all terms (E_{total}) is shown. For comparison, the values for native Crambin in the same force field are listed.

3. The genetic algorithm favored individuals with the lowest total energy, which in this case was most easily achieved by optimizing electrostatic contributions.

The final generation of only 10 individuals contained two fundamentally different families of structures (class 1: P1, P2, P4, P5, P6, P8, P9) and (class 2: P3, P7, P10). Members of one class have a RMS deviation of about 2 Å among themselves but differ from members of the other class by about 9 Å.

Taking into account the small population size, the significant increase in total energy of the individuals generated by the GA, and the fact that the final generation contained two substantially different classes of conformations with very similar energies, one is led to the conclusion that the search performance of the genetic algorithm was not that bad at all. What remains a problem is to find a better fitness function that actually guides the genetic algorithm to nativelike conformations. Because the only criterion currently known to determine native conformation is the free energy, the difficulty of this approach becomes obvious. One possible way to cope with the problem of inadequate fitness functions is to combine other heuristic criteria together with force field components in a multivalue vector fitness function. Before we turn to that approach, let us first examine the performance of the current version for side-chain placement.

3.2.6. Side-Chain Placement

Crystallographers often face the problem of positioning the side chains of a protein when the primary structure and the conformation of the backbone is

known. At present, there is no method that automatically does side-chain placement with sufficiently high accuracy for routine practical use. Although the side-chain placement problem is conceptually easier than *ab initio* tertiary structure prediction, it is still too complex for analytical treatment.

The genetic algorithm approach as described can be used for side-chain placement. The torsion angles φ, ψ, and ω simply have to be kept constant for a given backbone. Side-chain placement by the genetic algorithm was done for Crambin. For each five residues, a superposition of the native and predicted conformation is shown in stereo projection graphs in **Fig. 13**. As we can see, the predictions agree quite well with the native conformation in most cases. The overall RMS difference in this example is 1.86 Å. This is not as good, as but is comparable to, the results from a simulated annealing approach ***(28)*** (1.65 Å) and a heuristic approach ***(29)*** (1.48 Å).

It must be emphasized that these runs were done without optimizing either the force field parameters of the fitness function or the run-time parameters of the genetic algorithm. From a more elaborate and fine-tuned experiment, even better results should be expected.

3.3. Multiple-Criteria Optimization of Protein Conformations

In this section we introduce additional fitness criteria for the protein-folding application with genetic algorithms. The rationale is that more information about genuine protein conformations should improve the fitness function to guide the genetic algorithm toward nativelike conformations. Some properties of protein conformations can be used as additional fitness components, whereas others can be incorporated into genetic operators (e.g., constraints from the Ramachandran plot). For such an extended fitness function, several incommensurable quantities will have to be combined: energy, preferred torsion angles, secondary-structure propensities, or distributions of polar and hydrophobic residues. This creates the problem of how to combine the different fitness contributions to arrive at the total fitness of a single individual. Simple summation of different components has the disadvantage that components with larger numbers would dominate the fitness function whether or not they are important or of any significance at all for a particular conformation. To cope with this difficulty, individual weights for each of the components could be introduced. But this creates another problem. How should one determine useful values for these weights? Because there is no general theory known for the proper weighting of each fitness component, the only way is to try different combinations of values and evaluate them by their performance of a genetic algorithm on test proteins with known conformations. However, even for a small number of fitness components, a large number of combinations of weights arises that requires as many test runs for evaluation. Also, "expensive" fitness components as the van der

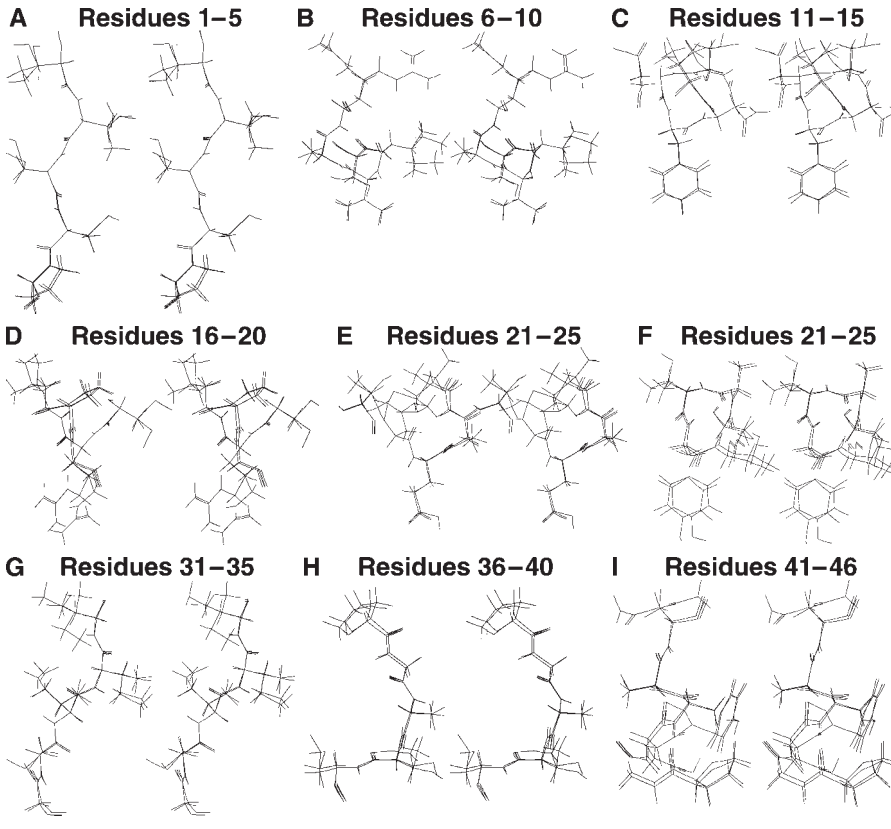


Fig. 13. Side-chain placement results. A spatial superimposition in stereoscopic wire frame diagrams is shown for every five residues of Crambin and the corresponding fragment generated by a genetic algorithm. The amino acid sequence of Crambin in one letter code is `TTCCP SIVAR SNFNV CRLPG TPEAI CATYT GCIII PGATC PGDYA N`.

Waals energy need considerable computation time. Here, two measures were taken to deal with this situation:

1. Different fitness components are not arithmetically added to produce a single numerical fitness value, but they are combined in a vector. This means that each fitness component is individually carried along the whole evaluation process and is always available explicitly.
2. Parallel processing is employed to evaluate all individuals of one generation in parallel. For populations of 20–60 individuals, this gave a speedup of about 20-fold compared to small single-processor workstations.

3.3.1. Vector Fitness Function

In this application two versions of a fitness function are used. One version is a scalar fitness function that calculates the RMS deviation of a newly generated

individual from the known conformation of the test protein. This geometric measure should guide the genetic algorithm directly to the desired solution, but it is only available for proteins with a known conformation. RMS deviation is calculated as follows:

$$\mathrm{RMS} = \sqrt{N}\sum_i (|\boldsymbol{u}_i - \boldsymbol{v}_i|)^2/N$$

Here, i is the index over all corresponding N atoms in the two structures to be compared, in this case the conformation of an individual ($\boldsymbol{u}_i$) in the current population and the known, actual structure ($\boldsymbol{v}_i$) of the test protein. The squares of the distances between the vectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ of corresponding atoms are summed and the square root is taken. The result is a measure of how much each atom in the individual deviates on average from its true position. RMS values of 0–3 Å signify strong structural similarity; values of 4–6 Å denote weak structural similarity, whereas for small proteins, RMS-values over 6 Å mean that probably not even the backbone folding pattern is similar in both conformations.

The other version of the fitness function is a vector of several fitness components, which is explained in the following paragraphs. This multivalue vector fitness function includes the following components:

$$\text{fitness} = \begin{pmatrix} \mathrm{RMS} \\ E_{\mathrm{tor}} \\ E_{\mathrm{vdW}} \\ E_{\mathrm{el}} \\ E_{\mathrm{pe}} \\ \mathit{polar} \\ \mathit{hydro} \\ \mathit{scatter} \\ \mathit{solvent} \\ \mathit{Crippen} \\ \mathit{clash} \end{pmatrix}$$

RMS is the RMS deviation as described earlier. It can only be calculated in test runs with the protein conformation known beforehand. For the multivalue vector fitness function, this measure was calculated for each individual to see how close the genetic algorithm came to the known structure. In these runs, however, the RMS measure was not used in the offspring selection process. Selection was done only based on the remaining ten fitness components and a Pareto selection algorithm, which is explained shortly.

E_{tor} is the torsion energy of a conformation based on the force field data of the CHARMM force field v.21 with k and n as force field constants depending on the type of atom and ϕ as the torsion angle:

$$E_{\mathrm{tor}} = |k_\phi| - k_\phi \cos(n\phi)$$

E_{vdW} is the van der Waals energy (also called the Lennard-Jones potential) with A and B as force field constants depending on the type of atom and r as the distance between two atoms in one molecule. The indices i and j for the two atoms may not have identical values and each pair is counted only once:

$$E_{vdW} = \sum_{excl(i<j)} (A_{ij} / r_{ij}^{12} - B_{ij} / r_{ij}^{6})$$

E_{el} is the electrostatic energy between two atoms with $q_{i,j}$ as the partial charges of the two atoms i and j and r as the distance between them:

$$\text{Eel} = \sum_{excl(i<j)} q_i q_j / 4\pi\varepsilon_0 r_{ij}$$

E_{pe} is a measure to promote compact folding patterns. The expected diameter of a protein can be estimated by a number of techniques. A penalty energy term is then calculated as follows:

$$E_{pe} = 4^{(\text{actual diameter} - \text{expected diameter})}$$

Polar is a measure that favors polar residues on the protein surface but not in the core. Because all fitness contributions should be minimized, a factor of minus one is required before the sum. The larger the distances of polar residues to the center of the protein, the better a conformation and the more negative the value of "polar." If residue i is one of k polar residues (any of Arg, Lys, Asn, Asp, Glu, or Gln) in a protein of length N residues and with $\boldsymbol{s}$ as the center of gravity, then the polar fitness contribution is calculated as follows:

$$\text{polar} = -\sum_{i}^{N} |\boldsymbol{u}_i - \boldsymbol{s}| / k$$

Hydro is a similar measure that favors hydrophobic residues (Ala, Val, Ile, Leu, Phe, Pro, Trp) in the core of a protein, whereas *scatter* promotes compact folds as it adds up the distances over all C_α atoms irrespective of amino acid type:

$$\text{hydro} = \sum_{i}^{N} |\boldsymbol{v}_i - \boldsymbol{s}| / k \quad \text{scatter} = \sum_{i}^{N} |\boldsymbol{v}_i - \boldsymbol{s}| / N$$

Solvent is the solvent accessible surface of a conformation in Å^2. It is calculated by a surface triangulation method.

Crippen is an empirical, statistical potential developed by G. Crippen ***(31)***. It is summed over all pairs of atoms that interact within a certain distance.

Clash is a term that counts the number of atomic collisions where any two atoms come closer than 3.8 Å to each other. This fitness term can be used to approximate the effect of the van der Waals energy at small distances but at only a fraction of the computational cost:

$$\text{clash} = \sum_{j=1}^{N} \sum_{j=i+1}^{N} \text{overlap}(i,j) \text{ with overlap}(i,j) = \begin{cases} 0 & \text{if dist(i,j)} \geq 3.8 \text{ Å} \\ 1 & \text{if dist(i,j)} < 3.8 \text{ Å} \end{cases}$$

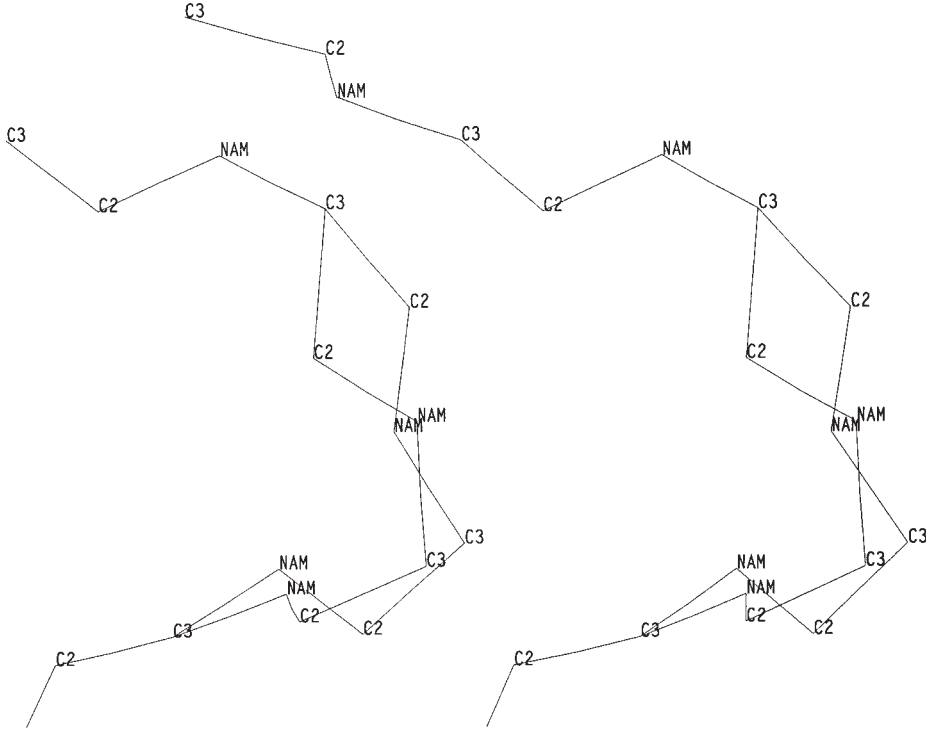


Fig. 14. Backbone conformation changed by LOCAL TWIST operator. Stereo-projection of a portion of three residues and an alternative fold found by the LOCAL TWIST operator.

3.3.2. Specialized Genetic Operators

3.3.2.1. Local Twist Operator

The LOCAL TWIST operator introduces local conformation changes by performing the ring closure algorithm for polymers of N. Go and H. A. Scheraga ***(32)*** for three consecutive amino acid residues (*see* **Fig. 14**). This algorithm was originally implemented in the `RING.FOR` program (Quantum Chemical Exchange Program [QCEP], program no. QCMP 046) in a general way that operated on six adjacent dihedral angles to bridge a gap with bonds of defined length and bond angle. The application of this algorithm for a polypeptide required translation of the program into the C programming language and some alterations to the program to account for the intermitting rigid ω torsion angle.

The basic concept of the ring-closure algorithm is to find suitable values for ϕ_1 that satisfy the following equation:

$$g(\phi_1) = \mathbf{u}^+ \mathbf{T}_\alpha \mathbf{R}_{\phi^1} \mathbf{T}_\beta \mathbf{R}_{\psi^1+\pi} \mathbf{T}_\alpha \mathbf{R}_{\phi^2} \mathbf{T}_\beta \mathbf{R}_{\psi^2+\pi} \mathbf{T}_\alpha \mathbf{e}_1 - \cos(\beta) = 0$$

Here, $\mathbf{u}^+$ (transposed) and $\mathbf{e}_1$ are vectors, and **T** and **R** are several translation and rotation matrices that define the constraints of a local conformation change, respectively. Angle β describes the rigid geometry of a peptide bond, and ϕ_1 is the first backbone torsion angle in sequence to be modified. The search for suitable values of ϕ_1 involves repeated numerical approximations and is therefore rather time consuming. Hence, it was decided to distribute the LOCAL TWIST operator over several processors on a parallel computer (Intel Paragon with 98 × i860 processors owned by Parallab, University of Bergen, Norway) so that the calculations can be carried out in parallel for all individuals. In test runs with RMS deviation to the native conformation as the fitness function, the LOCAL TWIST operator led to significant improvements in prediction accuracy and also to a substantial decrease in overall computation time.

3.3.2.2. Preferred Backbone Conformations

The MUTATE operator of **Subheading 3.2.4.** is rather crude because it always uses the left boundary of one of the 10 most frequently occurring 10° intervals for a torsion angle. To improve the chance of selecting favorable values for the backbone torsion angles ϕ and ψ, a cluster analysis with a modified nearest-neighbor algorithm ***(33)*** was performed for the main chain torsion angles of 66 proteins:

1. Cluster all ϕ/ψ pairs for each amino acid until 21 clusters are formed.
2. Collect all clusters with less than 10 pairs and add the center of each cluster to the set of ϕ/ψ pairs to be used by the MUTATE operator.
3. Repeat the clustering procedure with only the ϕ/ψ pairs from the clusters with at least 10 pairs in **step 2** and let the clustering program run again until 21 clusters are formed. The centers of all new clusters complete the list of ϕ/ψ pairs that MUTATE uses when substituting individual torsion angles.

This algorithm first identifies small clusters with only few examples in detail and then clusters more densely populated areas with a finer resolution than a single clustering would do in one pass. **Figure 15** shows the centers of 34 clusters for arginine.

3.3.2.3. Secondary Structure

In addition to a more accurate selection of preferable main chain torsion angles, predictions of secondary structure were used to reduce the search space. Two issues arise that must be considered:

1. Which secondary structure prediction algorithm should one rely on?
2. Which torsion angles should be used for the predicted secondary structures?

The first question was addressed by assembling a consensus prediction from two different methods: the PHD artificial neural network ***(34)*** and a statistical

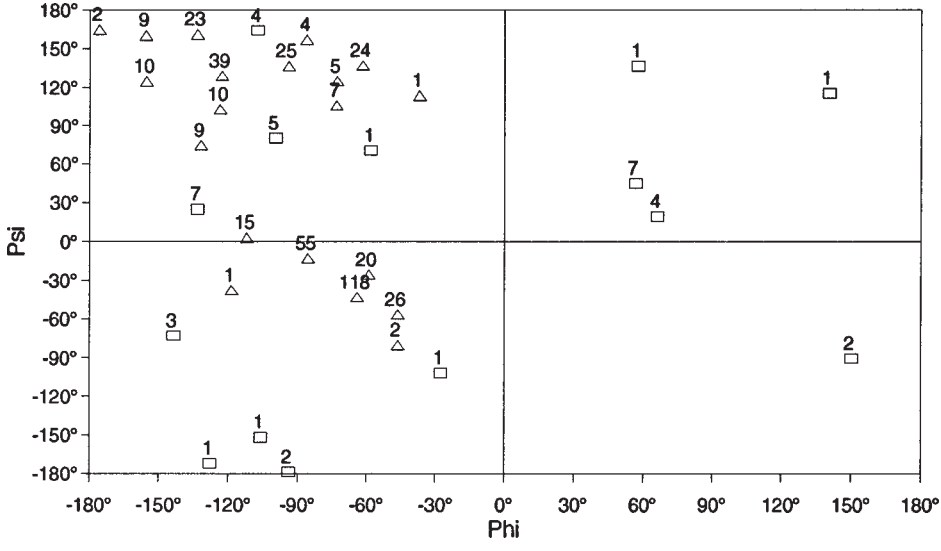


Fig. 15. Thirty-four ϕ/ψ clusters for arginine. There are 14 small clusters of the first pass with less than 10 pairs (shown as boxes) and 20 large clusters of the remaining pairs in the second pass (triangles).

analysis that uses information theory ***(35,36)***. For the second question, there are two alternate solutions. One alternative is to use torsion angles of idealized α-helices and β-strands, another is to constrain torsion angles of the predicted secondary-structures to an interval that includes the conformation with idealized geometry. The corresponding torsion angles are shown in **Table 7**.

3.3.3. Genetic Algorithm Performance

Using the genetic algorithm as described earlier produced the following results. **Figure 16** shows the best individual of the final generation of a run with a population of 30 individuals, the LOCAL TWIST operator in effect, and RMS deviation as the only fitness component ***(37)***. For Crambin, the final RMS deviation of the conformation generated by the genetic algorithm is 1.08 Å, which is well within the range of the best resolution from X-ray or NMR structure elucidation experiments. Another run with the same parameters produced an individual with an RMS deviation of 0.89 Å. This demonstrates the suitability of the genetic algorithm approach to protein folding. Given a reliable fitness function, the genetic algorithm is able to successfully traverse the torsion-angle search space.

Other proteins that were used for test purposes of the genetic algorithm with an RMS-fitness function are the trypsin inhibitor protein (Brookhaven data-

Table 7
Boundaries for Main Chain Torsion Angles in Secondary Structures

Secondary structure	ϕ_l	ϕ_u	ψ_l	ψ_u	ϕ_{exact}	ψ_{exact}
α-Helix (narrow interval)	–57°	–62°	–41°	–47°	–57°	–47°
α-Helix (broad interval)	–30°	–120°	10°	–90°	—	—
β-Strand (narrow interval)	–119°	–139°	135°	113°	–130°	125°
β-Strand (broad interval)	–50°	–180°	180°	80°	—	—

ϕ_l, ψ_l and ϕ_u, ψ_u are lower and upper values of the main chain torsion angles in the respective secondary structure. ψ extract, and φ extract are values for an idealized standard geometry. For β-strands the values are an average of parallel and antiparallel strands.

base code 5pti; final RMS deviation 1.48 Å; **Fig. 17**) and RNAse T1 (Brookhaven database code 2rnt, final RMS deviation 2.32 Å; **Fig. 18**).

That none of the structures produced in the runs with an RMS-fitness function were completely identical to the native conformations is explained by the following three observations:

1. The use of standard binding geometries for reconstructing 3D coordinates from a set of torsion angles could cause structural alterations where the native conformation does not adhere closely to the theoretically derived ideal bond lengths and bond angles. In this case the best match will always have an RMS deviation of greater than zero.
2. The operators MUTATE, VARIATE, and CROSSOVER in theory cannot produce an exact match even if the target structure is known in detail. This is a result of the representation formalism that these operators work on. If the current individual is already structurally similar to the desired protein, a single application of MUTATE or VARIATE is most likely to introduce mismatches of previously well-fitting fragments, and thus deteriorates the conformation. This happens because even if one bond becomes better aligned, the rest of the protein toward the C-terminal swings away and increases the RMS deviation. CROSSOVER is not able to improve this situation for the same reason.
3. Only the LOCAL TWIST operator can improve a fit locally without disturbing well-fitting fragments that surround the mutation site. However, the applicability of LOCAL TWIST is mathematically constrained: when starting from a less-fitting conformation the optimal local improvement is not always found in one pass. Sometimes it is even impossible to improve a local conformation at all.

Hence, with an increasing number of generations it becomes more and more difficult to achieve any further improvement in the RMS fitness and the search stagnates at RMS deviation values between 0–2 Å (**Fig. 19**).

Another conclusion to draw from the foregoing experiments with the RMS fitness function is that the fitness function is the crucial topic. This is clearly an unresolved issue and the subject of ongoing research in protein engineering.

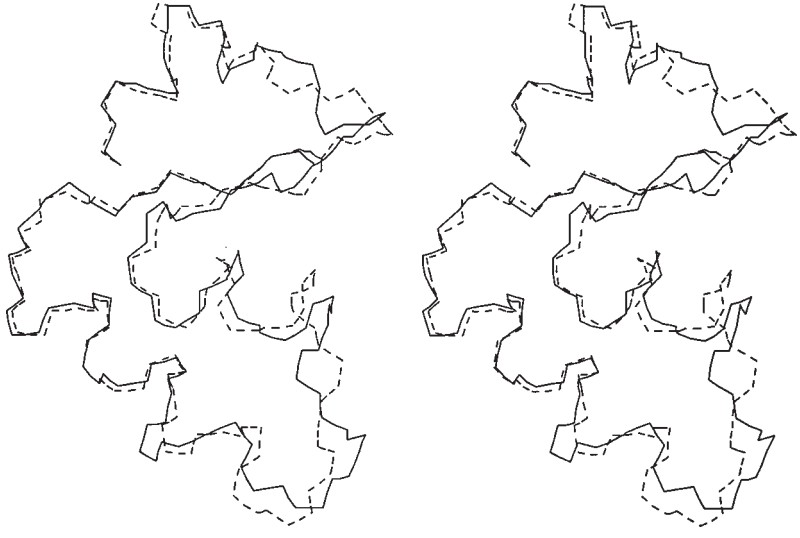


Fig. 16. Crambin predicted by the RMS fitness function. This conformation (solid line) with an RMS deviation of 1.08 Å to native Crambin (dashed line) was obtained after 10,000 generations using the LOCAL TWIST, MUTATE, VARIATE, and CROSSOVER operators and RMS deviation as the fitness function.

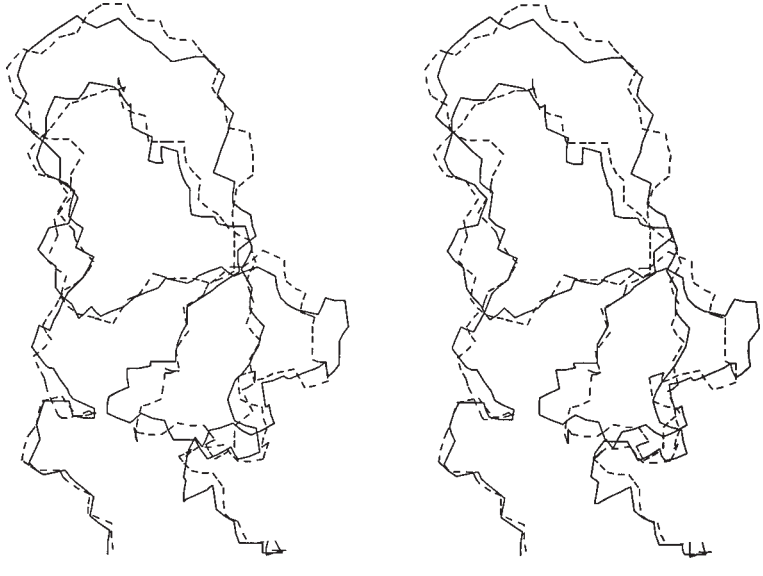


Fig. 17. Trypsin inhibitor predicted by the RMS fitness function. Stereoscopic superposition of the native conformation (dashed line) and one individual of the final generation (solid line). The RMS deviation is 1.48 Å.

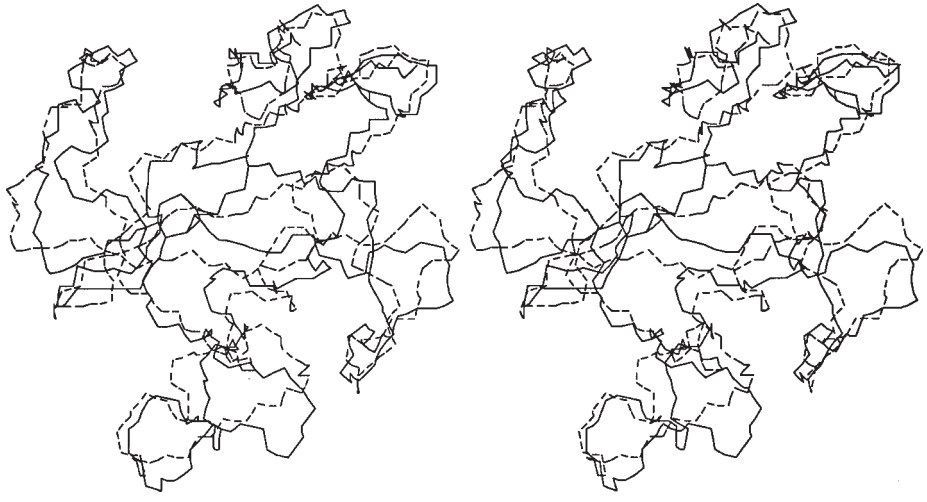


Fig. 18. RNase T1 predicted by the RMS fitness function. Stereoscopic superposition of the native conformation (dashed line) and one individual of the final generation (solid line). The RMS deviation is 2.32 Å.

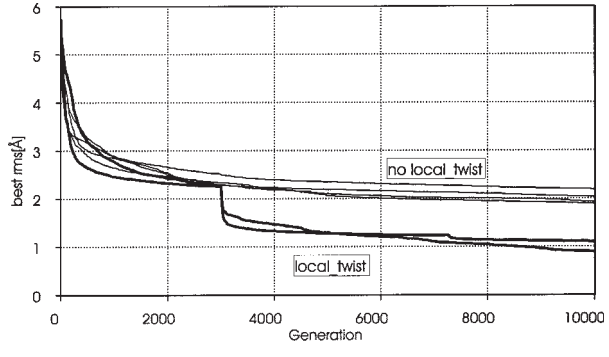


Fig. 19. Performance comparison for the LOCAL TWIST operator. This graph shows the course of six single experiments with the RMS deviation as the fitness function. The individual with the best RMS deviation is plotted for each generation. The two thicker lines at the bottom have the LOCAL TWIST operator switched on after 3000 generations. Reproduction was done by the roulette wheel algorithm. The four runs without LOCAL TWIST had a population size of 54 individuals, whereas the two runs with LOCAL TWIST had only 30.

Some aspects of the computational complexity have already been explained. This situation led to the following experiments with the genetic algorithm and a multivalue vector fitness function.

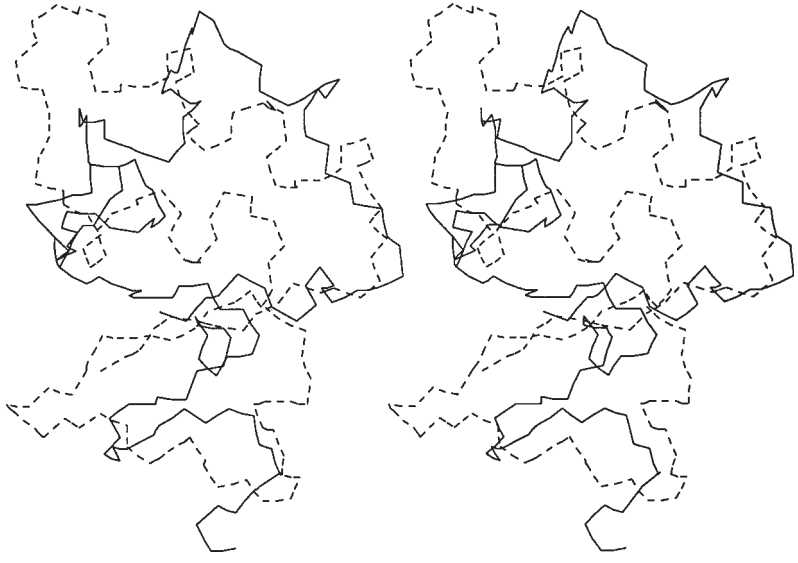


Fig. 20. Individual of the final generation of a multi-value fitness run. Only the fitness components *polar*, E_{pe}, E_{tor}, E_{el}, *hydro*, *Crippen*, and *solvent* were used to guide the genetic algorithm in this run. There is a vague similarity (RMS 6.27 Å) in the overall backbone fold of the generated individual (solid line) to native Crambin (dashed line).

Figure 20 shows the results of a run with the fitness components *polar*, E_{pe}, E_{tor}, E_{el}, *hydro*, *Crippen*, and *solvent*. This individual had an RMS deviation of 6.27 Å from the native conformation of Crambin. The genetic algorithm did not use the RMS deviation as part of the fitness function. Only the fitness components listed above were used to guide the genetic algorithm. Over the whole run, some of the fitness components decreased along with the RMS deviation (E_{pe}, *hydro*, *Crippen*, *solvent*), as was expected. However, the other fitness components (*polar*, E_{tor}, E_{el}) actually drove the genetic algorithm to conformations with less similarity to the native Crambin indicating that these propensities were no good indicators for the "nativeness" of Crambin. In general, no RMS values better than around 6 Å were detected in similar runs.

The following conformations were generated with the fitness components *Crippen*, *clash*, *hydro*, and *scatter*. In addition, constraints on the secondary-structures of Crambin were imposed by limiting the backbone angles to intervals between the upper and lower values of **Table 7**. Torsion angle ω was constrained to 180°. For a general application, the use of secondary structure constraints requires a highly accurate and reliable secondary structure prediction algorithm that, unfortunately, does not (yet) exist. **Figure 21** shows the backbone of an individual generated by the genetic algorithm with the afore-

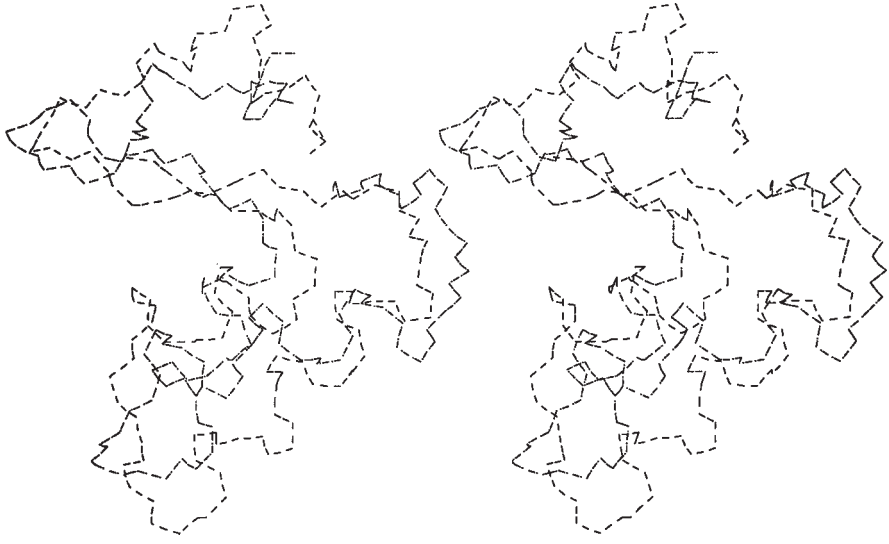


Fig. 21. Folding Crambin with secondary structure constraints. The backbone of the predicted conformation (solid line) and Crambin (dashed line) have only an RMS deviation of 4.36 Å. For this run, only the fitness components *Crippen*, *clash*, *hydro*, and *scatter* were used in the multivalue vector fitness function.

mentioned fitness components and that has an RMS deviation from native Crambin of 4.36 Å.

Another run with the same fitness components was performed for trypsin inhibitor (**Fig. 22**). The RMS deviation from native trypsin inhibitor is 6.65 Å. This is worse than the result for Crambin in **Fig. 21** because the lower content of secondary structure in trypsin inhibitor implies less rigid constraints on the conformation. This means there are more degrees of freedom, and therefore a larger search space to traverse.

4. Notes

Summarizing these findings and those of the previous subsections we are led to the following conclusions.

1. Genetic algorithms proved to be an efficient search tool for both 2D and 3D representations of proteins. In a 2D protein model, the genetic algorithm outperformed the Monte Carlo search in both the quality of the results and (less) required computation time. For a 3D protein model with a simple, additive force field as fitness function, and using a rather small population, the genetic algorithm produced several individuals (i.e., protein conformations) of dissimilar topology but each with highly optimized fitness values.



Fig. 22. Backbone folding of trypsin inhibitor. The backbone of the predicted conformation (solid line) and trypsin inhibitor (dashed line) have an RMS deviation of 6.65 Å. For this run, only the fitness components *Crippen*, *clash*, *hydro*, and *scatter* were used. The comparatively bad performance of the genetic algorithm in comparison to the run on Crambin (**Fig. 21**) is a result of the low content of secondary structure in trypsin inhibitor, which increases the number of rotational degrees of freedom.

2. Given an appropriate fitness function (for test purposes the RMS deviation from the *a priori* known conformation can be used) the genetic algorithm application described in this chapter finds the desired solution within only small deviations.
3. The major problem lies in the fitness function. If there were one index or a set of indices that return "1" for "the object is (part of) a native protein conformation" and "0" for "the object is not (part of) a native protein conformation," one could expect the genetic algorithm approach to deliver reasonably accurate *ab initio* predictions. However, neither mathematical models nor empirical, semiempirical, and statistical force fields are yet accurate enough to discriminate reliably native from nonnative conformations without additional constraints. Thus, the genetic

algorithm produces (sub-)optimal conformations in a different sense than that of "nativeness."

4. Because secondary structure in nature and J. H. Holland's building blocks in the genetic algorithm are analogous fundamental components for the construction of the individual, it was hoped that secondary structures would emerge as the building blocks in a subset of the population ***(1)***. This has not yet happened. One possible explanation is that the fitness functions used are not sensitive enough to detect and account for the structural benefits in secondary-structures.

References

1. Holland, J. H. (1973) Genetic algorithms and the optimal allocations of trials. *SIAM J. Comput.* **2,** 88–105.
2. Rechenberg, I. (1973) Bioinik, Evolution und Optimierung. *Naturwissenschaftliche Rundschau* **26,** 465–472.
3. Koza, J. (1993) *Genetic Programming*, MIT Press.
4. Kirkpatrick, S., Gelatt, C. D., Jr., and Vecchi, M. P. (1983) Optimization by simulated annealing. *Science* **4598,** 671–680.
5. Jones, T. and Forrest, S. (1993) An Introduction to SFI Echo, Santa Fe Institute, Santa Fe, NM. E-mail: terry@sanatfe.edu, forrest@cs.unm.edu. World Wide Web Server at ftp://alife. santafe. edu/pub/SOFTWARE.
6. Holland, J. H. (1993) Echoing emergence: objectives, rough definitions and speculations for echo-class models, in *Integrative Themes* (Cowan, G., Pines, D., and Melzner, D., eds.), Santa Fe Institute Studies in the Science of Complexity, Proc. Vol XIX, Addison-Wesley, Reading, MA.
7. Holland, J. H. (1992) Adaptation in Natural and Artificial Systems. 2nd Ed., MIT Press.
8. Davis, L. (ed.) (1991) *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.
9. Goldberg, D. E. (1989) Genetic algorithms, in *Search, Optimization & Machine Learning*. Addison-Wesley.
10. Unger, R. and Moult, J. (1993) Genetic algorithms for protein folding simulations. *J. Mol. Biol.* **231,** 75–81.
11. Grefenstette, J. J. Connect to <leland.stanford.edu>, log in as anonymous using your email address as the password, and change to the directory </pub/cs426/GA/GENESIS/genesis>. Then use `ftp` to download the software. Alternatively, use archie genesis or a Web search engine to find alternate locations of the file.
12. Schulz, G. E. and Schirmer, R. H. (1979) *Principles of Protein Structure*. Springer Verlag, New York.
13. Lesk, A. M. (1991) *Protein Architecture — A Practical Approach*. IRL Press at Oxford University Press, Oxford, UK.
14. Branden, C. and Tooze, J. (1991) *Introduction to Protein Structure*. Garland Publishing, New York.

15. Schulze-Kremer, S. (1992) Genetic algorithms for protein tertiary structure prediction, in *Parallel Problem Solving from Nature II* (Männer, R. and Manderick, B., eds.), North Holland, Amsterdam, pp. 391–400.
16. Dandekar, T. and Argos, P. (1992) Potential of genetic algorithms in protein folding and protein engineering simulations. *Protein Eng.* **7,** 637–645.
17. Le Grand, S. M. and Merz, K. M. (1993) The application of the genetic algorithm to the minimization of potential energy functions. *J. Global Opt.* **3,** 49–66.
18. Sun, S. (1994) Reduced representation model of protein structure prediction: statistical potential and genetic algorithms. *Prot. Sci.* **5,** 762–785.
19. Dandekar, T. and Argos, P. (1994) Folding the main chain of small proteins with the genetic algorithm. *J. Mol. Biol.* **236,** 844–861.
20. Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, E. F., Jr., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1997) The protein data bank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.* **112,** 535–542.
21. Vinter, J. G., Davis, A., and Saunders, M. R. (1987) Strategic approaches to drug design. An integrated software framework for molecular modelling. *J. Comput. Aided Mol. Des.* **1,** 31–51.
22. Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. (1983) Charmm: a program for macromolecular energy, minimization and dynamics calculations. *J. Comput. Chem.* **4,** 187–217.
23. Ngo, J. T. and Marks, J. (1992) Computational complexity of a problem in molecular-structure prediction. *Protein Eng.* **5,** 313–321.
24. Davis, L. (ed.) (1991) *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.
25. Lucasius, C. B. and Kateman, G. (1989) Application of genetic algorithms to chemometrics, in *Proceedings of the 3rd International Conference on Genetic Algorithms* (Schaffer, J. D., ed.), Morgan Kaufmann Publishers, San Mateo, CA, pp. 170–176.
26. Tuffery, P., Etchebest, C., Hazout, S., and Lavery, R. (1991) A new approach to the rapid determination of protein side chain conformations. *J. Biomol. Struct. Dyn.* **8,** 1267–1289.
27. Hendrickson, W. A. and Teeter, M M. (1981) Structure of the hydrophobic protein crambin determined directly from the anomalous scattering of sulphur. *Nature* **290,** 107.
28. Lee, C. and Subbiah, S. (1991) Prediction of protein side chain conformation by packing optimization. *J. Mol. Biol.* **217,** 373–388.
29. Tuffery, P., Etchebest, C., Hazout, S., and Lavery, R. (1991) A new approach to the rapid determination of protein side chain conformations. *J. Biomol. Struct. Dyn.* **8,** 1267–1289.
30. Maiorov, N. M. and Crippen, G. M. (1992) Contact potential that recognizes the correct folding of globular proteins. *J. Mol. Biol.* **227,** 876–888.

31. Go, N. and Scheraga, H. A. (1970) Ring closure and local conformational deformations of chain molecules. *Macromolecules* **3,** 178–187.
32. Lu, S. Y. and Fu, K. S. (1978) A sentence-to-sentence clustering procedure for pattern analysis. *IEEE Trans. Syst. Man Cybern.* **8,** 381–389.
33. Rost, B. and Sander, C. (1993) Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* **232,** 584–599.
34. Cármenes, R. S., Freije, J. P., Molina, M. M., and Martín, J. M. (1989) PREDICT7, a program for protein structure prediction. *Biochem. Biophys. Res. Commun.* **159,** 687–693.
35. Garnier, J., Osguthorpe, D. J., and Robson, B. (1978) Analysis of the accuracy and implications of simple methods for predicting the secondary structure of global proteins. *J. Mol. Biol.*, **120,** 97–120.
36. This is only available for test runs with known protein conformations.

10

Scoring Functions for *ab initio* Protein Structure Prediction

Enoch S. Huang, Ram Samudrala, and Britt H. Park

1. Introduction

The native conformation of a protein is generally assumed to be the one with the lowest free energy ***(1)***. The successful prediction of protein structure depends on the surmounting of three subproblems: (1) choosing a representation of protein conformation that includes structures similar to the correct conformation but limits the search space; (2) formulating a scoring function that relates a particular protein conformation to its free energy; and (3) devising a method to combine the first two elements in a search through conformational space for the state with the globally optimum score. These three requirements apply to the major classes of protein structure prediction: homology modeling, threading (fold recognition), and *ab initio* folding. In this chapter, we focus on the second of the three subproblems, that of developing energy functions, and place an emphasis on functions tailored for *ab initio* folding, although much of the discussion will also apply to threading.

The form of a scoring function is dependent on the particular type of problem to be tackled. For instance, in homology modeling, the backbone (or fold) of the target protein is assumed to be known, as it is derived from a related protein with known structure. A suitable function computes the total score for interactions between pairs of side chains, and side chains with the backbone, to build side-chain conformations. However, in threading and *ab initio* folding, one is primarily concerned with capturing the overall fold, or topology, of the backbone. For example, consider an *ab initio* folding scenario in which one starts with a fully extended polypeptide backbone and attempts to fold it with respect to some scoring function. In order to make the search problem more tractable by reducing the degrees of freedom afforded to the protein, side-chain

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

atoms are often reduced to a single coordinate ***(2)***, thereby decreasing the computational overhead; similarly, the applicable scoring functions are reduced in complexity. Threading techniques also use these simplified functions to score the alignment of probe sequences mounted on structures and substructures found in the Protein Data Bank (PDB) ***(3)***. Such functions are suitable because the original side-chain conformations of the template are discarded when a probe sequence replaces the identity of the residues.

Obviously, simplified functions cannot be rooted in the same physical principles as the all-atom functions used for the molecular simulation of proteins that require the explicit positions of all the atoms in the protein ***(4–7)***. Parameters for these potential energy functions, or force fields, are obtained from experimental data and quantum mechanical calculations. In contrast, most of the scoring functions used in protein structure prediction fall into the category of knowledge-based potentials of mean force ***(8,9)***. The term "knowledge-based" refers to the statistical analysis of the properties found within the database of experimentally determined protein structures. Knowledge-based functions mine the information-rich protein database by converting properties seen in native proteins into "pseudoenergies" that reflect the compatibility of a given sequence with a structure. A wealth of properties of native structures is readily extracted, for instance, the pairwise interaction of residues, the exposure of nonpolar groups to solvent, the propensity of sequences to form secondary-structure, and the close packing of protein atoms ***(10–13)***. The choice of the property is at the discretion of the modeler; hence, a knowledge-based function can be derived using a range of fold representations, from a string of secondary-structure assignments to a full-atom representation. Whereas simplified scoring functions are typically knowledge-based, the converse is not true.

Knowledge-based energy functions are not without problems in their theoretical justification ***(14–20)***. Although the details of this discussion are beyond the scope of this chapter, the main points are presented here. First, knowledge-based functions derive their parameter sets from experimental data, typically by applying the inverse Boltzmann equation to the observed properties in the protein database:

$$\Delta E = -kT \ln (f_1 / f_2) \quad (1)$$

where the energy difference ΔE between two states is related to the ratio of their occupancies (f_1 and f_2); T is the temperature (K) and k is the Boltzmann constant. f_1 is the frequency of observations of a certain type in the database, and f_2 is the number of observations expected by chance (defined by the chosen reference state, *see* **Subheading 2.1.4.**).

At least four assumptions underlie the application of the inverse Boltzmann equation in this fashion: (1) the set of known stable folds of different proteins

are representative of proteins in general; (2) the protein set represents a system at equilibrium; (3) the observed frequencies are independent of each other and their environment; and (4) the observed frequencies are distributed according to the Boltzmann equation.

However, Thomas and Dill have shown that interresidue interactions are not independent ***(17)***. Instead, the result of a dominating hydrophobic effect is to influence the types of interactions that polar residues make, simply because each structure can only make a limited number of interresidue contacts. For example, the extracted parameters for charged residues do not mainly reflect electrostatic interactions; charged residues are driven to the protein surface by the nonpolar interactions, coupled by chain connectivity and excluded volume effects. Also, Kocher et al. argue that because protein folding is cooperative, interresidue interactions cannot be independent ***(14)***. Finally, Thomas and Dill have shown that the size of the proteins used to compile the parameters can also skew the extracted scores ***(17)***.

To circumvent the need for the assumptions surrounding the conversion of database statistics to true energies, some methods rely instead on Bayesian formalism (i.e., conditional probabilities) to formulate a scoring function ***(21,22)***. The two formalisms are analogous and follow the same methodology in practice. We therefore refer to all knowledge-based functions discussed in this chapter as "scoring" or "objective" functions.

Given that there are a multitude of scoring functions designed for protein structure prediction by threading and *ab initio* folding, it is important to understand how they work. In **Section 2.**, we provide examples from work conducted in our laboratory and in the literature. We dissect out the essential components of scoring functions for *ab initio* folding, and compare and contrast the similarities and differences among them. Our intent is not to do an comprehensive review ***(8,9,16,23,24)*** but to stereotype the different components of the various scoring functions and explain their specific roles.

Ab initio folding methods can be largely placed into two main categories: fold generation by exhaustive enumeration or by minimization. Each of these classes can further be subdivided into lattice and off-lattice (torsion-based) approaches. We will look at an example of each of these four subtypes of *ab initio* folding methods. Threading functions will not be discussed explicitly, but many knowledge-based functions used in *ab initio* folding can be directly applied to evaluating sequence–structure compatibility in a threading context ***(10,12,13,25)***. However, successful threading or fold recognition is by no means limited to the knowledge-based functions described in this chapter. Many excellent alternatives exist, including methods that use environmental profiles ***(11)***, predicted secondary-structure ***(26–28)***, and multiple sequence alignment ***(29)***.

2. Methods

2.1. General Issues

Although all of the scoring functions discussed as follows were developed and tested for *ab initio* folding, some are exclusively knowledge-based. Some do not rely on the database of known structures but on model forces such as the hydrophobic effect and hydrogen bonding. Others combine the two approaches. For the knowledge-based functions, we discuss some general procedural issues.

2.1.1. Selection of a Database

The standard procedure for constructing a fold library to compile a scoring function is to choose a nonredundant set of proteins that reflect all known folds. One way to do this is to require that no two proteins in the set share more than 30% sequence identity. Undesired bias can arise from over-representing proteins of a certain size or topology (for instance, α-helical proteins), and thus a balanced mixture of proteins with different secondary-structures must be used. The set should also be as large as possible to make the observed statistics robust.

2.1.2. Jackknifing

Development and validation of a scoring function must proceed without specific knowledge of the target protein. A true threading or *ab initio* experiment would be carried out only in the absence of a template structure with suitably high sequence similarity (otherwise the problem shifts from fold recognition/generation to homology modeling). Thus, validation of a given scoring function for use in threading or *ab initio* folding must ensure that no inadvertent use of information occurs. One commonly employed technique is that of "jackknifing." Consider the case where parameters for a scoring function is extracted from a database of 300 proteins. Presumably, the parameters reflect the tendencies of native proteins in general with respect to some property of interest (for instance, frequencies of pairwise contacts), but in reality the parameters will be biased in some degree toward the 300 proteins. In practice, this implies that one cannot validate the scoring function on a test set of proteins that includes any of the proteins used to compile the parameters (or any related proteins thereof). Furthermore, training or optimizing a scoring function with respect to performance on a fixed test set, whether the database was previously jackknifed or not, is tantamount to introducing knowledge of the test set.

2.1.3. Correction for Sparse Data

If one is extracting many properties from the database, the problem of sparse data arises. Sippl ***(10)*** suggests the following correction for the observed frequency of sequence *s* in structural state *c:*

$$\rho'_{s,c} = \frac{1}{\rho/\sigma + m_s}(\sigma\rho_c + m_s\sigma_{s,c}) \quad (2)$$

where m_s is the number of occurrences sequence s appears in the database and $r_{s,c}$ is the unadjusted frequency that sequence s appears in structural state c. The effective sequence-dependent frequency $\rho'_{s,c}$ is equal to a combination of the sequence-independent frequency ρ_c and the actual number of sequence-dependent occurrences of structural state c. The adjustable parameter σ sets the relative weight of the sequence-independent term (chosen as 50 in **ref. *30***). This correction for sparse data is most commonly employed when one is generating potentials of mean force in at various sequence separations (*see* **Subheading 2.2.2.**).

2.1.4. Choice of a Reference State

Knowledge-based scoring functions express their pseudoenergies relative to a reference state. For example, a reference state might represent a system in which the actual interaction energy between residue pairs equals zero; i.e., a system exhibiting the contact frequencies of a randomly interacting system. This state may or may not include explicit solvent molecules, the presence of which dramatically affects the resulting effective energy of interaction between two residues. Because there are many ways to formulate a reference state ***(20)***, this issue is individually addressed where applicable.

2.2. Exhaustive Enumeration Methods

2.2.1. A Lattice Model

In a study by Hinds and Levitt ***(31)***, all possible conformations of a sequence were generated, subject to the bounds, spacing, and geometry of the lattice. The knowledge-based scoring functions used by the authors had the following functional form:

$$E = \sum_{\text{contacts}} e_{ij} \quad (3)$$

where e_{ij} is the contact score between residues types i and j and the total score E is the sum of all pairwise scores observed in the lattice structure. These so-called contact functions typically are square-welled, i.e., the interaction between a pair of residues is value e_{ij} if the residues are within a cutoff distance (6–8 Å is customary) and zero otherwise.

The parameters for the 210 values for e_{ij} (i.e., in a 20 × 20 symmetrical matrix) are calculated as

$$e_{ij} = N_{ij}^{\text{obs}} / N_{ij}^{\text{exp}} \quad (4)$$

where N_{ij}^{obs} is the number of observed contacts between residue types i and j. In the selected database and N_{ij}^{exp} is the number of contacts made in the reference state, or

$$N_{ij}^{\exp} = \sum_{p} C_p \, (T_{ijp} / T_p) \tag{5}$$

where p is a protein in the database, T_p is the total number of possible tertiary contacts, and C_p is the number of actual tertiary contacts. The total number of contacts T_p is a simple function of the total number of residues in protein p, N_p:

$$T_p = (N_p - 4)(N_p - 5) \tag{6}$$

T_p is not exactly equal to N_p^2 because interactions between nearest neighbors along the sequence ($|i - j| < 5$) are disregarded. T_{ijp} is equal to the number of i and j pairs that are not nearest neighbors in the sequence. The ratio T_{ijp}/T_p is effectively the product of the concentrations of i and j. Contacts in the database are counted whenever a heavy atom of one residue is within 4.5 Å of a heavy atom of another residue.

This technique of recovering effective contact energies from the database is also referred to as the "quasichemical approach" ***(2,20)***. Briefly, this approximation treats the interacting centers (e.g., residues) as disconnected units that interact randomly and whose expected (or reference) contact frequency is proportional to their relative concentrations. This particular method uses a reference state with the compactness and packing patterns of native proteins.

The goal of exhaustive *ab initio* methods is achieved when the fold closest to the native structure corresponds to the global energy minimum. If there is more than one fold that resembles the native fold to within some root-mean-square (RMS) or distance matrix error (DME) cutoff, then ideally that subset of folds has better scores than all the other, nonnative folds.

The tetrahedral lattice of Hinds and Levitt is a coarse lattice in that it is only able to generate walks suggestive of the overall native trace ***(31)***. On the other hand, this lattice can support exhaustive enumeration of most small proteins. The number of total walks is therefore very large (on the order of 10^7). Hinds and Levitt ***(31)*** did not report the rank of the nearest-native fold in the ensemble, but they note that out of the lowest-energy 10^3–10^4 folds, there are on the order of 10 nativelike folds (4–5 Å DME).

2.2.2. An Off-Lattice Model

Next, we examine the four-state off-lattice model of Park and Levitt ***(30,32)***. By using only four states in Ramachandran space, one can reproduce the native fold to about 2 Å RMS error. Unfortunately, exhaustive enumeration of a small protein (100 residues) implies 4^{100}, or 10^{60} conformations, which is intractably large. However, if one enforces idealized native secondary-structure (i.e., one state each to represent α and β states), allowing only 10 selected loop and turn residues to assume the four possible (ϕ, ψ) possibilities, then one only needs to contend with 4^{10} folds (about a million). After applying a generic compactness

Table 1
Performance of Four Selected Energy Functions

Function	Z-score
Histogram	–1.27
Shell	–1.78
Contact (MJ)	0.03
HF	–1.51

Four energy functions described by Park et al. ***(33)*** were tested on eight semiexhaustive off-lattice decoy sets. The average Z-score for the near-native folds (those within 4 Å RMS error of the native fold) is shown for each function.

filter, only approx 200,000 structures remain. One may think of this fold ensemble as the set of all possible arrangements of native secondary structure.

As in the case of Hinds and Levitt ***(31)***, for a given set of conformations there were many (on the order of 10^2) near-native folds (≤4 Å RMS deviation from the native structure) present. Park et al. ***(33)*** evaluated a series of scoring functions by computing a Z-score (defined as the number of standard deviations a particular score departs from the mean score in the set) for each near-native fold. The best functions had the most negative average Z-scores for the near-native folds (a Z-score ≥ 0 means that the function did not discriminate better than random for that structure). **Table 1** lists some representative functions and their average Z-scores for eight small proteins. Park et al. ***(33)*** also reported that for many functions, one of the near-native folds would rank very high in the score-sorted list. For instance, the Shell function placed a near-native fold within the top 100 of every fold ensemble for eight different proteins (corresponding to the top 0.1 to 1% of a score-scored list). However, many nonnative folds were also among the lowest-scoring conformations in each set, even though the near-native folds overall were favored. In other words, none of the simplified knowledge-based functions could identify near-native folds without also including some nonnative folds.

The Shell function, the top performer out of our representative set of four functions is a simple contact function. Whenever a pair of residues that is more than one residue apart in the sequence is within 7 Å, a score e_{ij} is counted. Nearest neighbors in the sequence are ignored simply because they are always in close spatial proximity with each other, and hence should not contribute to the signal. Residues are reduced to a single "interacting center" (or virtual centroid) 3 Å from the C_α atom along the C_α–C_β vector.

The 210 parameters e_{ij} are computed essentially in the same manner as described in **Subheading 2.2.1.**, in that a compact, randomly mixed reference state is employed.

Two subtle differences are:

1. The Shell function counts residues in contact (both in the database and in the set of *ab initio* folds) when their virtual centroid positions are within 7 Å of each other.
2. The value of T_p for the Shell function reflects the smaller sequence separation cutoff for interacting residues (only $|i - j| < 2$ are ignored).

The Histogram function is an implementation of the Sippl ***(10)*** potential of mean force (PMF). Unlike contact functions, which typically apply the quasi-chemical approximation in an explicit reference state, a PMF uses an implicit reference state (*see* next paragraph). The potential of mean force W between two interacting centers (e.g., residues) i and j is defined as:

$$W_{ij}(r) = -kT \ln (\rho_{ij}^{obs}(r) / \rho(r)) \quad (7)$$

where ρ_{ij} is the observed probability density that residues i and j are at distance r.

The reference state is a hypothetical state for the polypeptide that reflects the observed interresidue distances of the database with sequence information removed. As in the case for contact functions via the quasi-chemical approximation, the energy parameters are extracted from the observed amino acid distributions in a subset of the PDB. This function is named the Histogram function because it relates the energy of interaction as a function of observed interresidue distances (calculated as the distance between the C_β atoms). Hence, instead of recovering 210 pairwise contact parameters for the 20 amino acids, 210 histograms are generated. Each histogram reflects the relative frequency of interresidue distances sampled at 20 uniformly spaced intervals. Furthermore, the classic implementation of the Sippl function ***(10)*** involves modeling the role of local and long-ranged pairwise interactions by generating separate histograms for pairs of residues at a given separation along the polypeptide chain (called a topological level).

In the Park and Levitt ***(30)*** implementation of the PMF described by Sippl ***(10)***, 10 histograms for each of the residue–pair interactions were generated in the following manner: 8 for local interactions with sequence separation 3–10, inclusive; 1 for medium range interactions (sequence separation 11–50, inclusive); and 1 for all other long-range interactions.

Spatial distance bins were computed for each histogram by storing the minimum and maximum distances and dividing the range into 20 equal distance bins. The correction for sparse data was applied (*see* **Subheading 2.1.3.**). If there is no sample in a particular bin, its occurrence was reset to one to prevent the computed energy from going to infinity. Fortunately, these slots correspond to geometries that are very unlikely in real proteins.

The last of the three knowledge-based functions is called the Contact(MJ) function. This function differs from the Histogram and Shell functions in one key respect: the reference state is a random mixture of solvent and amino acids, which directly models the effect of desolvation in protein folding. A quasichemical reaction between two amino acids *i* and *j* in solvent can be expressed as:

$$i - 0 + j - 0 \Leftrightarrow i - j + 0 - 0 \tag{8}$$

where 0 represents a solvent molecule. The effective energy of desolvation and contact formation e_{ij} is determined by separate terms for the effective energy of breaking the $i - 0$ and $j - 0$ interactions and forming $i - j$ and $0 - 0$:

$$e_{ij} = e'_{ij} + e'_{00} - e'_{i0} - e'_{j0} \tag{9}$$

Each energy parameter e' is determined by the same equation used in the Shell energy function.

The effects of introducing solvation-dependent energies e'_{i0}, e'_{j0}, and e'_{00} include:

1. Desolvation energies that are more favorable for polar and charged residues than hydrophobic residues.
2. The introduction of a favorable solvent interaction term, e'_{00}, which causes all the energy terms to be more favorable (more negative) by a constant.

Each of the e' terms is computed separately using **Eq. 4** ***(31)***. To extract the parameters, the following are required:

1. Each residue type *i* has an average coordination number q_i, estimated by scanning the database of known structures for buried residues of type *i*. When q_i is greater than the number of interresidue contacts made by a particular residue *i*, then the difference is set to the number of contacts between residue *i* and solvent.
2. The number of solvent molecules is a free parameter equal to twice the number of residues in the protein ***(30)***.
3. The total number of contacts in the system is equal to T_p plus the number of solvent contacts. The coordination of water was set to the the average residue coordination number.
4. The total number of solvent–solvent contacts is equal to the number of solvent contacts minus the total number of residue–solvent contacts.

The hydrophobic fitness (HF) function ***(34)*** is unusual in that it derives no parameters from the PDB. Instead, it simply rewards favorable arrangements of hydrophobic and polar residues. A conformation is scored favorably if hydrophobic residues (of any type) make more contacts with other hydrophobic residues than would be expected on average. The over-

all score is weighted by a term that rewards the burial of hydrophobic residues. The form of the HF function is:

$$\mathrm{HF} = -(\sum_i B_i)(\sum_i [H_i - H_i^\circ]) \quad (10)$$

where i is hydrophobic {C, F, I, L, M, V, W}; B_i is the number of virtual side-chain centroids within 10 Å; H_i is the number of hydrophobic residues (plus Y) with 7.3 Å. H_i° is the expected number of hydrophobic contacts based on a random distribution of the other residues surrounding residue i, disregarding the nearest neighbors in the sequence. H_i° is computed by multiplying the fraction of hydrophobic residues with the number of contacts residue i makes.

2.3. Minimization Methods

Scoring functions take on different forms when structure prediction is attempted on a lattice by minimization protocols. When one is not concerned with exhaustive enumeration, a finer lattice may be used, thereby improving the accuracy to which a native fold may be represented. The trade-off is that one can never be sure that the best fold can be found by minimization, either because of imperfections in the energy function, search strategy, or both.

Unlike scoring functions used in exhaustive methods, a scoring function used in minimization must bear the additional burden of favoring generic features of native states, namely secondary-structures and compactness. Exhaustive methods can enforce compactness simply by setting the bounds of a lattice or by simply discarding structures that do not satisfy a radius of gyration cutoff. In contrast, minimization starts with a random or extended state, and compactness must be monitored by at least one component of the scoring function. The problem of secondary-structure formation may be surmounted by imposing native secondary-structure assignments. Otherwise, a combinatorial explosion in the search process is averted by biased sampling of conformational space ***(21)*** or by ad hoc terms in the energy function favoring secondary-structure formation (for instance, via hydrogen bonding). The implementation of these terms is typically specific to a given structural representation (e.g., lattice models with a certain geometry and spacing), so we will not discuss the functional forms or parameter derivation at length unless they are illustrative of general issues.

2.3.1. Minimization on a Lattice

In the study by Kolinski and Skolnick ***(35)***, a dual lattice model was used for folding by optimization of a scoring function. In their scheme, a coarse lattice was used for the early stages of folding from an expanded state, and refinement of the initial structures was performed on a finer lattice. The entire scoring function is written as:

$$E = E_{C_\alpha} + E_{\text{H-bond}} + E_{\text{rot}} + E_{\text{sg-local}} + E_{\text{one}} + E_{\text{pair}} + E_{\text{tem}} \quad (11)$$

and can be divided into three components: sequence-independent terms, sequence-dependent local and long-range terms, and multibody side-chain interactions. $E_{C\alpha}$ and $E_{\text{H-bond}}$ are the two sequence-independent terms. $E_{C\alpha}$ acts as an effective Ramachandran potential. Every i, $i + 3$ inter-C_α distance and its corresponding chirality (defined by the three intervening virtual C_α–C_α bonds) are compared with those extracted from the PDB. The resulting energy term enforces local geometries that favor secondary-structure formation. The second generic term, $E_{\text{H-bond}}$, models H-bond formation based on pairs of C_α–C_α vertices that are 4 or more residues apart in the sequence. A hydrogen bond between C_α vertices i and j must satisfy the following geometrical restrictions:

$$\begin{aligned} |(\mathbf{b}_{i-1} - \mathbf{b}_i) \cdot \mathbf{r}_{ij}| &\le \alpha_{\max} \\ |(\mathbf{b}_{j-1} - \mathbf{b}_j) \cdot \mathbf{r}_{ij}| &\le \alpha_{\max} \end{aligned} \quad (12)$$

where $\mathbf{b}$ is a backbone vector, $\mathbf{r}_{ij}$ is the vector between the C_α positions, and $\alpha_{\max}$ is a parameter set by the lattice spacing. An H-bonding cooperativity term rewards the formation of hydrogen–bond networks by adding to subtotal a separate score when consecutive sets of residues i, j and $i \pm 1$, $j \pm 1$ are hydrogen-bonded.

For sequence-specific energy terms, a simplified representation of side-chain rotamers (a single interaction center) was used. The energy of a given rotamer was simply tied to the frequency of that rotamer in the library (E_{rot}). The angle θ between two consecutive C_α-side-chain vectors was computed and scored as

$$E_{\text{sg-local}} = -\ln(\cos \theta^{\text{obs}} / \cos \theta^{\text{exp}}) \quad (13)$$

where the expected occurrence assumes a uniform distribution of states. $E_{\text{sg-local}}$ refers to the local interaction of side groups (or side chains).

The long-range interactions include a one-body term (E_{one}) and a pair potential (E_{pair}). The former is designed to drive hydrophobic residues into the interior of a folded chain. This term is designed to penalize extended states and addresses a central need of all minimization methods to drive the collapse of a polypeptide chain. This single-body term takes on two forms, one that is related to the position of a given residue from the center of mass of the polypeptide chain and a second that considers the number of contacts it makes relative to the average number for that residue in the database. The pair potential has a repulsive term that chases steric clash between side chains and other side chains and the main chain and a statistically derived scoring function similar to those described elsewhere in this chapter. The cutoff distances for repulsion and pairwise interaction are dependent on the residues involved. The strength of attraction is modulated by a factor f that reflects the average backbone orientation of the secondary-structures:

$$f = 1.0 - [\cos^2(\mathbf{u}_i, \mathbf{u}_j) - \cos^2(20°)]^2 \quad (14)$$

where $\mathbf{u}_i = \mathbf{r}_{i+2} - \mathbf{r}_{i-2}$ with $\mathbf{r}_i$ being the position of the ith C_α vertex. The maximum of this function occurs at 20° and the minimum at 90°.

Finally, the multibody term E_{tem} was added to simulate the cooperativity of side-chain packing from a molten state to a more nativelike state. The authors note that in the absence of this term, the folds have the character of molten globules, i.e., with well-defined secondary structure, but more expanded than close-packed tertiary structures. The multibody term assumes the following form:

$$E_{\text{tem}} = (e_{ij} + e_{i+k,j+n})C_{ij} \times C_{i+k,j+n} \quad (15)$$

where $C_{ij} = 1$ when residues i and j are in contact and residue spacing $|k| = |n|$; k and n assume values of $\{\pm 3, \pm 4\}$.

The relative strength of each of these contributions was set by requiring that the secondary structure be more prevalent in the collapsed states than in unfolded conformations.

Starting from a random configuration on the coarse lattice, folding was attempted for three small proteins ***(36)***. In the interest of conciseness, we focus on the folding of protein A, in many ways the most successful experiment of the three. The 60-residue fragment of this protein adopts a three-helical bundle topology. Folding of this protein was carried out 45 times using a simulated annealing protocol on the coarser lattice. In 19 trials the correct three-helical conformation was seen; in another 11 trials, a three-helical bundle of incorrect topology persisted. Overall, the average conformational energy of the correct folds was lower than that of the incorrect folds, and the reproducibility of the nonnative folds was much lower than for the nativelike folds. Further refinement of five near-native folds on the finer lattice yielded structures in the 2–3 Å RMS error range (excluding the residues at the N and C termini).

Note that evaluation of a scoring function *per se* in minimization experiments is difficult because the observed performance is dependent on the search strategy as well as the function used. Generally speaking, the best methods available today can provide nativelike folds in a significant fraction of the folding trials, as was the case for protein A summarized earlier. However, successful convergence to a nativelike fold is still limited to a handful of proteins.

2.3.2. Folding in Torsion Space

For our example for minimization in torsion space, we choose the work by Sun and co-workers ***(37)***. As with many *ab initio* methods, the authors rely on the constraint of native secondary-structure in order to overcome the vast conformational search problem. Unlike the exhaustive enumeration strategy of Park and Levitt ***(30)***, this minimization method has large dihedral library with

which to place the rigid secondary-structure elements. As a first step, the conformational search is powered by a genetic algorithm ***(38)*** that operates on a string of paired (ϕ, ψ) dihedral angles. A second step then refines the search by choosing a random adjustable residue and changing the torsion angles incrementally to probe the local energy surface for minima. The protein chain is reduced to a backbone with ideal bonds and angles and *trans* peptide conformations, and side chains are represented by a virtual atom centered at the average rotamer observed in the PDB.

The scoring function of Sun, et al. ***(39)*** could afford to be much simpler than the one described. Because native secondary structure was already in place, the energy terms favoring secondary structure formation were rendered unnecessary. In fact, their scoring function is surprisingly simple, as it relies mostly on hydrophobic interactions balanced by steric repulsion:

$$E_{\text{Total}} = E_{\text{HH}} + E_{\text{ex}} \tag{16}$$

where E_{HH} is an attractive interaction between hydrophobic residues {A, C, I, L, M, F, W, Y, V}. The magnitude of the attraction is distance-dependent, but the functional form is an analytical expression rather than a database derivation like the Histogram function (*see* **Subheading 2.2.2.**). The expression is:

$$E_{\text{HH}} = \sum_{i} \sum_{j>i+1} e_{ij} f(d_{ij}) \tag{17}$$

where e_{ij} is –1 if and only if i and j are hydrophobic residues and zero otherwise and d_{ij} is the distance between side-chain centroids of residues i and j. The coefficient f modulates the attraction by the following sigmoidal function:

$$f(d_{ij}) = \frac{1.0}{1.0 + e^{(d_{ij} - d_0)/d_t}} \tag{18}$$

d_t is a parameter that sets the sharpness of the sigmoidal function (set to 2.5 Å) and d_0 sets the interaction distance (6.5 Å) as the midpoint of the curve. The attraction is set to zero at 12 Å.

The excluded volume term is also a sigmoidal function between pairs of C_α atoms or side-chain centroids:

$$E_{\text{ex}} = \text{C} \times \sum_{ij} \frac{1.0}{1.0 + e^{(d_{ij} - d_{eff})/d_w}} \tag{19}$$

where d_w is 0.1 Å and d_{eff} is 3.6 Å for C_α atoms and 3.2 Å for side-chain centroids. The constant C sets the scale for the repulsive term higher relative to the attractive term.

To aid the formation of β-sheets during the folding process, a score of –1.0 for hydrogen bonding between β-strands was added for every instance when certain geometical conditions were met (O–H distance < 2.5 Å and N–H–O angle between 120° and 180°).

The method was tested on 10 small proteins. Of these, four of the lowest-scoring models were within 4 Å RMS error. The authors did not report the scores of the most nativelike folds the representation could allow in terms of RMS error, but 8 of the native structures had scores less favorable than the structures found by the genetic algorithm.

2.4. Extending Knowledge-Based Functions to the Atomic Level

Regardless of the initial fold representation used, protein structures are most useful when detailed atomic coordinates are known. Although simplified scoring functions are capable of distinguishing near-natives from nonnatives a significant fraction of the time, they will not work as well in situations where subtle differences between different conformations exist. To capture the finer details of atom–atom interactions in proteins, such as interactions between side-chain atoms and the rest of protein, a more detailed representation is necessary. For example, in a situation where two conformations are quite similar to the experimental structures (within 1–3 Å RMS error for the C_α atoms), we need all the information we can possibly obtain from the two conformations to determine which one is more accurate. A one-point-per-residue scoring function may not be able to discriminate as well as an all-atom discriminatory function, which takes into account the environment of all the atoms of the main and the side chain of each residue.

The all-atom probability discrimination function (PDF) as formulated by Samudrala and Moult ***(22)*** is similar to potential of mean force by Sippl ***(10)***, but the formulation is in Bayesian terms, and there is greater detail in the representation. There are 167 different atom types used. Scores for interactions between pairs of atoms for all 167 × 167 possible pairs and for 18 distance ranges (0.3,3–4,4–5,..,19–20 Å) are compiled using the expression:

$$s(d_{ab}|C) = -\ln P(d_{ab}|C) / P(d_{ab}) \tag{20}$$

$s(d_{ab}|C)$ is the conditional probability of observing two atoms a and b interacting at a distance d in a correct/native conformation C. $P(d_{ab}|C)$ is the probability of seeing atom types a and b in distance bin d in a correct conformation and is calculated by:

$$P(d_{ab}|C) = N(d_{ab}) / \sum_{d} N(d_{ab}) \tag{21}$$

$P(d_{ab})$ is the probability of seeing atom types a and b in the distance d in any conformation, correct or incorrect:

$$P(d_{ab}) = \sum_{ab} N(d_{ab}) / \sum_{d} \sum_{ab} N(d_{ab}) \tag{22}$$

$N(d_{ab})$ is the number of occurrences of a, b pairs in distance bin d.

A scoring function S proportional to the negative log-conditional probability of conformation being correct is used to calculate the total score of a conformation, given a set of i,j interatomic distances:

$$S(\{d_{ab}^{ij}\}) = \sum_{ij} s(d_{ab}|C) \tag{23}$$

The PDF we have described avoids sparse data problems by not separating local and nonlocal interactions. Although this leads to an averaging of the two sorts of environments in the parameters for the scoring function, it does not appears to diminish predictive ability ***(22)***.

In Bayesian terms, the reference state d_{ab} is referred to as a "prior distribution." In this case, the prior distribution is that found in the set of possible compact conformations, with the assumption that averaging over different atom types in experimental conformations is an adequate representation of the random arrangements of these atom types in any compact conformation.

Samudrala and Moult ***(22)*** have shown that discrimination between native and nonnative folds deteriorates as the detail in fold representation is reduced. To illustrate that point here, we run a detailed all-atom scoring function that takes into account interactions between all 167 pairs of atoms, and another function that uses only C_β–C_β interactions, for two sets of protein structure conformations. The first is a set of 269 conformations of 434 repressor (PDB entry 1r69) ranging in RMS deviation (RMSD) from 0.95–14.95 Å. The second is a set of "deliberately misfolded structures" created by Holm and Sander ***(39)***. In the latter case, 26 "misfolded conformations" are created by placing the sequences of the proteins on completely different structures of identical length, and then energy minimizing them to make them look as proteinlike as possible. These misfolded conformations range from 8.66–22.43 Å RMSD with respect to the corresponding native structures.

Table 2 gives the results for the two types of scoring function for 1r69 set of conformations, and **Table 3** gives the results for the two types of functions for the misfolded decoy set.

In the case of the 1r69 decoy set, even though the C_β–C_β scoring function does quite well, the best scoring conformation selected by the all-atom function is slightly lower in RMS error, and there is a better correlation between the score of the conformation and the RMS error to the native conformation. The Z-score for the single conformation below 1.0 Å and the 27 conformations below 2 Å is also slightly better in case of the all-atom function.

Given the results in **Table 2**, it might seem better to use a reduced representation to speed up the calculation of the fitness of a conformation, as the detailed representation is only slightly better. When we examine the results in **Table 3**, we see that, for the 26 misfolded structures, the all-atom function is able to identify all the 26 misfolded conformations as being incorrect, with a signifi-

Table 2
Comparison of the All-Atom and Scoring C_β–C_β Functions for a Set of 269 Conformations of 434 Repressor (PDB entry 1r69)

	RMSD of best scoring structure	Correlation between score and RMSD	Z score (1 and 2 Å cutoff)
All-atom	1.67 Å	0.80	–1.75/–1.42
C_β–C_β	1.80 Å	0.63	–1.65/–1.32

For each function, the root-mean-square deviation (RMSD) of the best scoring conformation, the correlation between the scores and the RMSD of the conformation with that score, and Z-score, using two different cutoffs to identify near-natives, is given. The detailed all-atom function performs slightly better than the C_β–C_β scoring function.

Table 3
Comparison of the All-Atom and C_β–C_β Scoring Functions for a Set of 26 Deliberately Misfolded Structures

	Percent of structures correctly discriminated	Average discrimination ratio
All-atom	100%	0.38
C_β–C_β	77%	0.66

For each function, the percentage of structures correctly discriminated and the average discrimination ratio (score of the incorrect conformation divided by the score of the correct conformation; the lower the ratio, the better the discrimination) is given. The all-atom function performs significantly better than the C_β–C_β function.

cant degree of discrimination (the ratio is the score of the incorrect conformation divided by the score of the correct conformation; the lower the ratio, the greater the discrimination). However, the C_β–C_β scoring function is unable to correctly identify 6 of the 26 structures as being nonnative, and the average discrimination ratio is poor relative to the ratio for the all-atom scoring function.

Although a function should be able to do more than just discriminate native conformations from nonnative ones, this results indicates that, in an exhaustive or semiexhaustive folding simulation, the simplified scoring function is more likely to fail, as it is unable to tell a native structure from a conformation that is significantly different in this simple test.

From these and other similar tests, it appears that taking into account as much information as is available in a protein conformation enables one to achieve better near-native discrimination. Given that it is not too difficult to generate all-atom models from approximate representations ***(40,41)***, the all-atom scoring function is an useful tool for protein structure prediction.

2.5. Summary

A well-suited scoring function for *ab initio* folding represents the most nativelike conformation as more favorable than all other nonnative ones. Current methods do not entirely succeed in this regard, as nonnative folds have scores that are as good as the near-native candidates, thereby presenting false positives in exhaustive sampling or traps in minimization. In general, functions that employ compact reference states are more effective when selecting near-native folds from sets of compact folds.

The style of protein structure prediction largely dictates the functional forms and components necessary to compute the score of a conformation. A complete minimization without external constraints generally requires terms that enforce secondary-structure and compactness along with pair-specific interactions. However, applying a biased conformational search based on sequence information ***(21)*** can greatly reduce the complexity of the energy function necessary to recover a significant number of native-like folds by minimization.

The success of the binary (hydrophobic and polar) functions suggests that most of the specificity of the knowledge-based functions, at least with respect to reduced representations, is due to the frequent occurrence of hydrophobic contacts in the interior of native proteins. However, this success was observed in the context of tertiary fold recognition; the native secondary-structure was already in place.

The use of all-atom scoring functions for selecting near-native folds bears promise. To overcome the computational overhead involved in using an all-atom function, one approach could involve sampling large amounts of conformational space using a simplified fold representation and selecting the top scoring conformations using a simple and fast scoring function. All-atom coordinates for these conformations can then be built, and the best conformations selected using the all-atom function. This complementary method of structure prediction would reduce the number of false positives selected by the simplified function and help avoid local minima traps.

3. Notes

3.1. Generic Simplified Energy Functions

3.1.1. Interaction Centers

Contact functions may vary with respect to their designated "interaction centers." Park et al. ***(33)*** test contact energy functions that use the Cα as a separate type of interaction center (in addition to the 20 amino acid centroids). It appears that the inclusion of the Cα is detrimental for threading methods, as it crudely monitors the local backbone fitness. Because threading methods derive their backbone conformations directly from native structures, the Cα energy terms only add noise to the signal ***(33)***.

The placement of a virtual centroid is also arbitary. For instance, one might take the mean projection of side-chain centroids in the database onto the Cα–Cβ vector ***(13)*** or the average atomic coordinate centers of all side-chains of a given type ***(14)***. However, the overall performance of a scoring function does not seem to be very sensitive to the placement of a single interaction center.

3.1.2. Distance-Dependent Energetics

Contact functions are step-functions; when residues are within an arbitrary cutoff distance an energy term is added to the total score. A single cutoff can be applied, as in the case of the Shell function described earlier. Alternatively, one could define different effective interaction distances depending on the pair of residues ***(30)***.

Any "on/off" contact approach may be considered as nonphysical because Coulombic and van der Waals interactions smoothly increase and decrease as a function of spatial distance. To address this issue, Park et al. ***(33)*** tested a series of functions with pairwise energetics identical to the contact functions, but with Lennard–Jones style functional forms ***(42)***:

$$E = \sum_{ij} (A/r_{ij}^8 - B_{ij}/r_{ij}^4) \quad (24)$$

where A_{ij} and B_{ij} are energy parameters dependent on the contact energy e_{ij} between residues i and j and the effective distance of interaction between i and j. However, the more complex distance-dependent functions did not perform any better than simple contact functions at discriminating near-native folds in the test set described earlier ***(33)***.

3.1.3. Multibody Interactions

Most statistical potentials are based on frequencies of pairwise interaction, but functions that include higher-order terms have been developed ***(25,43***A recent study on four-body interactions describes tendencies that cannot be captured by a pair potential, such as the preference for certain side-chain size combinations in the hydrophobic core ***(43)***. It would be interesting to test the performance of these potentials on the decoy sets described in this chapter.

3.1.4. Reference State

In the Park and Levitt ***(30)*** implementation of the solvent-exposed reference state (*see* **Subheading 2.2.2.**), all 210 residue pairwise energies are negative, which means that the formation of new protein–protein contacts is always preferred. Practically speaking, if one were to use a solvent-exposed reference state to fold an polypeptide chain from an extended conformation, a function such as the Contact(MJ) would favor compact conformations and drive chain collapse. However, the drawback of using the solvent-exposed reference state

in screening already-compact conformations is that the discrimination between the states is weak. Thus, the Shell function, which uses a generic compact shape as a reference, exhibits far better performance in the Park and Levitt *ab initio* test ***(30)***. On the other hand, the Shell function is less adept at recognizing a native fold from an semifolded, expanded decoy conformation generated by molecular dynamics at high temperature ***(33)***, suggesting that this function cannot be used in minimization methods without another term that monitors compactness.

3.2. Histogram Function

Park et al. ***(33)*** observed that the distance-dependent energies extracted by this function can lead to undesirable results in certain situations. Because the database of proteins used to compile the parameters includes proteins of all sizes, the most-favored interresidue distances for a given pair do not reflect those of the small proteins that serve as *ab initio* targets. This implies that if one tries to fold a small protein using only a PMF without an additional term to enforce compactness, then the most-favored structures will be more expanded than the native protein. For example, Simons et al. ***(21)*** used a scoring method related to the Histogram function to drive the folding of their small proteins, but also considered the radius of gyration as part of their final objective function.

3.3. Hydrophobic Fitness Function

This function, which does not require any parameters from the database, performed surprisingly well in most of our tests. However, because of its unusual functional form, is expected to be less amenable for minimization than screening discrete folds. Moreover, as it does not consider disulfide pairings, near-native fold recognition for small proteins that depend on disulfide bridges is noticeably worse than average ***(33)***.

3.4. All-Atom Scoring Function

All the interatomic distances in the conformation are calculated given a set of coordinates. The number of occurrences of atom pairs at particular distances are stored. This process is repeated for all the coordinate files in the database. Once the raw counts are collated, a table of negative log conditional probability scores for all the 167x167 possible pairs of atoms for the 18 distance ranges ***(22)*** is computed (*see* **Subheading 2.4.**).

The all-atom scoring function is susceptible to the problems that plague other knowledge-based functions because of the following issues: (1) the non-independence of pairwise interactions, (2) the lack of sufficient observations for accurate “pseudoenergies”, (3) an arbitrary reference state, and (4) an averag-

ing of environments. In practice, (2) is not a severe problem in this implemention, as the function does not use sequence separation, resulting in a greater number of observations in a given distance bin; (3) is chosen for the application at hand: to discriminate compact native conformations from nonnative ones; (1) and (4) require taking into account higher-order interactions, which, given the size of the current PDB (7) leads to sparse data. As a consequence, a compromise must be made between the number of parameters used and the size of the database. Based on our studies on various decoy sets (*see* **Subheading 3.5.**), we feel these compromises are justified.

3.5. Using Decoy Sets to Evaluate Scoring Functions

Decoys (nonnative or near-native conformations) are generally used to test whether a scoring function is useful. Although the utility of a function lies in its use in exhaustive or minimization methods, a scoring function has to at least do well in decoy-based tests before it can be considered for simulation. Use of decoys has its pitfalls, the primary one being that there may be artifacts in a particular decoy set that are picked up by a scoring function, resulting in accurate discrimination for that decoy set but not for others. For example, the misfolded decoys described in **Subheading 2.4.** are slightly expanded relative to the native structure. Thus a simple function that measures the amount of compactness does better than the C_β–C_β scoring function with a compact reference state. However, this simple function does not work as well as the C_β–C_β function for the 1r69 decoy set.

Thus an "ideal" function is one that discriminates well (100%) for a variety of decoy sets. Adding detail to the function appears to move us closer to this goal ***(22)***.

Acknowledgments

We thank Michael Levitt for his support. Part of this work was supported by NIH Grant GM45514.

References

1. Anfinsen, C. B. (1973) Principles that govern the folding of protein chains. *Science* **181,** 223–230.
2. Miyazawa, S. and Jernigan, R. L. (1985) Estimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation. *Macromolecules* **18,** 534–552
3. Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, E. F., Brice, M. D., Jr, Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1977) Protein Data Bank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.* **112,** 535–542.

4. Levitt, M., Hirshberg, M., Sharon, R., and Daggett, V. (1995) Potential energy function and parameters for simulations of the molecular dynamics of proteins and nucleic acids in solution. *Comp. Phys. Commun.* **91,** 215–231.
5. Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* **4,** 187–217.
6. Cornell, W. D., Cieplak, P., Bayly, C. I., Gould, I. R., Merz, K. M., Jr, Ferguson, D. M., Spellmeyer, D. C., Fox, T., Caldwell, J. W., and Kollman, P. A. (1995) A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc.* **117,** 5179–5197.
7. Jorgensen, W. and Tirado-Rives, J. (1988) The OPLS potential function for proteins. Energy minimizations for crystals of cyclic peptides and crambin. *J. Am. Chem. Soc.* **110,** 1657–1666.
8. Sippl, M. J. (1995) Knowledge-based potentials for proteins. *Curr. Opin. Struct. Biol.* **5,** 229–235.
9. Jernigan, R. L. and Bahar, I. (1996) Structure-derived potentials and protein simulations. *Curr. Opin. Struct. Biol.* **6,** 195–209.
10. Sippl, M. J. (1990) Calculation of conformational ensembles from potentials of mean force. An approach to the knowledge-based prediction of local structures in globular proteins. *J. Mol. Biol.* **213,** 859–883.
11. Bowie, J. U., Lüthy, R., and Eisenberg, D. (1991) A method to identify protein sequences that fold into a known three-dimensional structure. *Science* **253,** 164–170.
12. Jones, D. T., Taylor, W. R., and Thornton, J. M. (1992) A new approach to protein fold recognition. *Nature* **358,** 86–89.
13. Bryant, S. H. and Lawrence, C. E. (1993) An empirical energy function for threading protein sequence through folding motif. *Proteins: Struct. Funct. Genet.* **16,** 92–112.
14. Kocher, J.-P. A., Rooman, M. J., and Wodak, S. J. (1994) Factors influencing the ability of knowledge-based potentials to identify native sequence–structure matches. *J. Mol. Biol.* **235,** 1598–1613.
15. Godzik, A., Kolinski, A., and Skolnick, J. (1995) Are proteins ideal mixtures of amino acids? Analysis of energy parameter sets. *Protein Sci.* **4,** 2107–2117.
16. Godzik, A. (1996) Knowledge-based potentials for protein folding: what can we learn from known protein sequences? *Structure* **4,** 363–366.
17. Thomas, P. D. and Dill, K. A. (1996) Statistical potentials extracted from protein structures: how accurate are they? *J. Mol. Biol.* **257,** 457–469.
18. Ben Naim, A. (1997) Statistical potentials extracted from protein structures: are these meaningful potentials? J. Chem. Phys. **107,** 3698–3706.
19. Rooman, M. J. and Wodak, S. J. (1995) Are database-derived potentials valid for scoring both forward and inverted protein folding? *Protein Eng.* **8,** 849–858.
20. Skolnick, J., Jaroszewski, L., Kolinski, A., and Godzik, A. (1997) Derivation and testing of pair potentials for protein folding. When is the quasi-chemical approximation correct? *Protein Sci.* **6,** 676–688.
21. Simons, K. T., Kooperberg, C., Huang, E., and Baker, D. (1997) Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *J. Mol. Biol.* **268,** 209–225.

22. Samudrala, R. and Moult, J. (1997) An all-atom distance-dependent conditional probability discriminatory function for protein structure prediction. *J. Mol. Biol.* **275,** 893–914.
23. Wodak, S. J. and Rooman, M. J. (1993) Generating and testing protein folds. *Curr. Opin. Struct. Biol.* **3,** 247–259.
24. Moult, J. (1997) Comparison of database potentials and molecular mechanics force field. *Curr. Opin. Struct. Biol.* **7,** 194–199.
25. Godzik, A., Kolinski, A., and Skolnick, J. (1992) Topology fingerprint approach to the inverse protein folding problem. *J. Mol. Biol.* **227,** 227–238.
26. Rice, D. W. and Eisenberg, D. (1997) A 3D–1D substitution matrix for protein fold recognition that includes predicted secondary structure of the sequence. *J. Mol. Biol.* **267,** 1026–1038.
27. Russell, R. B., Copley, R. R., and Barton, G. J. (1996) Protein fold recognition by mapping predicted secondary structures. *J. Mol. Biol.* **259,** 349–365.
28. Di Francesco, V., Garnier, J., and Munson, P. J. (1997) Protein topology recognition from secondary structure sequences: application of the hidden Markov models to the alpha class proteins. *J. Mol. Biol.* **267,** 446–463.
29. Defay, T. R. and Cohen, F. E. (1996) Multiple sequence information for threading algorithms. *J. Mol. Biol.* **262,** 314–323.
30. Park, B. and Levitt, M. (1996) Energy functions that discriminate X-ray and near-native folds from well-constructed decoys. *J. Mol. Biol.* **258,** 267–392.
31. Hinds, D. A. and Levitt, M. (1992) A lattice model for protein structure prediction at low resolution. *Proc. Natl. Acad. Sci. USA* **89,** 2536–2540.
32. Park, B. and Levitt, M. (1995) The complexity and accuracy of discrete state models of protein structure. *J. Mol. Biol.* **249,** 493–507.
33. Park, B. H., Huang, E. S., and Levitt, M. (1997) Factors affecting the ability of energy functions to discriminate correct from incorrect folds. *J. Mol. Biol.* **266,** 831–846.
34. Huang, E. S., Subbiah, S., and Levitt, M. (1995) Recognizing native folds by the arrangement of hydrophobic and polar residues. *J. Mol. Biol.* **252,** 709–720.
35. Kolinski, A. and Skolnick, J. (1994) Monte Carlo simulations of protein folding. I. Lattice model and interaction scheme. *Proteins: Struct. Funct. Genet.* **18,** 338–352.
36. Kolinski, A., and Skolnick, J. (1994) Monte Carlo simulations of protein folding. II. Application to Protein A, ROP, and crambin. *Proteins: Struct. Funct. Genet.* **18,** 353–366.
37. Sun, S., Thomas, P. D., and Dill, K. A. (1995) A simple protein folding algorithm using a binary code and secondary structure constraints. *Protein Eng.* **8,** 769–778.
38. Holland, J. H. (1975) *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, Ann Arbor, MI.
39. Holm, L. and Sander, C. (1992) Evaluation of protein models by atomic solvation preference. *J. Mol. Biol.* **225,** 93–105.
40. Levitt, M. (1992) Accurate modeling of protein conformation by automatic segment matching. *J. Mol. Biol.* **226,** 507–533.

41. Holm, L. and Sander, C. (1991) Database algorithm for generating protein backbone and side-chain coordinates from a Cα trace: application to model building and detection of coordinate errors. *J. Mol. Biol.* **218,** 183–194.
42. Wallqvist, A. and Ullner, M. (1994) A simplified amino acid potential for use in structure prediction of proteins. *Proteins: Struct. Funct. Genet.* **18,** 267–280.
43. Munson, P. J. and Singh, R. K. (1997) Statistical significance of hierarchical multibody potentials based on Delaunay tessellation and their application in sequence–structure alignment. *Protein Sci.* **6,** 1467–1481.

11

Ab Initio Loop Modeling and Its Application to Homology Modeling

Robert E. Bruccoleri

1. Introduction

The modeling of loops remains a challenging theoretical and practical problem in the prediction of protein structure ***(1)***. There are several general methods for modeling such loops including the use of databases ***(2–5)***, simulation ***(6–8)***, and *ab initio* methods ***(9–10)*** as well as other methods described in this volume. In this chapter, an *ab initio* method is described that uses conformational search to thoroughly explore the possible conformations of a loop, and that uses an energy function to rank this conformations for the prediction of the loop. In addition, a detailed protocol for homology modeling using the program, CONGEN, will be presented along using an example taken from the recent Comparative Assessment of Structural Prediction 2 (CASP2).

2. Materials

The materials required for homology modeling are the CONGEN molecular modeling program and a fast computer to run it on. CONGEN is a program for modeling loops using conformational search ***(10)***. In addition, the program has a large set of molecular modeling commands that are needed to support the process of loop construction. The program can be obtained over the World Wide Web by going to the URL, `http://www.congen.com/`, or by contacting the author via E-mail, `bruc@acm.org`.

The program currently runs on most UNIX computers, and runs best on a Silicon Graphics machine. For most loop-modeling efforts, a machine with 64 MB of RAM, and 1 GB of disk storage should suffice, although more RAM may be helpful with larger problems. The program can perform calculations in parallel, so multiprocessor SGI installations can be effectively used.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

2.1. Operation of CONGEN

The fundamental problem of generating loop conformations is finding a set of atomic coordinates for the backbone and side chains that satisfy all its stereochemical and steric constraints. For the sake of efficiency, it is presumed that bond lengths and bond angles are fixed* and in addition, it is assumed that the peptide ω torsion angle is also planar. Under these assumptions the only degrees of freedom in the loop are the torsion angles. Given the chemical structure of proteins, the search process is divided into backbone and side-chain constructions. The backbone conformational space is normally sampled before the side chains because the chain-closure condition is very restrictive. As a result, fewer samples are generated early in the process, which helps to reduce the necessary computer time.

2.1.1. Backbone Construction

The generation of backbone coordinates depends heavily on the modified G ō and Scheraga chain closure algorithm ***(12,13)***. The algorithm is designed to calculate local deformations of a polymer chain, i.e., finding all possible arrangements of a polymer anchored at two fixed endpoints. Given stereochemical parameters for the construction of the polymer, and six adjustable torsion angles between the two fixed points, this algorithm calculates values for the six torsion angles in order to perfectly connect the polymer from one endpoint to the other. In the sampling of the backbone, the use of a planar ω torsion angle reduces the number of free backbone torsion angles per residue to two, and therefore, three residues are required for the application of the G ō and Scheraga algorithm. For generating conformations of loops with more than three residues, the backbone torsion angles of all but three residues are sampled, and the G ō and Scheraga procedure is used to close the backbone.

The free sampling of backbone torsion angles is done with the aid of a backbone energy map. Bruccoleri and Karplus (1987) calculated the energetics of constructing the backbone for three different classes of amino acids: glycine, proline, and all the rest ***(13a)***. This information is stored as a map ***(14)*** which gives the energy as a function of discrete values ϕ, υ, and ω, where ω can only be 0° (*cis*) or 180° (*trans*). A set of maps corresponding to grids of 60°, 30°, 15°, 10°, and 5° have been calculated; typically, a 30° sampling is sufficiently fine for good agreement.

With regard to the peptide ω angle, only the proline ω angle is normally allowed to sample *cis* values. However, CONGEN can be directed to sample *cis* omega angles for all amino acids.

*The chain-closure algorithm can perturb the bond angles in the peptide backbone a small amount.

The ring in proline creates special problems. The proline ring constrains the phi torsion to be close to –65°; any deviation from –65° distorts the ring. The minimum energy configuration of the proline ring (specifically, 1,2 dimethyl pyrrolidine) has been determined for a range of ϕ angles (±90°) about –65° using energy minimization with a constraint on ϕ, and a file has been constructed that contains these energies and the construction parameters necessary to calculate the position of C_β, C_γ, and C_δ of the proline. All of these energies are adjusted relative to a minimum ring energy equal to zero. After a chain closure is performed, any conformations that have a proline ϕ angle whose energy exceeds the minimum energy by more than the parameter, `ERINGPRO`, are discarded. Generally, a large value for `ERINGPRO` is used (50 kcal/mole) so that the chain-closure algorithm does not overly restrict proline closures. The *cis–trans* peptide isomerization is handled by trying all possible combinations of *cis* and *trans* configurations. The user has complete control over which residues can be built in the *cis* isomer. As there are only three residues involved in the chain closure, this results in no more than eight (2^3) attempts at chain closure.

There are two optimizations performed during the sampling of backbone torsions. First, whenever any atom is constructed, a check is made to see if the atom overlaps with the van der Waals radius of any other atom in the system. If so, that conformation is discarded. The option, `MAXEVDW`, is used to specify the maximum allowed van der Waals energy of such a contact. It must be set to a value of at least 5 kcal/mole. Second, as backbone residues are generated, CONGEN calculates the distance from the growing end back to the other fixed point. If that distance is greater than can be reached by fully extended backbone, then those conformations are discarded. The option, `CLSA`, in the backbone degree of freedom is used to specify the other endpoint of the loop which is used for this optimization.

The backbone can be constructed either forward from the N-terminus or backward from the C-terminus order until only three residues remain. The N-terminus of the internal segment is anchored on the peptide nitrogen; the C-terminus is anchored on C_α. When the construction direction is from the N-terminus to the C terminus, the first torsion to be sampled in a residue is the ω angle (which normally is sampled just at 180°, and can be sampled at 0° for prolines or, as an option, all the amino acids). It determines the C_α and the peptide hydrogen positions. The ϕ angle determines the position of the carbonyl carbon and the beta carbon of the side chain; and finally, the φ angle determines the carbonyl oxygen and peptide nitrogen of the next residue. When the construction is in the reverse direction; the φ angle determines the peptide nitrogen; the ϕ angle determines the carbonyl carbon of the preceding residue, the peptide hydrogen, and the beta carbon; and the ω angle determines the position of the preceding residue's C_α and carbonyl oxygen.

2.1.2. Sidechain Construction

Given a set of backbone conformations, it remains to generate a set of side-chain atom positions for each of the backbone conformations. This problem is divided into two parts — construction of individual side chains and combining results from individual side chains for all the residues.

As with the backbone atom placement, the atoms of a side chain are positioned based on free torsion angles. The side-chain torsions are processed from the backbone out as each succeeding atom requires the position of the previous atom for its placement. The sampling interval of each torsion can be either some fixed number of degrees or the period of the torsion energy. If the latter is used, and the parameters for the torsion energy specify only a single term in the Fourier series for the torsion energy, then the side-chain torsion energy is always zero.

It is common for one free torsion to generate the position of more than one atom because of side-chain branching, nonrotatable bonds, and rings. For example, although an explicit hydrogen ***(15)*** tryptophan has 11 side-chain atoms to be placed, it has only two free torsion angles. Also, some side-chain branching is symmetric, e.g., phenylalanine, and CONGEN can use such symmetry to reduce the sampling necessary.

As with the backbone construction, a search of the surrounding space is made for any constructed atom to see if there are any close contacts. However, with the side chains, there are two ways of checking for such overlaps. The first method is very simple: given the sampling of the torsion angles, each atom is constructed and checked for contacts.

The second method — van der Waals avoidance — is more time consuming, but it yields better quality structures. It is a straightforward geometrical problem to determine the range of torsion angles that will avoid constructing an atom within a given distance of other atoms in the system. As a side-chain torsion angle, χ_i, varies, it specifies a circular locus of points on which atoms can be constructed. If atoms in the vicinity of this circle are examined, the sectors of the circle that will result in the repulsive overlap of the constructed atom with its spatial neighbors can be calculated. The complement of these sectors can be used to determine values for the χ_i angles that avoid bad contacts.

The information needed for side-chain construction is stored in a side-chain topology file. It is a straightforward matter to add new amino acids to this file so that the structure of unnatural amino acids can be predicted.

Given this method for constructing individual side chains, it remains to combine side-chain conformations for all the side chains attached to a particular backbone conformer.

Because the backbone construction process provides the position of C_β, there is a strong bias to the side-chain orientation. Thus, an acceptable course of action is the generation of only one side-chain conformation for each backbone conformation. A substantial effort must be made to ensure that this one conformation the lowest energy possible for the given backbone. Second, because the side chains close together in sequence frequently are not close together in space, and therefore do not interact strongly, it is a reasonable approximation to treat the side chains quasiindependently. Instead of finding all combinations of side-chain atomic positions, the side chains can be processed sequentially so that the time required for side chain placement increases linearly, rather than exponentially, with the number of residues.

This is the basis of the Iterative side-chain construction option. It begins with an energetically acceptable side-chain conformation for all the side chains. This conformation is generated, if possible, using the First method described below. Starting with this conformation, all the possible positions for the side chain atoms of the first residue are recalculated, and the conformation with the lowest energy is selected. The value of the evaluation function is also saved. This regeneration is done with all the other side-chain atoms present so that their effect can be accounted for. The process is repeated sequentially for the rest of the side chains in the gap. The process then returns to the first residue and it is repeated over each side chain until the energies of the side-chain atoms do not change or until the number of passes reaches an iteration limit. This method has the virtue that only one conformation is generated per backbone conformation, and it is an energetically reasonable one. However, if there are significant interactions between the side-chain atoms, the initial state of the side chains will bias the iterative process, and the lowest energy side-chain conformation may be missed.

Other options exist for constructing side chains, and these are described in the manual. The most relevant for homology construction are the First method and the Independent method. The First method attempts to find one way of placing the side chains by performing a series of nested iterations over every side chain until all atoms are placed with no individual van der Waals contacts exceeding `MAXEVDW` in energy. The Independent method performs an exhaustive conformational search for each side chain, but the atoms of the other side chains in the peptide are ignored; interactions with all other atoms in the system are included. This method is only used when a single side chain is being constructed.

With any of the methods described, the CONGEN command can apply any of the minimization algorithms to the generated conformations. Minimization provides an ability to reduce the small van der Waals repulsions that are inevitable with coarse torsion grids.

2.1.3. Degree-of-Freedom Operators

Within CONGEN, the specification of a degree of freedom signifies a computer operation applied to a group (zero or more) of atoms by either sampling a set of variables or performing an operation on existing atoms. When a conformational search is specified, the user indicates which degrees of freedom are to be sampled and also their order. The program automatically sets up a nested iteration over all of them. Only successful samples of a degree of freedom will invoke the succeeding degrees of freedom.

There are two reasons for taking this abstract approach to the operation of the search. First, it allows searches of arbitrary complexity to be performed. Second, the operations inherent in sampling a degree of freedom can be separated from the process of managing the search. Such modularity greatly simplifies the implementation of the program. In addition, one can apply the methods of state space search as developed in research into artificial intelligence ***(16)***.

Currently, six degree-of-freedom operators are provided in CONGEN. Three of them deal with atomic construction using stereochemistry. The backbone degree of freedom generates the position of the peptide backbone atoms, and the chain-closure degree of freedom closes a loop. Because the G ō and Scheraga procedure ***(12)*** finds multiple solutions to the chain closure, each solution is treated as a separate sample. The side-chain degree of freedom will construct side chains onto any number of backbone residues and, depending on the method, it will generate either single samples or multiple ones.

Two degrees of freedom are involved with input and output. The "Write" degree of freedom writes a conformation to a file each time it is invoked. It can also do some limited filtering of what is written by comparing the energy of each conformer against the minimum energy seen thus far. Normally, this filter will greatly reduce the number of conformers written to a file. In all cases, this degree of freedom does not generate any atomic positions, and it always succeeds. The "Read" degree of freedom can be viewed as an inverse of "Write." It reads a set of conformations from a file, and then invokes succeeding degrees of freedom on each one. Conformations can be selected based on their energies, so it is possible to set up a "buildup" procedure ***(17)*** where the best conformations from one search are used as the starting point for adding additional residues. In addition, this degree of freedom allows a user to input his or her own set of conformations, which can be generated by arbitrary means. This approach was used by Martin et al. to process loop conformations as found in a database ***(3)***.

The final degree of freedom is the "Evaluate" degree of freedom. This operation is responsible for calculating either energies or root-mean-square (RMS) deviations. When used for energy evaluations, this degree of freedom

can either calculate the energy, or it can perform minimization or dynamics on each of the conformers. When used for RMS deviations, it compares the coordinates of the conformations against a reference coordinate set. This is used for testing the search process; in particular, to see if a search is capable of generating the original experimental coordinates.

3. Methods

Starting with the sequence of protein for which a structure is desired (referred to as the target sequence), and the knowledge that there are other homologous structures already solved, the protocol for homology modeling consists of seven basic steps. These steps will be illustrated using command files and results from the construction of the phosphotransferase enzyme IIA domain ***(18)***, which was target 3 in the Comparative Modeling section of CASP2. Except where otherwise noted, all the command file excerpts assume that the data structures needed for a CONGEN run have already been created. In addition, the full input files for CONGEN are included as the `casp2` subdirectory of the documentation directory for CONGEN.

3.1. Parent Structure Determination

Using a sequence analysis package such as the Genetics Computer Group programs ***(19)***, and a database of protein sequences taken from the solved structures in the Brookhaven Protein Data Bank (PDB), use an gapped alignment tool, such as FASTA, to find the best sequence matches to the target sequence.

The reliability of the homology model can be accurately estimated from the sequence similarity found ***(5,20)***. High-sequence identities, on the order of >70%, indicate that the homology models will be of high accuracy, with RMS deviations from the native structure on the order of 1 or 2 Å. Low-sequence identities, on the order of <30%, are likely to result in poor-quality models with RMS deviations greater than 4 Å. Intermediate sequence identities will have errors within these values. You must judge whether the information requirements you have for your structures can tolerate the expected errors before proceeding.

3.2. Alignment

The most critical step in the success of a homology modeling effort is the alignment of the target sequence to the structure you are building the target from ***(5)***. The method used in my efforts in the CASP2 competition used the `COMPARE` command in CONGEN to prevent insertions and deletions in regions of secondary-structure in the parent structure. To use this method, it is best to print a table of ϕ, φ, and ω angles and to prepare a labeled C_α stereo plot of the parent structure. The table can be prepared using analysis facility in CONGEN as shown in the input fragment in **Fig. 1** and the labeled stereo plots can be

```
ANAL
BUILD TORSION GEOMETRY
DELETE TAGS EXCEPT ALLSEG ALLRES C-N-CA-C N-CA-C-N CA-C-N-CA $
PRINT TABLE PRETTY
END
```

Fig. 1. CONGEN input to generate a torsion-angle table. The analysis facility is used to construct a torsion angle table, which is edited to remove all torsion angles except for ϕ, φ, and ω.

```
COOR ORIE                                ! Center the molecule on the origin
ANAL                                     ! Enter analysis facilitiy
OPEN UNIT 40 NAME 1F3G.PLT FORM WRITE    ! Create plot file for plt2
BUILD ATOM X                             ! Generate a dummy table of atoms
DELETE TAGS EXCEPT ALLSEG ALLRES CA      ! Delete everything except alpha carbons
DRAW PLT2 UNIT 40 CONNECT TABLE -        ! Draw the alpha carbon plot.
     LABEL FREQ 10 IUPAC CA $
end
```

Fig. 2. CONGEN input for generating an C_α plot.

```
dev postl 1f3g.ps
font duplex
font inline
sa 10.0
ori 8. 10.
noecho
str 1f3g.plt
echo
ryl -6
ori 18. 10.
noecho
str 1f3g.plt
echo
cta 13.0 1.0 "1f3g"
```

Fig. 3. PLT2 input for generating a C_α plot in stereo. These commands make two drawings at different positions on the paper, with the second drawing rotated 6° around the *y*-axis.

generated using the input fragment in **Fig. 2** and displayed using PLT2 using the input in **Fig. 3**.

The regions of secondary-structure can be identified by either display only, but it is valuable to compare the visual and tabular outputs to identify irregularities in the structure. Alpha helical regions are identified by ϕ, φ values near –57° and –47°, and beta sheet regions are identified by values around –130° and 130°. Once the regions of secondary-structure are identified, they are

```
Construction of 1f3g - phosphocarrier III(GLC) from E. Coli.
*
DEBUG CORE 0
OPEN UNIT 1 NAME CGDATA:AMBER94RTF.MOD UNFORM READ
READ RTF FILE UNIT 1
OPEN UNIT 1 NAME CGDATA:AMBER94PARM.MOD UNFORM READ
READ PARAM FILE UNIT 1
READ SEQUENCE CARD A94P ABBREV AA            ! Generate a PSF for the target
     Phosphotransferase enzyme IIA domain
*
148
N L K V L A P C D G T I I T L D E V E D E V F K E R M L G
D G F A I N P K S N D F H A P V S G K L V T A F P T K H A F G I Q T K
S G V E I L L H I G L D T V S L D G N G F E S F V T Q D Q E V N A G D
K L V T V D L K S V A K K V P S I K S P I I F T N N G G K T L E I V K
M G E V K Q G D V V A I L K
GENERATE A
OPEN UNIT 3 NAME TEMP.PSF UNFORM WRITE       ! Write out target PSF
WRITE PSF FILE UNIT 3
phosphotransferase enzyme IIA domain
*
CLOSE UNIT 3
OPEN UNIT 4 NAME TEMP.MOD UNFORM WRITE       ! Write out dummy coords
WRITE COOR FILE UNIT 4                       ! for use by analysis.
Blank coordinates
*
CLOSE UNIT 4
OPEN UNIT 11 NAME 1F3G.PSF UNFORM READ       ! Parent structure must
OPEN UNIT 12 NAME 1F3G.MOD UNFORM READ       ! be loaded for PROTECT
READ PSF FILE UNIT 11                        ! option to work.
READ COOR FILE UNIT 12
OPEN UNIT 3 NAME TEMP.PSF UNFORM READ
OPEN UNIT 4 NAME TEMP.MOD UNFORM READ
ANAL
COMPARE PSF 3 COOR UNIT 4 $ -                ! Match against target
        RESMATCH ALLSEG PRINT HOMOLOGY -     ! Specify conserved AA
                 CONSERVE ALA MET VAL ILE LEU $ -
                 CONSERVE THR SER $ -
                 CONSERVE GLU ASP $ -
                 CONSERVE GLN ASN $ -
                 CONSERVE PHE TRP $ -
                 CONSERVE LYS ARG $ -        ! Below we avoid INDEL's
                 PROTECT   1   6    8  14   21  24   32  36 -
                          40  44   46  53   58  63   67  73 -
                          85  89   93  97  102 105  106 112 -
                          117 122  130 135  137 140 $$
END
```

Fig. 4. CONGEN input to compare parent sequence to target sequence.

included as values in the PROTECT option of a COMPARE command in the analysis facility of CONGEN, and the sequence alignment generated by the program will avoid insertions and deletions in these regions.

In addition, it is valuable to include sets of conserved amino acids into the RESMATCH option of the COMPARE command, so that the homology will be computed more accurately. Typically, the amino acids are grouped into aliphatic, aromatic, hydroxyl, acid, amide, and base categories.

A sample input is given in **Fig. 4**.

3.3. Splicing

Once the alignment is determined, the next step is the transformation of the protein from the parent structure to the target structure. This is done using the SPLICE command. The COMPARE command described in **step 2** above will generate a set of SPLICE commands that you can cut and paste into the command file for splicing. The splicing command will change the structure of the protein in the computer. It does this by changing the sequence of the protein, rebuilding the protein structure file, and then shuffling coordinates to match atom names. When side chains are changed, CONGEN will initialize the atom coordinates for those side chains, and you will have to rebuild them as described in the following step.

The splicing operation is also the time to prepare the plan for modeling all the changes in sequence. There are three possible choices: use of the parent coordinates, side-chain-only reconstruction, and loop construction. The first choice is applicable only when the amino acid is same between the parent and target sequence, but there are circumstances when modeling should be done even if there is no change in sequence. Side-chain only reconstruction typically applies in regions where changes in secondary-structure are unlikely. A full-loop reconstruction is done wherever backbone conformational change is expected.

To decide among these choices, you examine each change along the sequence. All insertions and deletions require a loop reconstruction. Loops must be at least four residues long. The endpoints of the loop should correspond to the ends of secondary-structure or highly conserved sequence, but if the length of a loop is greater than 10 residues, then you must either consider using shorter loops or use the directed search methods as described in **Subheading 3.5**. If you have multiple structures that you can use as parents, you can examine them all to see if the structures of the end of loop are conserved, and therefore, you can move the endpoint of the loop to the nearest, structurally conserved residue.

In regions of the alignment that do not have insertions or deletions, loop reconstructions may be necessary if there are changes in amino acid sequence involving glycine or proline. Glycine is much more flexible than the other amino acids, and proline is much less flexible than the others, specifically because its ϕ angle is restricted to approx –60°. Thus the following rules apply in general, but always keep in mind that glycine and proline serve important structural purposes, and many changes of sequence involving these amino acids indicates a change in structure.

Gly → other — If ϕ and φ of the glycine are within permitted values for the new amino acid, then one can leave the backbone alone. Otherwise, the surrounding region must be rebuilt.

```
! To initialize the sidechain coordinates for residue SEGID RESID
! which is assumed not to be proline or glycine.

COOR INIT CLEAR ATOM <SEGID> <RESID> * ENTER -
          CLEAR ATOM * * N ATOM * * H ATOM * * CA ATOM * * HA -
                ATOM * * CB ATOM * * C ATOM * * O EXCL

! To initialize the atoms in a loop in segment, SEGID, from
! residue, RESID1 through RESID2

COOR INIT CLEAR RANGE <SEGID> <RESID1> H <SEGID> <RESID2> C ENTER -
          CLEAR ATOM <SEGID> <RESID2> CA ATOM <SEGID> <RESID2> C EXCL
```

Fig. 5. Coordinate initialization commands.

Pro → other	If the proline is found in a conserved region of secondary-structure, then only the side chain of the new amino acid should be modeled. Otherwise, the surrounding region must be rebuilt.
Other → gly	If the proline is not in a conserved region of secondary-structure, rebuild surrounding region.
Other → pro	If the ϕ angle of the parent amino acid is close to –60° and if the proline is in a conserved region, then minimized the proline ring into place. Otherwise, rebuild as a loop.

If you have multiple structures homologous to your parent structure, it is important to examine all the differences in structure, and, in general, it is best to rebuild any part of your target where the parent molecules have any variability in structure. There are examples where the same sequence folds into completely different structures ***(21)***. Since we operate on the presumption that sequence homology implies structural homology, only variation of multiple parent structures can indicate when this presumption fails.

Finally, you should initialize the coordinates of all the atoms you will be rebuilding during the construction process. Although CONGEN will initialize the coordinates for the atoms being constructed in a single conformational search, it cannot do this for all the searches you perform. Any atoms in a loop whose length is changed can confound the search. For side-chain-only searches, you must delete every atom in the side chain except C_β. For loops, you must initialize all the atoms in the loop except for the peptide nitrogen of the amino terminal residue, and the C_α and carbonyl carbon and oxygen of the carboxy terminal end. **Figure 5** illustrates the necessary commands.

3.4. Reconstruction of Changed Side Chains

With the exception of proline, the reconstruction of changed side chains is performed using a single side-chain conformational search. Prolines require

```
OPEN UNIT 66 NAME sidechains.STS FORM WRITE           ! Status file
OPEN UNIT 60 NAME sidechains.CG UNFORM WRITE          ! Conformation output file
OPEN UNIT 70 NAME CGDATA:AMBER94TOPCG.INP FORM READ   ! Sidechain construction
                                                      ! rules file
OPEN UNIT 51 NAME CGDATA:AM94_EMAPGLY30.OMP FORM READ ! Glycine backbone energy map
OPEN UNIT 52 NAME CGDATA:AM94_EMAPALA30.OMP FORM READ ! Alanine backbone energy map
OPEN UNIT 53 NAME CGDATA:AM94_EMAPPRO30.OMP FORM READ ! Proline backbone energy map
OPEN UNIT 55 NAME CGDATA:AM94_PRO.CNS FORM READ       ! Proline constructor file

CGEN -                                          ! Beginning of multiline search command
STATUS UNIT 66 END -                            ! Specify status output unit
CHECKPOINT UNIT -1 TIME 60 NODE 1000000 END -   ! Specify timing of status
SEARCH DEPTH END -                              ! Exhaustive search
NBCG CUTNB 12.0 CTONNB 98.0 CTOFNB 99.0 ATOM END - ! Set non-bonded energy cutoffs
HBCG CUTHB 0.5 CTONHB 98.0 CTOFHB 99.0 -        ! These are dummy hydrogen bond
     CUTHBA 90.0 CTONHA 90.0 CTOFHA 90.0 END -  ! cutoffs required by AMBER 94
-                                               ! Specify sidechains for
-                                               ! reconstruction
SIDE MAXEVDW 20 SIDEOPT ITER VAVOID SGRID SELECT 30 30 60 60 MIN END EVAL E -
     STARTRES A   1 LASTRES A   5   STARTRES A   8 LASTRES A   9 -
     STARTRES A  11                 STARTRES A  13 LASTRES A  17 -
     STARTRES A  19                 STARTRES A  21 -
     STARTRES A  24                 STARTRES A  26 LASTRES A  28 -
     STARTRES A  32                 STARTRES A  35 -
     STARTRES A  37 LASTRES A  38   STARTRES A  40 LASTRES A  42 -
     STARTRES A  46                 STARTRES A  48 LASTRES A  52 -
     STARTRES A  56                 STARTRES A  59 -
     STARTRES A  62 LASTRES A  64   STARTRES A  69 LASTRES A  71 -
     STARTRES A  73                 STARTRES A  75 -
     STARTRES A  79                 STARTRES A  81 -
     STARTRES A  83                 STARTRES A  85 -
     STARTRES A  86 LASTRES A  92   STARTRES A  94 -
     STARTRES A  96 LASTRES A  97   STARTRES A 100 LASTRES A 104 -
     STARTRES A 107 LASTRES A 111   STARTRES A 113 -
     STARTRES A 116 LASTRES A 118   STARTRES A 120 LASTRES A 123 -
     STARTRES A 132 LASTRES A 133   STARTRES A 144 LASTRES A 147 $ -
EVL MINI ENERGY END $ -                         ! Energy evaluation
WRITE CUNIT 60 $ -                              ! Write results
ERINGPRO 50 GLYMAP 51 ALAMAP 52 PROMAP 53 PROCONS 55 - ! Specify map units
GLYEMAX 5 ALAEMAX 5 PROEMAX 5 -                 ! Specify backbone map cutoffs
STUNIT 70                                       ! Sidechain topology file unit
Construction of phosphotransferase IIA domain.
from Phosphocarrier protein III(glc)
*
```

Fig. 6. Side chain construction input file.

minimization, and a somewhat cumbersome procedure, which is described at the end of this section.

The main issues with the side chain conformational search are the selection of the maximum-allowed close contact energy (MAXEVDW), and the possible inclusion of other side chains in the structure into the conformational search. **Figure 6** gives the prototypical conformational search command.

For the initial attempt to reconstruct the side chains, you should use of a value of 10 for MAXEVDW, and try the search. If it succeeds, you are done. Otherwise, set the value to "1.0E20," run the program again, and examine the table

```
COOR COPY COMP                                        ! Copy all coordinates
IC SETUP                                              ! Build all missing coordinates
IC BILD
CONS FIX CLEAR ATOM <segid> <resid> * not             ! Freeze all coordinates
                                                      ! except the proline being
                                                      ! minimized.
MINI ABNR NSTEP 250 CUTNB 12.0 CTOFNB 98.0 CTONNB 99.0
                                                      ! Minimize proline
CONS FIX CLEAR                                        ! Unfreeze everything
COOR COPY CLEAR ATOM <segid> <resid> * NOT            ! Merge all other coordinates
                                                      ! with minimized proline.
```

Fig. 7. Proline ring minimization.

resulting from the close contact search. Find all the residues that have side chains with high-energy van der Waals contacts to the side chains you are building, and add these side chains to the ones you have selected for reconstruction. Repeat the search with a value of 10 for MAXEVDW, and see if it completes. If not, check the list again, and see if other side chains have to be added.

If the close contacts arise with backbone atoms, then you have four choices. (1) You can either treat the residues around the backbone atoms as a loop, and thereby rebuild them completely; (2) you can treat the backbone around changed side chains as a loop; (3) you can treat both backbones as loops; or (4) you can raise MAXEVDW specifically for the side chains involved. The choice depends on whether the backbone structures should be conserved or not. However, if a changed side chain results in bad contacts with nearby backbone atoms, it strongly suggests that there is going to be change of structure.

The reconstruction of proline residues requires some effort (*see* **Note 1**). You must first make a copy of the coordinates to the comparison set, use the internal coordinate construction commands to rebuild all the missing atoms in the system, copy the proline atoms to the comparison set, swap the comparison and main set, and then minimize the proline atoms. The command sequence is illustrated in **Fig. 7**.

3.5. Construction of Loops

Before the loops can be constructed, it is necessary to visualize the location of the loops on the structure, and determine a construction order. If a specific loop is not in contact with any other loop, then it can be constructed in any order. When there are loops than can interact, it is best to start with the loops that are shortest (as they are generally predicted more accurately) or to start with loops that have the most known structure around them (*see* **Note 2**). If two loops are intimately interacting, they can constructed together, but the CPU time requirements can be substantial. The directed conformational search meth-

```
OPEN UNIT 66 NAME L1.STS FORM WRITE                    ! Status file
OPEN UNIT 60 NAME L1.CG UNFORM WRITE                   ! Conformations output file
OPEN UNIT 70 NAME CGDATA:AMBER94TOPCG.INP FORM READ    ! Sidechain construction
                                                       ! rules file
OPEN UNIT 51 NAME CGDATA:AM94_EMAPGLY30.OMP FORM READ ! Glycine backbone energy map
OPEN UNIT 52 NAME CGDATA:AM94_EMAPALA30.OMP FORM READ ! Alanine backbone energy map
OPEN UNIT 53 NAME CGDATA:AM94_EMAPPRO30.OMP FORM READ ! Proline backbone energy map
OPEN UNIT 55 NAME CGDATA:AM94_PRO.CNS FORM READ        ! Proline constructor file

CGEN -                                                 ! Beginning of multiline
                                                       ! search command
STATUS UNIT 66 END -                                   ! Specify status output unit
CHECKPOINT UNIT -1 TIME 60 NODE 1000000 END -          ! Specifies timing of status
-
TREE LIMIT 500000 PRTFRQ 500000 END -                  ! Limits on the search tree size
SEARCH EVAL ENERGY END -                               ! Directed search specification
NBCG CUTNB 12.0 CTONNB 98.0 CTOFNB 99.0 ATOM END -     ! Non-bonded energy calculation
HBCG CUTHB 0.5 CTONHB 98.0 CTOFHB 99.0 -               ! Amber has no hydrogen bond
     CUTHBA 90.0 CTONHA 90.0 CTOFHA 90.0 END -         ! energy, so this turns it off.
BACK CISTRANS STARTRES A 125 MAXEVDW 20 -              ! Backbone A 125
     CLSA A 130 CA GRID 30 $ -
BACK CISTRANS REVERSE STARTRES A 130 MAXEVDW 20 -      ! Backbone A 130
     CLSA A 126 N GRID 30 $ -
BACK CISTRANS STARTRES A 126 MAXEVDW 20 -              ! Backbone A 126
     CLSA A 129 CA GRID 30 $ -
CHAIN CISTRANS STARTRES A 127 MAXEVDW 20 $ -           ! Chain closure A 127-129
SIDE STARTRES A 125 LASTRES A 130 MAXEVDW 20 -         ! Sidechains for A 125-130
     SIDEOPT ITER VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
EVL MINI ENERGY END $ -                                ! Energy evaluation
WRITE CUNIT 60 MINCUT 3.0 $ -                          ! Write results
STUNIT 70 GLYMAP 51 ALAMAP 52 PROMAP 53 PROCONS 55 -   ! Maps and proline constructors
GLYEMAX 5 ALAEMAX 5 PROEMAX 5 ERINGPRO 50              ! Energy limits.
FIRST LOOP IN PHOSPHOTRANSFERASE IIA DOMAIN.           ! Conformation file title.
*
```

Fig. 8. Loop construction input.

odology ***(22)*** can be used to shorten the search time for long loops, but at the risk of missing the lowest energy conformation.

The easiest way to prepare an input file for the construction of one loop is to take an existing input file, and edit it. The aforementioned splicing step will specify the residues at the ends of the loops, and the remaining issues are the order of construction, and the inclusion of additional side chains in addition to those in the loop. Typically, the order of construction is selected so that backbone residues on each side of the loop are constructed from the ends towards the middle. If one end of a loop is "deeper" in the protein than the other, then the section that is deeper is constructed first, and then each is constructed towards the middle. **Figure 8** illustrates a sample input file for the construction of residues 125–130 in segment A of the phosphotransferase IIA domain.

If a loop is so long that an exhaustive search is not feasible, then it may be necessary to use the directed conformational search ***(22)***. In this method, CONGEN explores the search tree ordered by the energy of the partial confor-

mations. This method generates low-energy conformations early in the search, but is not guaranteed to generated the lowest energy structure. There are two possible directed search options to try, `EVAL` and `MIX`. The `EVAL` option used the best first method, and the `MIX` method uses the mixed strategy method. For shorter loops, the `EVAL` option is usually better whereas for longer loops, the `MIX` can give better results. You can monitor the results by using the `graph_ebytime.perl` script, which shows the plot of the energy of generated conformations against output order. If the curve trends upward, then the directed search is working directly. If not, then you will have to settle for low-energy loop conformations instead the lowest possible.

If CONGEN fails to find any conformations for a loop or if the energy of the conformations is high (*see* **Note 3**), you must investigate the reason, correct it, and rebuild the loop. The first thing to do is to look at the loop region in stereo using molecular graphics and see if you can identify the problem. The loop endpoints may be too far apart, or they may be obstructed by other parts of the molecule. If they are too far apart, then you must add additional residues to the loop definition so that the new endpoints can be bridged with the additional residues. If there is an obstruction, then it must either cleared or the van der Waals cutoff must be raised in order to allow the new loop to be constructed through the obstruction.

If a visual examination does not reveal why the loop cannot be constructed, there are a number of tests that can be made using CONGEN. A quick test for an inadequate number of residues is to set `MAXEVDW` to `1.0E20`, add `MAXCONF 1` to the `WRITE` degree of freedom, and see if any conformations are generated. If not, then the distance or geometry of the endpoints precludes any loop construction. If increasing the van der Waals cutoffs results in loop conformations, then it is useful to make a series of runs using successively larger values of `MAXEVDW` starting at 10 kcal/mole, and see which value results in conformations. Then, the `SEARCH` command in the analysis facility can be used to identify the contacting residues. If the contacts are to other side chains, then these can be added to the side-chain degree of freedom. If the contacts involve backbone positions, it may be necessary to review the alignment, and see if another loop should defined.

One should also be careful that coordinate initialization errors have not occurred. Errors in initialization can manifest themselves with high bond, bond angle, or van der Waals energies. Such errors require correction in the splicing step, and rerunning all the steps starting from that point.

3.6. Construction of Termini

At this point, the only construction remaining is the terminal residues at the ends of each polypeptide. Each end is constructed using separate CONGEN

```
OPEN UNIT 66 NAME NTER.STS FORM WRITE                ! Status file
OPEN UNIT 60 NAME NTER.CG UNFORM WRITE               ! Conformations file
OPEN UNIT 70 NAME CGDATA:AMBER94TOPCG.INP FORM READ  ! Sidechain construction
                                                     ! rules file
OPEN UNIT 51 NAME CGDATA:AM94_EMAPGLY30.OMP FORM READ ! Glycine backbone energy map
OPEN UNIT 52 NAME CGDATA:AM94_EMAPALA30.OMP FORM READ ! Alanine backbone energy map
OPEN UNIT 53 NAME CGDATA:AM94_EMAPPRO30.OMP FORM READ ! Proline backbone energy map
OPEN UNIT 55 NAME CGDATA:AM94_PRO.CNS FORM READ      ! Proline constructor file

CGEN -                                               ! Beginning of multiline
-                                                    ! conformational search
STATUS UNIT 66 END -                                 ! Run status output
CHECKPOINT UNIT -1 TIME 60 NODE 1000000 END -        ! Specifies timing of status
-                                                    ! outputs, and no checkpointing
SEARCH EVAL ENERGY END -                             ! Directed search
MAXLEAF 1000 -                                       ! Limit number of conformations
-                                                    ! to 1000
TREE LIMIT 500000 PRTFRQ 500000 END -                ! Limits on the search tree size
NBCG CUTNB 12.0 CTONNB 98.0 CTOFNB 99.0 ATOM END -   ! Non-bonded energy calculation
HBCG CUTHB 0.5 CTONHB 98.0 CTOFHB 99.0 -             ! Amber has no hydrogen bond
     CUTHBA 90.0 CTONHA 90.0 CTOFHA 90.0 END -       ! energy, so this turns it off.
-                                                    ! Now build each backbone and
-                                                    ! and sidechain successively.
BACK CISTRANS REVERSE STARTRES A 7 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 7 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 6 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 6 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 5 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 5 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 4 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 4 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 3 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 3 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 2 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 2 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
BACK CISTRANS REVERSE STARTRES A 1 MAXEVDW 10 GRID 30 $ -
SIDE STARTRES A 1 MAXEVDW 10 SIDEOPT INDE VAVOID SGRID SELECT 30 30 60 END EVAL E $ -
EVL MINI ENERGY END $ -                              ! Energy evaluation
WRITE CUNIT 60 MINCUT 3.0 $ -                        ! Write results
ERINGPRO 50 GLYMAP 51 ALAMAP 52 PROMAP 53 PROCONS 55 -! Maps and proline constructors
GLYEMAX 5 ALAEMAX 5 PROEMAX 5 STUNIT 70              ! Energy limits.
N-terminal segment in phosphotransferase IIA domain  ! Conformation file title.
*
```

Fig. 9. Amino terminal construction.

runs, using successive backbone and side-chain degrees of freedom so that each residue is completely built before the next one is started. A directed search is used because the number of possible conformations can be very large if more than three residues are constructed. **Figure 9** illustrates the construction of seven residues on the amino terminus of a protein.

3.7. Short Constrained Minimization

The final step in the construction is a set of short constrained minimizations to clean up any strain in the molecule. You should also check at this point to see if all the coordinates have been constructed. **Figure 10** illustrates the minimi-

```
CONS HARM FORCE 20.0 ALL          ! Set constraints on current position
MINI ABNR NSTEP 50                ! Minimize.
CONS HARM FORCE 20.0 ALL          ! Reset constraints on current position
MINI ABNR NSTEP 50                ! Minimize.
CONS HARM FORCE 20.0 ALL          ! Reset constraints on current position
MINI ABNR NSTEP 50                ! Minimize.
CONS HARM FORCE 20.0 ALL          ! Reset constraints on current position
MINI ABNR NSTEP 50                ! Minimize.

PRINT COOR CLEAR COOR X EQ 9999.0 ! Display all initialized coordinates
```

Fig. 10. Constrained minimization input.

zation step and the check of coordinates. The value, 9999, is used to identify initialized coordinates in CONGEN.

4. Notes

1. This is a part of CONGEN that needs some redesign.
2. The known structure will help guide the placement of the loop atoms. Our work on antibody reconstruction***(23)*** shows this principle with construction of antibody variable domains.
3. In this context, "high" is somewhat subjective, but typically, large positive energies for loops is not acceptable.

Acknowledgments

The author acknowledges the innumerable contributions of Jiri Novotny to the development and application of CONGEN, and thanks Malcolm Davis, Donna Bassolino-Klimas, and Susan Cottingham for their work on CONGEN.

References

1. Dunbrack, R. L. J., Gerloff, D. L., Bower, M., Chen, X., Lichtarge, O., and Cohen, F. E. (1997) Meeting review: the second meeting on the critical assessment of techniques for protein structure prediction (CASP2), Asilomar, CA, December 13–16, 1996. *Fold. Des.* **2,** R27–R42.
2. Jones, T. A. and Thirup, S. (1986) Using known substructures in protein model building and crystallography. *EMBO J.* **5,** 819–822.
3. Martin, A. C. R., Cheetham, J. C., and Rees, A. R. (1991) Molecular modeling of antibody combining sites. *Methods Enzymol.* **203,** 121–153.
4. Bajorath, J., Stenkamp, R., and Aruffo, A. (1993) Knowledge-based model building of proteins: concepts and examples. *Protein Sci.* **2,** 1798–1810.
5. Sanchez, R. and Sali, A. (1997) Advances in comparative protein-structure modelling. *Curr. Opin. Struct. Biol.* **7,** 206–214.
6. Snow, M. E. and Amzel, L. M. (1986) Calculating three-dimensional changes in protein structure due to amino-acid substitutions: the variable region of immunoglobulins. *Proteins: Struct. Funct. Genet.* **1,** 267–279.

7. Collura, V., Higo, J., and Garnier, J. (1993) Modeling of protein loops by simulated annealing. *Protein Sci.* **2,** 1502–1510.
8. Li, H., Tejero, R., Monleon, D., Bassolino-Klimas, D., Abate-Shen, C., Bruccoleri, R. E., and Montelione, G. T. (1997) Homology modeling using simulated annealing of restrained molecular dynamics and conformational search with CONGEN: application in predicting the three-dimensional structure of murine homeodomain Msx-1. *Protein Sci.* **6,** 956–970.
9. Moult, J. and James, M. N. G. (1986) An algorithm for determining the conformation of polypeptide segments in proteins by systematic search. *Proteins: Struct. Funct. Genet.* **1,** 146–163.
10. Bruccoleri, R. E. (1993) Application of systematic conformational search to protein modeling. *Mol. Sim.* **10,** 151–174.
11. Cardozo, T., Totrov, M., and Abagyan, R. (1995) Homology modeling by the ICM method. *Proteins: Struct. Funct. Genet.* **23,** 403–414.
12. G¯o, N. and Scheraga, H. A. (1970) Ring closure and local conformational deformations of chain molecules. *Macromolecules* **3,** 178–187.
13. Bruccoleri, R. E. and Karplus, M. (1985) Chain closure with bond angle variations. *Macromolecules* **18,** 2767–2773.
13a. Bruccoleri, R. E. and Karplus, M. (1987) Prediction of the folding of short polypeptide segments by uniform conformational sampling. *Biopolymers* **27,** 137–168.
14. Ramachandran, G. N., Ramakrishnan, C., and Sasisekharan, V. (1963) Stereochemistry of polypeptide chain configurations. *J. Mol. Biol.* **7,** 195–199.
15. Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy minimization and dynamics calculations. *J. Comput. Chem.* **4,** 187–217.
16. Pearl, J. and Korf, R. E. (1987) Search techniques. *Ann. Rev. Comput. Sci.* **2,** 451–467.
17. Pincus, M. R., Klausner, R. D., and Scheraga, H. A. (1982) Calculation of the three dimensional structure of the membrane-bound portion of melittin from its amino acid sequence. *Proc. Nat. Acad. Sci. USA* **79,** 5107–5110.
18. Zhu, P. P., Nosworthy, N., Ginsburg, A., Miyata, M., Seok, Y. J., and Peterkofsky, A. (1997) Expression, purification, and characterization of enzyme IIA(glc) of the phosphoenolpyruvate:sugar phosphotransferase system of Mycoplasma capricolum. *Biochemistry* **36,** 6947–6953.
19. Genetics Computer Group (GCG) (1996) Wisconsin package version 9, Madison, WI.
20. Sali, A. (1995) Modeling mutations and homologous proteins. *Curr. Opin. Biotechnol.* **6,** 437–451.
21. Kabsch, W. and Sander, C. (1985) Identical pentapeptides with different backbones. *Nature* **317,** 207.
22. Bruccoleri, R. E. (1995) Energy directed conformational search of protein loops and seg ments, in *Proceedings of the DIMACS Workshop: Global Minimization of Nonconvex Energy Functions: Molecular Conformation and Protein Folding* (Pardalos, P. M., Shalloway, D., and Xue, G., eds.), American Mathematical Society, Providence, RI, pp. 15–28.
23. Bruccoleri, R. E., Haber, E., Novotny, J. (1988) Structure of antibody hypervariable loops reproduced by a conformational search algorithm. *Nature* **335,** 564–568. See Errata, vol. 336, p. 266.

12

The Dead-End Elimination Theorem:

Mathematical Aspects, Implementation, Optimizations, Evaluation and Performance

Marc De Maeyer, Johan Desmet, and Ignace Lasters

1. Introduction

The placement of amino acid side chains in a given fixed main-chain template forms a recurrent but nontrivial task in protein modeling. Even for a small set of side chains in a given protein, the degrees of freedom for the side chains lead to an enormous number of combinatorial possibilities, inevitably prohibiting a brute force approach to pinpoint the global minimum energy conformation (GMEC). The recognition of the existence of statistically relevant discrete combinations of the dihedral angles (called rotamers) of a side chain forms the basis of all current side-chain placement techniques ***(1)***. Several research groups have published methods to predict the side chain positions in a fixed protein main-chain trace (review in **refs. *2*** and ***3***)

The dead-end elimination (DEE) algorithm is able to efficiently tackle the combinatorial problem, i.e., the problem of finding the globally optimal arrangement of a collection of side chains attached to a fixed main-chain structure. Contrary to most of the other methods, which try to tackle the combinatorial problem directly, the DEE method is based on an elimination technique. Avoiding a combinatorial explosion, the DEE method detects and eliminates iteratively those rotamers that cannot be members of the GMEC. The rotamer elimination occurs on the basis of energetic criteria balancing rotamers against each other by comparing their main-chain interaction energy and a lower or higher limit for their interactions with the other side chains of the protein.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

Since the discovery of the DEE algorithm in 1992 ***(4)*** several major theoretical and practical improvements and fields of application have matured the method as a novel and promising tool in protein modeling and design.

The first and most natural application has been in the field of homology modeling. The method is highly suitable for local optimization of a few side chains (core remodeling, interface between proteins, etc.) as well as the redesign of complete proteins. Using the DEE theorem, it was possible to unravel the fine detailed side chain–main chain and side chain–side chain interactions working concurrently, stabilizing the protein structure ***(5,6)***. The algorithm has also been transposed to the docking of small molecules ***(7)*** and peptides ***(8)*** to proteins. The method was also very instrumental in the initial positioning of the side chains in a crude X-ray density map leading to the 3D structure ***(9)*** and the elucidation of the high-oxygen affinity of the trematode myoglobins ***(10)***. Recently, the DEE method has been used in a protein design automation cycle ***(11)***, a de novo protein design experiment ***(12)***, and the exploration of the sequence space that is compatible with a given scaffold ***(13)***, forming an even less trivial application field.

2. Theoretical Considerations

2.1. General Concepts

2.1.1. The Global Minimum Energy Conformation (GMEC)

The ultimate goal of any side-chain placement method is the prediction of the conformation of the studied protein as it occurs in nature. In this chapter we concentrate on how we reach the so-called minimum energy conformation (GMEC) and assume that this is also the conformation of the protein in its active form. In this respect, it is worth stressing the fact that the DEE method is, first of all, very instrumental in reducing the huge combinatorial complexity of the tackled problem. Because the DEE-method eliminates, in a systematic and rigorous way, all rotamers incompatible with this GMEC state, it is not at all guaranteed that for all residues all rotamers but one can be eliminated. On the other hand we have supplemented the DEE method with additional techniques to track down this smaller combinatorial problem. Up to now, the GMEC conformation was reached for all the studied proteins. These additional methods are also discussed in this chapter.

2.1.2. The Side-Chain Conformations (Rotamers)

It has been shown in the past that side-chain conformations can be well described by a library of possible rotameric states ***(1,14–16)***. For each of the 20 amino acids these rotamers describe a statistically relevant dihedral angle combination. Each rotamer is characterized by a set of χ angles, following the

International Union of Pure and Applied Chemistry (IUPAC) rules ***(17)***. For an overview of the currently used rotamer libraries we refer to the work of De Maeyer et al. ***(6)***. In general, two types of rotamer libraries are used. The first library ***(13)*** is mainly based on the Ponder and Richards analysis ***(16)*** and contains for each amino acid side-chain type a number of different rotamers. These are denoted as a main-chain–independent rotamer library (MIRL). The second class of libraries splits each amino acid rotamer set into several subsets. These classes are driven by the ϕ–ψ angle combination for the amino acid under study. This library is referred to as the main-chain–dependent rotamer library (MDRL). This library has the advantage that at each position the number of rotamers is limited compared to the first type of library. On the other hand, it has been shown by Schrauber et al. ***(18)*** that most libraries suffer from incompleteness, resulting in the inability to correctly predict the GMEC ground state. In addition, we have convincingly shown that the accuracy of the prediction reflects the accuracy of the library ***(6)***. In this work, we have started from the rotamer library used in the work of Lasters et al. ***(19)*** containing 275 rotamers distributed over 17 amino acid types. This library was created from the standard Ponder and Richards library ***(16)***, completed with all physically possible rotamers (standard *gauche* and *trans* conformations) not present in the original set and supplemented with 65 additional rotamers, originating from the lack of well-defined rotameric states for the amide plane orientation and carboxylate groups of Asn, Gln, Asp, and Glu ***(19)***. Combining this library with the analysis of Schrauber et al. ***(18)*** leads to the new basic rotamer library with 331 elements. In addition, by taking one or more user-defined steps around the rotamer χ angles, an even more detailed library is obtained. In the work of Ponder and Richards ***(16)*** this step-size corresponds to the standard deviation in c angles distribution in their analysis of 19 well-resolved and refined proteins. In all tests described in this study we expanded the library by taking two steps of 10° around the χ_1 angle of the aromatics (Phe, Tyr, His, Trp), and for each of these new rotamers we took 2 steps of 20° around the χ_2 angle. This enlarges the rotamer library to 859 elements, referred to as the "large library" ***(6)***. In previous work we used a subset of this large library, referred to as the "small library," of only 213 rotamers.

2.1.3. The Conformational Energy

When we describe the side chain conformation as a rotamer, each rotatable side-chain i may adopt some rotameric state r selected from the library of all possible conformations. The conformational energy corresponding with a selected rotamer state that is embedded in the template (i.e., the main-chain and all the other fixed side chains not to be modeled) can be written as

$$E_{\text{tot}} = E_{\text{template}} + \sum_i E(i_r) + \sum_i \sum_j E(i_r, j_s) \quad i < j \tag{1}$$

where E_{template} is the self-energy of the template, $E(i_r)$ the energy of the side-chain atoms of rotamer i_r, including their self-energy and the interaction with the template, and $E(i_r,j_s)$ the nonbonded pairwise interaction between rotamers i_r and j_s. It is clear that $E(i_r,j_s)$ in itself is composed of pairwise atom interaction energies. The dimensions of the terms $E(i_r)$ and $E(i_r,j_s)$ grow, respectively, linearly and quadratically as a function of the number of atoms in the studied system, meaning that the system under consideration is tractable by modern computers and does not require excessive amounts of memory.

2.2. The Original DEE Theorem

The original DEE theorem ***(4)*** states that i_r is dead ending (meaning not being part of the GMEC conformation) with respect to another rotamer i_t, if **Eq. 2** holds true

$$E(i_r) + \sum_j \min_s E(i_r,j_s) > E(i_t) + \sum_j \max_s E(i_t,j_s) \quad i \neq j \tag{2}$$

This means that only one case j_t, satisfying **Eq. 2** has to be found in order for i_r to be dead ending. In **Eq. 2** the terms *min* and *max* refer to the interaction energy of, respectively, the "best" and the "worst" interacting rotamer of residue j. In words, this inequality means that i_r must be dead ending if the energy of its best possible interactions with the surroundings (left-hand side of **Eq. 2**) is larger than that for another rotamer taken in its worst situation (right-hand side of **Eq. 2**). As a consequence, we can state that, in searching the GMEC, the rotamer i_r can safely be qualified as a dead-end rotamer and discarded from further considerations. In practice, it is useful to rewrite **Eq. 2** into **Eq. 2'** as follows:

$$\frac{E(i_r) + \sum_j \min_s E(i_r,j_s) > \min_n \left[E(i_n) + \sum_j \max_s E(i_n,j_s)\right]}{i \neq j} \tag{2'}$$

Indeed, from this equation we learn that if i_r is dead ending relative to another rotamer i_t then this rotamer must also be dead ending relative to the rotamer i_n, which from the right-hand side of the **Eq. 2'**, shows the lowest possible value. We may also say that, if i_r is not dead ending relative to such rotamer i_n, it will not be qualified as dead ending versus any other rotameric state for residue i. Accordingly, when searching for dead-end rotamers, one can make an ordered list of the worst rotamer interaction energies for all rotamers of residue i and use the best of these worst energies as a threshold value for the possible elimination of rotamers i_r — as seen from the left-hand side of **Eq. 2'**. This equation has been plotted in the total conformational space versus the interaction energy of i with the other rotamers in **Fig. 1**.

2.3. Enhancements of the DEE Theorem

A more powerful form of the DEE criterion has recently been formulated by Goldstein ***(20)***. This criterion ascertains that i_r is dead ending if **Eq. 3** is fulfilled.

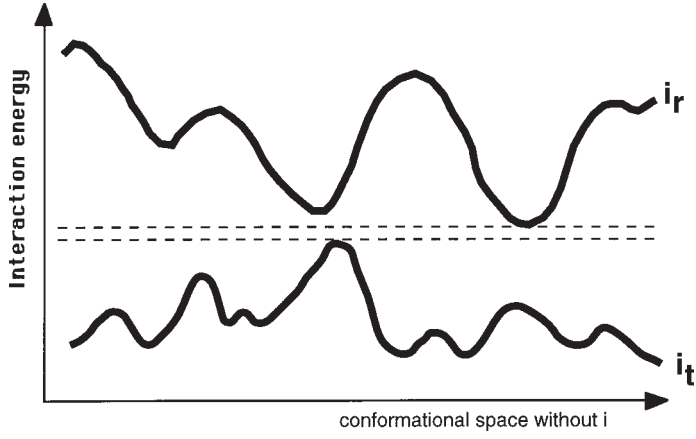


Fig. 1. Illustration of the original DEE theorem. The curve shows arbitrary interaction energy profiles for two rotamers i_r, i_t, with each of the possible rotamer combinations for the residues j, $j \neq i$. Following **Eq. 2**, the rotamer i_r is dead-ending versus rotamer i_t because the minimal interaction energy with all other rotamers is worse than the maximum of all interaction energies of rotamer i_t.

$$E(i_r) - E(i_t) + \sum_j \min_s \left[E(i_r,j_s) - E(i_t,j_s)\right] > 0 \quad i \neq j \tag{3}$$

The DEE criterion in **Eq. 3** qualifies i_r as dead ending if we can always lower the energy by taking rotamer i_r instead of i_t while keeping the other rotamers fixed, the interactions being taken with respect to the same j_s. Clearly, this modified criterion is less restrictive than **Eq. 2** and consequently has an increased effectiveness ***(20)***. However, it should be noted that this criterion is slower in execution time as compared to **Eq. 2**. This equation has been depicted in **Fig. 2**. However, we have shown that DEE **Eq. 2** remains of great value when searching for dead-end rotamer pairs. This variant of the DEE inequality is further discussed in view of two important extensions in **Subheading 2.6.**

2.4. Extending the DEE to Rotamer Pairs

So far, the DEE has been applied to the interaction of single rotamers. This criterion alone is, in practical cases, not powerful enough to determine the global minimum energy conformation. In the original paper of Desmet et al. ***(8)*** the DEE principle is already extended to rotamer pairs. Of course, this principle can be extended to group several side chains into a "superrotamer" (R_i). This R_i contains all possible combinations of the individual side-chain rotamers. For ease of reading, we restrict the equations to pairs of rotamers. The intrinsic energy for a rotamer pair can be written as **Eq. 4**:

$$\varepsilon([i_r,j_s]) = E(i_r) + E(j_s) + E(i_r,j_s)\ i \neq j \tag{4}$$

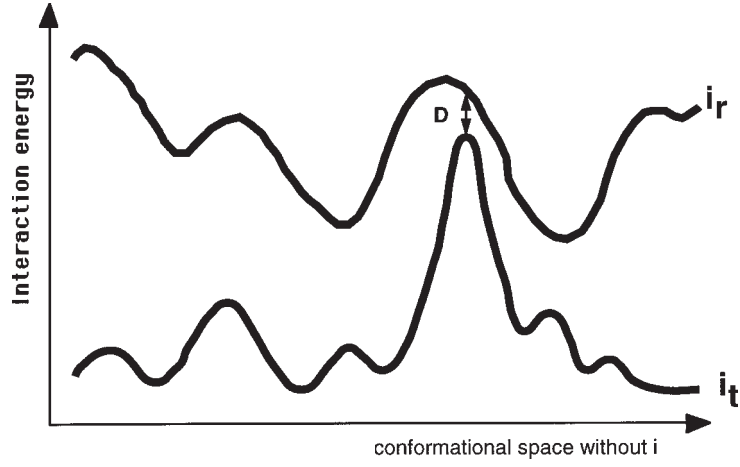


Fig. 2. Illustration of the variant of the DE criterion following **Eqs. 3** and **10**. The curve shows arbitrary energy profiles for two rotamers i_r, i_t, with each of the possible rotamer combinations for the residues j, $j \neq i$. This criterion qualifies i_r as dead-ending versus rotamer i_t because the minimal distance D between the two profiles is positive.

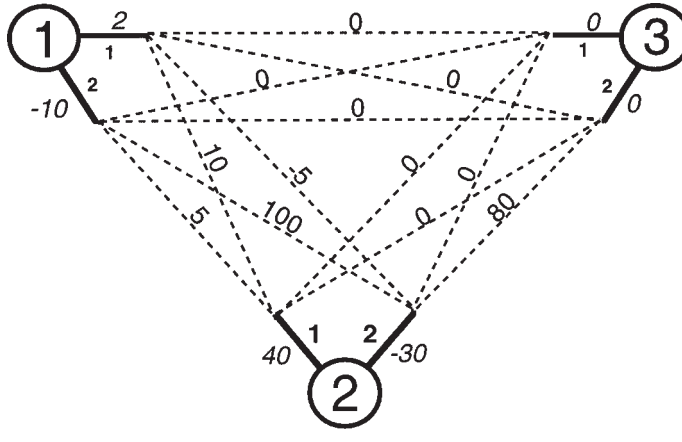


Fig. 3. This figure contains a simple three-residue system, illustrating that dead-ending rotamer pairs cannot simply be ignored when reiterating the DEE theorem. This figure has to be interpreted with the help of **Tables 2–4**, while **Table 1** enumerates all possible rotamer interactions. The three residues (i) are represented by circles and the rotamers (i_r) by bold lines and labels. The inherent energy $E(i_r)$ of rotamer i_r is indicated in italic. This inherent energy is the sum of the conformational energy of i_r in interaction with itself and the surrounding template. The dashed lines denote the pairwise rotamer interaction energy terms $E(i_r j_s)$ of **Eq. 1**. All energy values are given in arbitrary units.

Table 1
Calculation of the GMEC Energy of the Simple System Shown in Fig. 1

Combination residue i and rotamer n (i_n)	E_{global}
$1_1 2_1 3_1$	(2 + 40) + 10 = 52
$1_1 2_1 3_2$	(2 + 40) + 10 = 52
$1_1 2_2 3_1$	**(2 – 30) –5 = –33**
$1_1 2_2 3_2$	(2 – 30) – 5 + 80 = 47
$1_2 2_1 3_1$	(–10 + 40) + 5 = 35
$1_2 2_1 3_2$	(–10 + 40) + 5 = 35
$1_2 2_2 3_1$	(–10 –30) + 100 = 60
$1_2 2_2 3_2$	(–10 –30) + 100 + 80 = 140

Exhaustive calculation of E_{global} for all possible residue–rotamer combinations. Each combination is defined by a six-digit number where the subscript denotes the chosen rotamer. The rotamer combination yielding the lowest energy (indicated in bold) corresponds to the GMEC.

A rotamer pair interacting with some rotamer k_t has an interaction energy of

$$\varepsilon([i_r,j_s],k_t) = E(i_r,k_t) + E(j_s,k_t) \quad i,j \neq k \tag{5}$$

Applying the DEE theorem to rotamer pairs $[i_r,j_s]$ following DEE criteria can be deduced from the **Eqs. 2** and **3**, yielding respectively

$$\varepsilon([i_r,j_s]) + \sum_k \min_t \varepsilon([i_r,j_s],k_t) > \varepsilon([i_u,j_v]) + \sum_k \max_t \varepsilon([i_u,j_v],k_t) \quad i,j \neq k \tag{6}$$

$$\varepsilon([i_r,j_s]) - \varepsilon([i_u,j_v],k_t) + \sum_k \min_t \{\varepsilon([i_r,j_s],k_t) - \varepsilon([i_u,j_v],k_t)\} > 0 \tag{7}$$

From a computational and programmatorial point of view, it is important to note that the implementation of superrotamers becomes quite complex. In practice it is even impossible to use rotamer pairs in the early stages of the program. It is worth stressing the fact that a dead-ending pair (DEP) means that it is the *combination* of the two rotamers that is incompatible with the GMEC, whereas one of them may well be part of this GMEC. Only in particular cases is it allowed to ignore DEPs. This is ruled by the fuzzy-end elimination (FEE) criterion explained in **Subheading 2.5**. For a full proof of the theorem, we refer to the work of Lasters and Desmet ***(21)***.

2.5. The Fuzzy-End Elimination Problem

Let us use a simple case to shed more light on the origin of the FEE theorem. **Figure 3** depicts a simple three-residue system, each residue having two rotamers. This is a small system, yet complex enough to explain the problems, allowing easily the calculation of the GMEC. In **Table 1** all possible combina-

Table 2
Applying the Dead-End Elimination Theorem for the Single Rotamers in Fig. 1

Combination residue i and rotamer n (i_n)	E_{inherent} + BEST	E_{inherent} + WORST	Conclusion
1_1	2 – 5 = –3	2 + 10 = 12	
1_2	–10 + 5 = –5	–10 + 100 = 90	No DE rotamer found
2_1	40 + 5 = 45	40 + 10 = 50	
2_2	–30 – 5 = –35	–30 + 100 + 80 = 150	No DE rotamer found
3_1	0 + 0 = 0	0 + 0 = 0	
3_2	0 + 0 = 0	0 + 80 = 80	No DE rotamer found

Using the data of **Fig. 1**, the dead-end elimination theorem is utilised for each rotamer of each residue. With **Eq. 2** the best and the worst interaction energies for each of the rotamers are shown in the BEST and WORST labeled column. Upon application of the dead-end elimination criterion for the system under study it is found that there can be no rotamers qualified as dead-ending.

Table 3
Applying the Dead-End Elimination Theorem for Rotamers Pairs in Fig. 1.

Rotamer pair	E_{inherent} + BEST	E_{inherent} + WORST
1_12_1	(2 + 10 + 40) + 0 = 52	(2 + 10 + 40) + 0 = 52
1_12_2	(2 – 5 –30) + 0 = –33	(2 – 5 – 30) + 80 = 47
1_22_1	(–10 + 5 + 40) + 0 = 35	(–10 + 5 + 40) + 0 = 35
1_22_2	(–10 + 100 –30) + 0 = 60	(–10 + 100 –30) + 80 = 140

Application of **Eq. 6** to all rotamer pairs for the residues 1 and 2 of **Fig. 4** is in detail listed in this table. It can be concluded that the rotamer pairs 1_12_1 and 1_22_2 of the residue pair 1 and 2 are dead-ending.

tions are enumerated and this identifies rotamer combination $1_12_23_1$ as the GMEC. Application of the original dead-end elimination criterion of **Eq. 2** on each of the rotamers of the minisystem does not discover any dead-end rotamer. The values are enumerated in **Table 2**. Further utilization of the DEE for rotamer pairs allows the identification for residues 1 and 2 that DEPs can be found. From **Table 3** and **Fig. 3** it becomes clear that the rotamer pairs 1_12_1 is a DEP, because in its best possible interaction with the surrounding template and other rotamers, it is in a worst situation than the worst possible interaction energy for the rotamer combination 1_22_1, which is the best of the worst interac-

Table 4
Showing the Origin of the FEE Theorem

Combination residue i and rotamer n (i_n)	E_{inherent} + BEST	E_{inherent} + WORST
1_1	2 – 5 = –3	2 – 5 = –3 (wrong)
1_2	–10 + 5 = –5	–10 + 5 = –5 (wrong)

Discarding the two dead-ending rotamer pairs 1_12_1 and 1_22_2 obtained from **Table 3**, from further consideration in the evaluation of the dead-end elimination criterion for single rotamers leads to the erroneous result that rotamer 1_1 would be a dead-end rotamer (*see also* **Table 1**)

tion energies (*see* **Table 3**). Upon reapplication of the dead-end criterion for single rotamers, intuitively one would be appealed to remove the previously found DEP from further consideration in the evaluation of the best and worst interaction energies as prescribed by the DEE criterion. As shown in **Table 4** this may lead to erroneous results if one would think that rotamer 1_1 is dead ending, which is in contradiction with the data in **Table 1**, identifying rotamer 1_1 being a member of the GMEC. As a consequence we are confronted with a serious problem, as it is unclear how DEPs may contribute to further elimination of dead-ending single rotamers. The formulation and proof of a new DE criterion removes this uncertainty. This theorem has been called "fuzzy-end" because if the energy contributions of rotamer-pairs are excluded from the "worst" interaction terms of the DEE criterion, then there is no guarantee that rotamers that satisfy this inequality are incompatible with the GMEC, i.e., they may or may not be members of the GMEC.

In perspective of the previous remarks, one may wonder whether DEPs are at all of any practical use in tracking down the combinatorial rotamer tree. First, it has been shown and proven ***(21)*** that the interaction energies of DEPs can safely be removed from the "best" interaction energies of the DEE criterion. In this case, novel single dead-end rotamers could be identified by the simple fact that this left-hand side might become augmented as compared to the case where all rotamer pairs would have been considered. Second, if, for a given rotamer i_r, all the possible rotamer pairs $[i_r,j_s]$ made with another residue j are dead ending (denoted as an all-DEP case), then of course i_r can never be member of the GMEC and, as a consequence, i_r may safely be removed from the current set of remaining rotamers. This situation is illustrated in **Fig. 4**. A third mechanism by which dead-ending single rotamers may be detected on the basis DEPs is explained in the *logical pairs theorem* in **Subheading 2.7**.

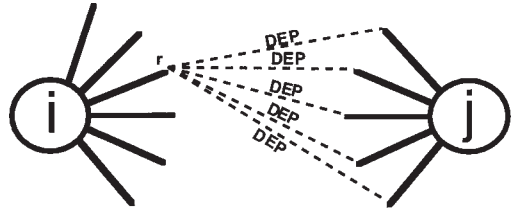


Fig. 4. Illustration of the usage of dead-ending pairs in the elimination of single dead-ending rotamers based on logical grounds. In this example, the rotamer i_r is in a dead-ending pair situation with all of the remaining rotamers of residue j.

2.6. Optimizing the DEE Criterion

In order to demonstrate further optimizations to the method, we first introduce the concept of reduced energies. The reduced energy can be written as

$$E'(i_r,j_s) = \frac{E_{\text{template}}}{C_n^2} + \frac{E(i_r)}{n-1} + \frac{E(j_s)}{n-1} + E(i_r,j_s) \tag{9}$$

where n is the number of residues and $C_n^2 = [n.(n-1)]/2$.

Using the reduced energy terms E', the DEE criterion **Eq. 3** can be rewritten more compactly as

$$\sum_j \min_s \left[E'(i_r,j_s) - E'(i_t,j_s)\right] > 0 \quad i \neq j \tag{10}$$

Equation 10 is more attractive than **Eq. 1** and does not affect the enhanced DE criterion presented in **Eq. 3**.

This criterion has been further generalized ***(20)*** where the T alternate rotamers $i_t \neq i_r$ are given weight factors C_t in the DE evaluation of i_r. The generalized criterion, expressed in reduced energy terms E' reads as follows:

$$\sum_{j \neq i} \min_s \left[E'(i_r,j_s) - \sum_{t=1}^{T} C_t \cdot E'(i_t,j_s)\right] > 0 \tag{11}$$

where $C_t \geq 0$ and $\sum C_t = 1$. However, the determination of these weight coefficients was left undetermined. We have presented ***(19)*** an iterative procedure to determine these weight factors that may lead to a more efficient elimination of single rotamers. First, single dead-ending rotamers are eliminated until exhaustion using the criterion in **Eq. 10**. In the following steps, we describe the program flow to be executed.

1. As will become clear, we need the interaction energies to be positive values. Because the GMEC is not affected by shifting all energy terms by some constant value, all interaction energies are augmented by a constant that is chosen appropriately.

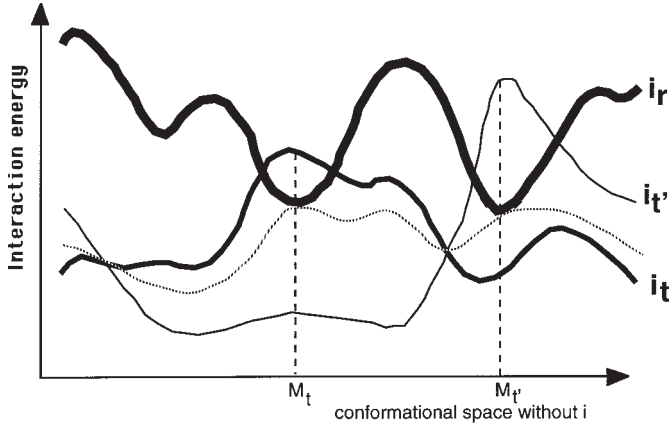


Fig. 5. Some of the notations in the equations are illustrated by these arbitrary energy profiles. The curves depict interaction energies *I* of various rotamers i_r, i_t, $i_{t'}$ with each of the possible rotamer combinations for the residues $j, j \neq i$. These rotamer combinations are mapped arbitrarily on the *x*-axis with each point on this axis denoting some specific combination. Thus these points constitute the conformational space $S_{\neg i}$, (read as *S not i*). $\boldsymbol{M}_t$ and $\boldsymbol{M}_{t'}$ denote those rotamer combinations that result from the DE criterion ***(3)***. These points correspond to rotamer combinations for which the difference in interaction energies $(I_r - I_t)$ or $(I_r - I_{t'})$ is minimal. Using the DEE criterion ***(3)***, i_r would not be qualified as dead-ending, as its interaction curve is not always above the others. However, by using proper weighting factors in the generalized DE criterion ***(11)*** it is seen that i_r is dead-ending. In the shown example, weight factors of 0.7 and 0.3 are applied to the i_t and $i_{t'}$ curves, respectively, resulting in the dashed curve that nowhere is above the i_r curve. The computation of the weight factors is explained in the text.

2. For each of the rotamers $i_t \neq i_r$, the *min* operators of **Eq. 10** result in the identification of a rotamer combination, one for each $j \neq i$ residue. We define $\boldsymbol{M}_t$ as the ensemble of rotamers for the $j \neq i$ residues that result from the min-operators in **Eq. 10** as is graphically illustrated in **Fig. 5**. Thus, $\boldsymbol{M}_t$ can be seen as a point in the conformational space S ¬ i (read as *"S not i"*) for which the $j \neq i$ residues have done their very best to interact favorably with i_r and unfavorably with i_t. Define now for any rotamer *x* of residue *i*

$$I_x(\boldsymbol{M}_t) = \sum_{j \neq i} E'(i_x, j_{\boldsymbol{M}_t}) \text{ where } j_{\boldsymbol{M}_t} \in \boldsymbol{M}_t \tag{12}$$

In other words, $I_x(\boldsymbol{M}_t)$ is the sum of interaction energies of some rotamer i_x evaluated at the rotamer combinations $j_{\boldsymbol{M}_t}$ defined by $\boldsymbol{M}_t$. In this step we evaluate the interaction energies for each i_x at each of the available $\boldsymbol{M}_t$ points in the S ¬ i space.

3. In a sense, the points $\boldsymbol{M}_t$ determined in the preceding phase define a list of "critical" points where the interaction energies for at least one rotamer exceeds maximally that computed for the DE candidate i_r, thus preventing i_r to become a dead-ending rotamer (at these points, i_r is a better choice than i_t). Clearly, we seek to remedy this situation by estimating proper weight factors for each of the I_t energy profiles. To this end, we solve first the following min–max problem.

Maximize $W = \sum_{x=1}^{T} w_x$ subject to the following constraints for each of the

$$w_x > 0 \text{ and } I_r(\boldsymbol{M}_t) - \sum_{x=1}^{T} I_x(\boldsymbol{M}_t) \cdot w_x > 0 \tag{13}$$

points $\boldsymbol{M}_t$. These constraints are equivalent to urging that the weight coefficients are such that the I_r curve in **Fig. 1** exceeds at each $\boldsymbol{M}_t$ the sum of the other weighed curves. This problem can be efficiently solved using a numerical analysis method known as linear programming, for which detailed algorithms have been published in textbook form ***(22)***. Depending on the obtained W value we follow a different route.

Case $W < 1$: Exit the procedure, as i_r cannot be dead ending by applying the generalized **Eq. 11**. Indeed, the situation $W < 1$ means that the sum of the weighed curves can only be pushed below the I_r curve by assigning weight coefficients for which the sum is smaller than unity. But we still have to normalize the weight coefficients by calculating $C_x = w_x/W$.

Because $0 \leq w_x \leq C_x$, the sum of the weighed curves will be shifted upward, thereby exceeding I_r for at least one of the points $\boldsymbol{M}_t$ (because the normalized coefficients w_x were already maximized). Thus the situation $W < 1$ implies that i_r cannot be considered as dead ending, and consequently forms an exit condition.

Case $W \geq 1$: Put $C_x = w_x/W$. Given the truth of **Eq. 13**, this guarantees that at each $\boldsymbol{M}_t$: $I_r(\boldsymbol{M}_t) - \sum_{x=1}^{T} C_x \cdot I_x(\boldsymbol{M}_t) \cdot > 0$. This follows immediately from $0 \leq C_x \leq w_x$, $E' \geq 0$, and the truth of **Eq. 13**.

We are now sure that the function $\sum_{x=1}^{T} C_x \cdot I_x(\boldsymbol{M}_t)$ is situated below $I_r(_{\boldsymbol{M}t})$ at each point $\boldsymbol{M}_t$. Of course, there is no guarantee that this will be the case at all points of $S \neg_i$. This question can be answered quickly, however, by applying the generalized dead-end criterion **Eq. 11** using the obtained set of weight coefficients. This may result in the identification of i_r as a dead-ending rotamer, and in this event we have a successful exit. In the other case, the computation of **Eq. 11** results in an additional critical point that is added to the list of points $\boldsymbol{M}_t$. This defines an additional constraint and we reiterate to the beginning of **step 3**.

Note that one cannot get caught in an endless iteration loop, because at each step W will decrease. Inevitably, once W drops below unity we automatically meet an exit condition. In practice, it is found that only a few iteration steps are needed before exiting the procedure.

2.7. The Logical Pairs Theorem

The concept of DEE can also be applied to rotamer pairs using the DEE criteria (**Eqs. 6** or **7**). DEPs may lead to the identification of additional single dead-ending rotamers thereby further tracking down the size of the rotamer conformational space. Previously, it has been shown ***(21)*** that DEPs may be safely ignored from the min terms in **Eq. 2**, leading to

$$E(i_r) + \sum_k \min_{\substack{s \\ \text{no DEP with } i_r}} E(i_r,j_s) > E(i_t) + \sum_j \max_s E(i_t,j_s) \quad i \neq j \tag{14}$$

With regard to the modified criterion (**Eq. 3**) DEPs can be safely ignored from the left-hand terms, which leads to the following criterion

$$\sum_j \min_{\substack{s \\ \text{no DEP with } i_r}} [E'(i_r,j_s) - E'(i_t,j_s)] > 0 \quad i \neq j \tag{15}$$

The validity of this criterion can be shown following the same strategy as proposed previously ***(21)***, and along the same lines as exemplified in the proof of **Eq. 11** (for a full proof, see the Appendix of **ref. *21***). To increase the usefulness of **Eq. 15**, a modified form is used in practice:

$$\sum_j \min_s [P(i_r,j_s) - E'(i_t,j_s)] > 0 \quad i \neq j \tag{16}$$

where

$$P(i_r,j_s) = E'(i_t,j_s) \text{ if } [i_r,j_s] \neq \text{DEP}$$

$$P(i_r,j_s) = \infty \text{ if } [i_r,j_s] = \text{DEP}$$

The advantage of this criterion is that it eliminates automatically rotamers i_r that form DEPs with each of the rotamers of some residue j. Clearly, in such a case, which we denote as an all-DEPs case, i_r cannot be a member of the GMEC and thus i_r becomes excluded solely on logical grounds.

There is also another logical mechanism by which a rotamer i_r may be declared as dead-ending. Suppose that in the course of the DE elimination only one rotamer remains for some residue j. Clearly this rotamer, denoted as j_g, must be a member of the GMEC. Consequently, all rotamers that are part of DEPs that contain j_g are bound to be dead ending in a logical sense, and thus can be eliminated without even requiring the evaluation of **Eq. 16**. This situation is graphically shown in **Fig. 6**.

It is interesting to unravel the prevailing mechanism in the elimination of single rotamers by **Eq. 16**. In addition to its theoretical interest, the obtained insights will lead to an optimization in the algorithmic implementation of the DEPs computation. The following theorem, referred to as the *logical pairs* theorem, is instrumental for this discussion.

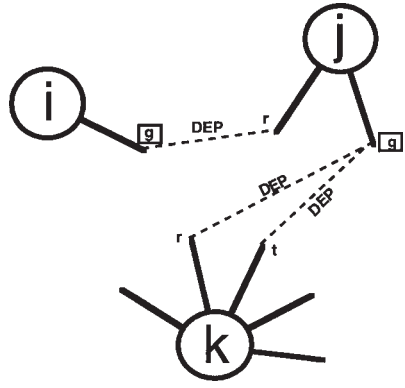


Fig. 6. Illustration of the usage of dead-ending pairs in the elimination of single dead-ending rotamers based on logical grounds. This figure illustrates the elimination of rotamers as a cascade effect triggered by the identification of a uniquely defined residue rotamer, indicated with the letter *g*. Given the shown DEP cases, this leads to the identification of three other dead-ending rotamers j_r, k_r, and k_t.

Given that i_r is eliminated by evaluating the DE criterion (**Eq. 3**) relative to another rotamer i_t, it follows that all pairs $[i_r,j_M]$ where the j_M rotamers are determined from the min operators of **Eq. 3** are dead ending by the DE criterion (**Eq. 7**) for rotamer pairs.

The following corollary to this theorem is of interest. First, we have eliminated until exhaustion all dead-ending single rotamers, and subsequently we screen for DEPs. Suppose now that $[i_r,j_d]$ is DEP relative $[i_t,j_d]$. Such situations will often occur, as all it takes is that i_r has an inherently bad interaction with j_d, whereas i_t is not in conflict with j_d. However, from the logical pairs theorem j_d cannot result from the min operator in the DE criterion for rotamers. Otherwise, i_r would be a dead-ending rotamer, which is in contradiction with the foregoing, given that all single dead-ending rotamers have been eliminated. As a consequence, in this case, **Eq. 15** becomes identical to the original criterion (**Eq. 3**), their min operators yielding the same rotamer j_M, thereby precluding the identification of an additional dead-ending rotamer. A different j_M can be selected only if the j_M determined by **Eq. 3** is implied in a DEP with i_r, whereas $[i_r,j_M]$ is not DEP relative $[i_t,j_M]$.

From the foregoing reasoning, it becomes clear that a predominant mechanism by which the removal of DEPs from **Eq. 15** is contributing to the further elimination of single rotamers is by eliminating rotamers i_r that make DEPs with all rotamers of another residue j and this is indeed observed in practice. Importantly, all-DEPs cases can be computed much more rapidly as compared to the straight computation of DEPs. Indeed, as shown in **Fig. 7**, the implemen-

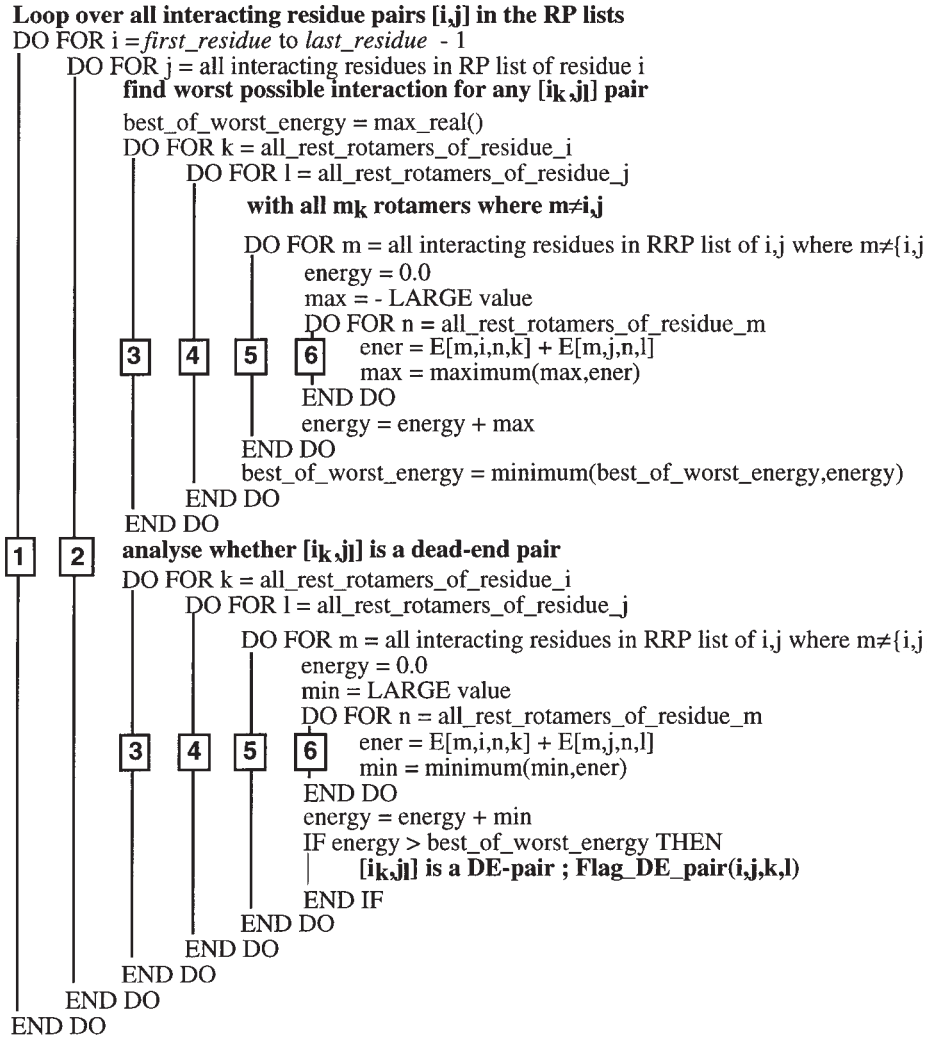


Fig. 7. General flow in pseudolanguage representation of the Classic form of the dead-end pair computation. Some of the programmatorial details are not shown in order not to overload the charts.

tation of the search for all-DEPs cases implies an exit condition from the nested-loop structure that is required to examine all possible pairs. Only, after exhausting the all-DEPs cases can a full computation of DEPs be done. At this moment, a large amount of rotamers have already disappeared from the system, and consequently the computational requirements to determine the DEPs become strongly reduced.

3. Optimizing the Code

3.1. The Rotamer Library

The current implementation of the DEE method uses a collection of side-chain orientations for each of the rotatable residues. It is clear that sufficient rotameric states have to be included in this library, allowing a correct prediction of the experimentally determined protein structure. Contrary to intuition, we have proven that it is exactly this enlargement of the rotamer library that allowed an important reduction in the number of rotamers at each position of the protein ***(6)***. If one uses a smaller library (being a subset of the large library), not only are the results inferior, but the time required to eliminate all dead-ending rotamers becomes larger.

3.2. List of Pairs (LP)

3.2.1. LP for Atom/Atom Interactions in Energy Calculations

As described in the previous sections a pairwize atom interaction energy term is used to calculate the energy of the protein. It is a well known and safe practice in energy minimizations and molecular dynamics to take only interacting atoms lying in a sphere around each atom. The cutoff radius in this work has been set to 8 Å. As a consequence, the initial calculation of the interaction energies of each rotamer with the template grows roughly linearly with the number of residues in the protein. The calculation of these list of pairs in the cutoff sphere is greatly accelerated by using a optimized cubing algorithm.

3.2.2. LP for Rotamer/Rotamer Interactions in DEE Calculations

A similar situation exists for the calculation of rotamer/rotamer interactions in the DEE calculations. Because the DO loops in the calculations are residue driven, and to keep memory requirements within bounds, we use residue pair lists instead of rotamer pair lists. After the initialization phase, a list of interacting residue pairs is set up based on the interaction energy of the two considered residues, more precisely, if one rotamer pair of the two residues has a nonzero interaction energy, the residue pair is counted. This list is referred to as the residue pairs list (RP list). This reduces in a dramatic way the number of iterations in the DO loops when searching for dead-ending rotamers. Whenever a residue becomes fixed, the RP list is updated.

The concept of RP lists can also be applied to the DEE of the rotamer pairs, explained in **Eq. 6** for the classic rotamer pairs and **Eq. 7** for the Goldstein rotamer pair representation. For easier reading we refer to this as the residue rotamer-pair list (RRP list). This list is built starting from the RP list. Here, we even gain more computation time, as larger chunks of the eight-level-deep DO loop are eliminated. The general flow of the search for dead-ending pairs is

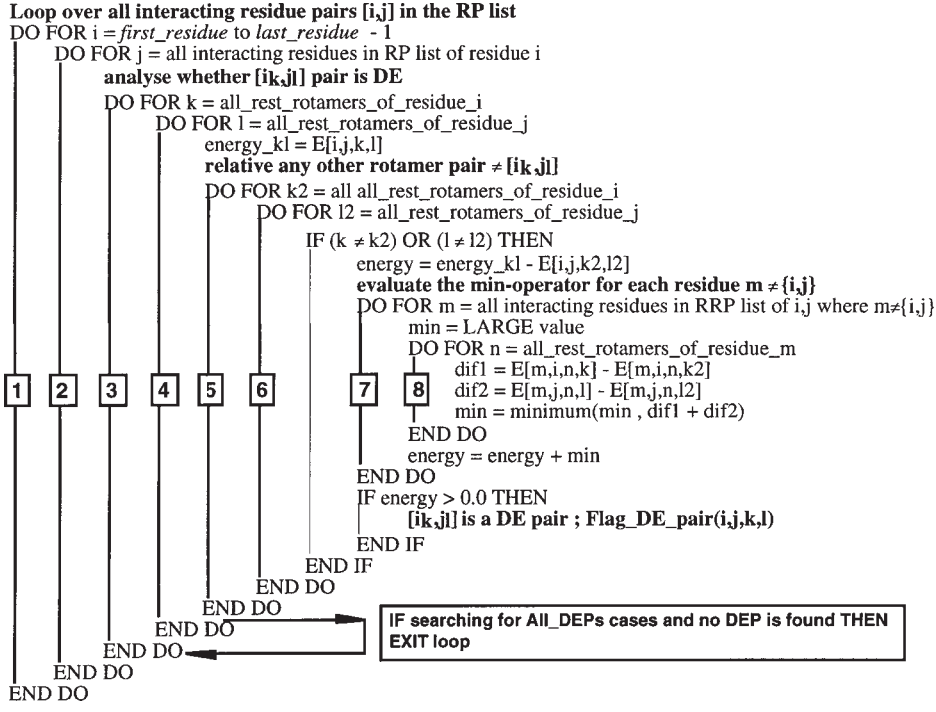


Fig. 8. General flow in pseudolanguage representation of the generalized form of the DE pair computation. Some of the programmatorial details are not shown in order not to overload the charts.

found in **Figs. 7** and **8**, respectively, for the classic and Goldstein implementation of DEE rotamer pairs. How these phases interact with the general flow of the program is shown in **Fig. 9**.

3.3. High Energy Threshold Reduction Values (HETR)

3.3.1. HETR for a Rotamer Versus Template

In the initial stage of the program, all side-chain orientations are generated based on the χ values defined in the rotamer library. Although already from the start a lot of rotamer conformations may be eliminated from this collection merely based on a clashing criterion with the template, still a large number of conformations exist at each position. Different strategies have been followed also in literature to avoid the combinatorial explosion that all modelers face when searching the combination of rotamers that shows the least energy. Some authors *(23)* have even challenged the existence of a combinatorial barrier in

```
Initialisation_phase
    Read pdb protein structure
    Define residues to be modeled
    Read in the rotamer library
    Read the overlap matrix
    Expand rotamer library steps on χ1 and χ2 for aromatic residues
    Create at each position the side-chain conformation with rotamer library
    Calculate for each side-chain rotamer the template interaction energy E(i_m)
    Remove all template incompatible side-chain conformations
    Sort all rotamer conformations by energy and reject those with E(i_m) > E_min(i) + HETR
    Flag all rotamer/rotamer conformations as DE pair if E(i_kjn) > E_min(i,j) + HETR_pair
DO TWICE
|   DO UNTIL no more DE rotamers are found
|   |   DO UNTIL no more DE rotamers are found
|   |   |   DO UNTIL no more DE rotamers are found
|   |   |   |   DEE Goldstein DEE for single residues
|   |   |   |   Fix residues with only one rotamer left
|   |   |   |   Reconstruct the RP lists
|   |   |   END DO
|   |   |   DO UNTIL no more DE rotamers are found
|   |   |   |   DEE Goldstein DEE with optimized coefficients for single residues
|   |   |   |   Fix residues with only one rotamer left
|   |   |   |   Reconstruct the RP lists
|   |   |   END DO
|   |   |   Apply Classic DEE for rotamer pairs
|   |   |   IF all DEPS cases found THEN
|   |   |   |   Fix residues with only one rotamer left
|   |   |   |   Reconstruct the RPP lists
|   |   |   END IF
|   |   END DO
|   |   Apply Goldstein DEE for rotamer pairs
|   |   IF all DEPS cases found THEN
|   |   |   Fix residues with only one rotamer left
|   |   |   Reconstruct the RPP lists
|   |   END IF
|   END DO
|   DO ONCE
|   |   Search at each position rotamers with > 90% overlap
|   |   Replace with rotamer conformation with best interaction energy
|   |   Fix residues with only one rotamer left
|   |   Reconstruct the RP lists
|   END DO
END DO

End phase of the program
    Divide and Conquer strategy (DAC)
    DEE assisted by local modeling
    Combinatorial build up, assisted by DEE method

Save GMEC structure

Energy minimisation on side chains
```

Fig. 9. General flow in pseudolanguage representation of the DEE implementation in the Brugel package.

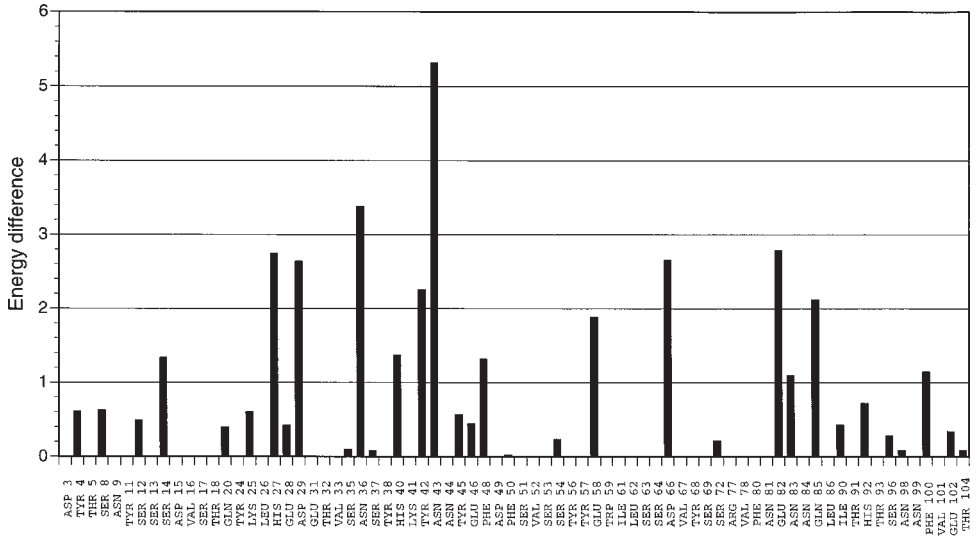


Fig. 10. Difference between the template interaction energy of the rotamers i_m and i_g for the rotatable side chains of ribonuclease T1 (PDB code 1rpga) for all rotatable residues, using the "large library" of 859 rotamer elements. The term i_m denotes the rotamer with the least interaction energy versus the template, whereas i_g is the GMEC energy of the considered residue.

side-chain placement. These authors argue that each side-chain can be modeled taking into account only the environment of the template's backbone atoms. However, this opinion has been modified by the work of Tanimura et al. ***(5)***, who have clearly shown that side-chain–main-chain and side-chain–side-chain interactions work concurrently to stabilize the protein structure. That the GMEC rotamer i_g often does not coincide with the rotamer i_m of least template interaction energy is demonstrated in **Fig. 10**. This figure shows the difference in interaction energy with the template for the i_g and i_m rotamers. The template is defined as all main-chain atoms; all C_β atoms; all proline, glycine, alanine, and disulphide-bonded cysteine residues; and possibly nonmodeled residues.

In order to reduce the number of rotamer elements at each position and without risking the elimination of the i_g conformer, we have studied the $E(i_g) - E(i_m)$ energy difference for a series of proteins ***(6)*** using a very detailed rotamer library of 859 elements. It is observed that, in general, between 40–60% of all rotamers, the i_g does not coincide with the i_m. On the other hand, it is also observed that the $E(i_g) - E(i_m)$ energy difference is relatively small. Based on these experimentally determined energy differences, it was possible to determine a high-energy threshold reduction (HETR) value of 10 kcal mol^{-1}. As a

Table 5
Overview of the 22 Studied Proteins

PDB code	Protein name	# res[a]	# rbl[b]	Resolution[c]	Hetero groups	Ref.
1crn	Crambin	46	26	1.0	PO4 + K	***60***
4rxn	Rubredoxin	54	39	1.2	Fe (II)	***61***
2ovo	Ovomucoid	56	39	1.5	—	***62***
5pti	BPTI	57	36	1.0	PO4 + K	***63***
1igd	Protein G immuno gl binding	61	49	1.1	—	***64***
3ebx	Erabutoxin	62	45	1.4	SO4	***65***
2sn3	Scorpion neurotoxin	65	41	1.2	2methyl24pentadiol	***66***
1hoe	α-Amylase inhibitor	74	53	2.0	—	***67***
1ubq	Ubiquitin	76	65	1.8	—	***68***
351c	Cytochrome C	82	54	1.6	Heme	***69***
1rga	Ribonuclease T1	104	77	1.7	Ca + GMP	***70***
256b	Cytochrome B562	106	82	1.4	Heme	***71***
4bp2	Pro-phospholipase A2	115	85	1.6	Ca + met-pentadiol	***72***
7rsa	Ribonuclease A	124	97	1.2	met-propanol	***73***
1rpg	Ribonuclease A in complex	124	97	1.4	met-pentadiol + CPA	***74***
2aza	Azurin	129	98	1.8	Cu + SO4	***75***
1lz1	Lysozyme	130	95	1.5	—	***76***
1mba	Myoglobin	147	100	1.6	Heme	***77***
9wga	Wheat-germ agglutinin	170	82	1.8	—	***78***
2ptc	B-trypsin + BPTI	281	200	1.9	Ca	***79***
3app	Penicillopepsin	323	245	1.8	—	***80***
2apr	Acid proteinase	325	239	1.8	Ca	***81***

[a]Total number of residues in the protein.
[b]Number of rotatable residues.
[c]Resolution is given in angstroms.

consequence, all rotamers in a window of 10 kcal mol^{-1} above the i_m rotamer have to be taken as valid solutions in the DEE computations, whereas the others can safely be eliminated. On average, for the studied proteins this step leads to a reduction of ±30% of the number of possible rotameric states before starting the DEE calculations. In total, the whole initialization step removes an average of between 50–70% of the total possible number of rotameric states.

Such an HETR value could not be determined when working with a subset of the large library. Indeed, in view of the coarseness of this smaller library, a larger HETR value has to be defined. As a consequence, the DEE calculations

start with more rotamer states compared to using the large library. In addition to the longer execution time, the quality of the prediction is also inferior.

3.3.2. HETR for Rotamer/Rotamer Interactions

Another important step in the initialization phase of the DEE implementation is the calculation of the pairwise rotamer interaction energies. First, all rotamers that are incompatible with all rotamers of another residue are dead ending and removed from the list of rotamers. It is also possible to define an HETR pair value, similar to the HETR criterion. By examining all rotamer pair interactions for the list of proteins in **Table 5**, the HETR pair value has been determined to 20 kcal mol^{-1}. All rotamer pairs of the residues *i,j* having an interaction energy that fall outside the window of 20 kcal mol^{-1} above the minimum $E(i_k,j_l)$ pairwise energy are flagged as DEPs. It is important to note that they are not removed from the list of active rotamers but can indirectly lead to the discovery of DE rotamers, either in the all DEPs cases, the logical pairs theorem, or in the next dead-end cycle.

3.4. Removal of Highly Overlapping Rotamers After First Complete DEE Cycle

For each type of library we have once calculated the complete matrix of the rotamer/rotamer overlap expressed in percentage volume overlap. The term "volume overlap" is explained in **Subheading 4**. This information is stored in a file and may be read if required. After one complete cycle of the DEE program it is possible to reduce the remaining pool of rotamers solely on this overlap criterion. The user may provide the program with an overlap percentage cutoff (here 90%), above which the overlapping rotamers are replaced by the rotamer having the best interaction energy with the template. The template is here defined as the backbone, all: Gly-, Ala-, Pro-, and Cys-bridged residues and all the — so far — fixed residues. If by this procedure there are residues that become uniquely defined, they are fixed. With this reduced set of rotamers the DEE cycle is executed once more. It is important to note that since the system has changed, all flagged DE rotamer pairs have to be recalculated.

4. Flow of the Program

4.1. The Modeling Package

The DEE algorithm has been implemented in the Brugel package ***(24)***. Although mainly written in FORTRAN 77, advanced memory management is incorporated, allowing the use of dynamic memory allocation and pointer-based record structures. This facilitates the ease of programming and does not require ad hoc buffer size allocation. Besides a graphical menu-driven user

interface to visually check the results of a computation, the Brugel package also contains a command line user interface, allowing the execution of a series of operations in batch form. One of the main characteristics of this modeling package is the object-oriented user interface. Objects are defined in a very broad sense. They may contain objects related to the protein like collections of atoms, residues, chains, tables of one- or three-dimensional values associated with each residue or atom, i.e., accessible surface area (ASA), but also nonatom-related objects like lists of strings, integers, and float values. Commands take input and output objects, allowing the user to manipulate in a logical way the complete description of the studied protein. The basic set of commands allows the creation of objects, another set allows us to logically manipulate these objects by creating new objects. To facilitate the manipulation of objects, the commands may be used as inline functions. The input for these functions is a list of BRUGEL objects in Reverse Polish notation (RPN). Of course, strong object type checking prevents erroneous results and facilitates debugging complex procedures. On top of this command line interface, the user is allowed to build his or her own supercommands (called procedures) taking advantage of the object oriented grammatical language. These procedures may take any number of arguments (being objects as specified) and form the basis of a computer language on itself. All major flow control constructions like "Do loops" and "If Then Elseif Endif" are implemented in this procedural language. The advantage of this method is that the procedure can be build with the help of any type of text editor, creating a file with a sequence of commands and procedures to be executed and tested beforehand. This procedures are collected in procedure libraries and do not require the recompilation and relinking of new FORTRAN or C/C + + code. Of course, in case a completely new algorithm is developed or one wants to accelerate a prototyped procedure, a user interface is provided to link the new code with the existing package.

The energy function used ***(25)*** includes the usual terms for bond stretching, bond–angle bending, a periodic function for the torsion angles, a Lennard–Jones potential for the nonbonded atom pairs, a 10–12 potential for hydrogen bonds, and a coulombic function for charged atoms. The dielectric constant has been set to r_{ij}, the distance between the atoms i and j ***(26)***. The energy parameters are based on the CHARMM force field and are used throughout this work ***(27)***. Other groups have already implemented, with success, the DEE method using their own force field ***(5,11,20,28)***. Although it is possible to use a united-atom representation, usually all atoms including all hydrogens are used in the energy calculation. In this chapter, carboxylate and imidazole groups were not protonated. In order to understand better the current implementation of the DE elimination, we have included a schematic overview of the flow of the program in **Fig. 9**.

4.2. Program Initialization

The package takes as input a PDB-formatted ***(29)*** file, the rotamer library, and the residues to be modeled. This may be the complete set of all rotatable residues or a subset (buried residues, a hydrophobic cluster, or an interface with another protein). The user is allowed to define a number of steps with a well-defined angle size to expand the rotamer library. In this and previous studies we expanded the library by taking two steps of 10° around the $\chi 1$ angle of the aromatics (Phe, Tyr, His, Trp), and for each of these new rotamers we took two steps of 20° around the $\chi 2$ angle. This enlarges the rotamer library to 859 elements. Next, with the dihedral angles defined in the library, all rotamer conformations (x-, y-, and z-coordinates of the side chain only) at each position are generated and stored in memory, avoiding recalculation of the side-chain coordinates in later steps. In a final initialization step, the file containing the matrix with the rotamer/rotamer volume overlap percentages is read in.

4.3. Elimination of Template and Sidechain Incompatible Rotamers

In this phase of the program two types of energies are calculated. First, the inherent energy, $E(i_k)$ being the sum of the rotamer self-energy and the interaction energy with the template is calculated. The template is defined as the backbone and all Gly-, Ala-, Pro-, and Cys-bridged residues. From this global rotamer pool all rotamers having an interaction energy with the template exceeding the absolute value of 30 kcal mol^{-1} are eliminated. Next, at each residue position, the rotamer interaction energies with the template are sorted and all rotamers removed from the list having an energy larger than the window of 10 kcal mol^{-1} (HETR) above the lowest value.

Second, for the remaining set of rotamers, all pairwise interaction energies $E(i_k,j_l)$ are calculated. A rotamer of a given residue having a interaction energy greater than 30 kcal mol^{-1} in absolute value with all other rotamers of another residue is removed. Finally, the application of the HETR pair criterion flags all those rotamer pairs that are dead-ending.

4.4. The DEE Phase

The previous preparative steps lead the way to eliminate in a reasonably fast timeframe all dead-ending rotamers, because so many useless rotamers are eliminated from the total pool of rotamers. Nevertheless, for large proteins, the number of rotamer combinations is still enormous. It is of the greatest importance to execute first the less complex equations before proceeding to the more complex. Also, the computational requirements for the assessment of DEPs are

much more exigent than those for single rotamers. Both **Eqs. 6** and **7** allow the identification of DEPs. Although **Eq. 7** is a much stronger criterion than **Eq. 6**, we nevertheless do not relinquish the use of the original DE criterion. In the first place, **Eq. 6** appears to be, in practice, very effective in retrieving additional single dead-end rotamers. In the second place, this criterion can be evaluated much faster as compared to the modified criterion (**Eq. 7**). The rationale behind this is that the scanning for all possible DEPs evidently requires a loop over all possible rotamer pairs $[i_r,j_s]$. The decision whether each such pair is a DEP relative some other rotamer pair requires an extra loop running over all alternative pairs. This second loop is nested inside the first loop in case of **Eq. 7** since the comparison between the involved energies is contained within the min operators of this criterion (*see* **Fig. 8**). With regard to **Eq. 6**, these loops do not have to be interlaced. Indeed, as shown in **Fig. 7**, one can first compute the worst possible interaction energy (right-hand terms in **Eq. 6**) for each possible rotamer pair $[i_k,j_l]$. The lowest energy value (the so-called *best-of-worst* energy) is memorized ***(4,30)***. Subsequently, any rotamer pair that has a minimum interaction energy (left-hand terms in **Eq. 6**) that is higher than the *best-of-worst* energy is bound to be a DEP. Consequently, **Eq. 7** is to be executed after exhausting **Eq. 6**, at which moment the size of the rotamer system is already strongly reduced.

The present flow of dead-end elimination is as follows.

1. Iterative DE elimination of rotamers until exhaustion using the modified criterion (**Eq. 15**) taking into account previously determined DEPs, if any. Subsequently, we search iteratively until exhaustion for DE rotamers using **Eq. 11** with optimized weight coefficients.
2. Computation of DEPs using **Eq. 6** as outlined in **Fig. 7**.
3. Computation of all-DEPs cases using **Eq. 7**.
4. Full exploration of **Eq. 7** as outlined in **Fig. 8**.

It is understood that as long as new DEPs are found in each of phases 2–4, we iteratively search for new, single, dead-ending rotamers using the procedure described in **step 1**.

4.5. Elimination of Redundant Rotamers

At this stage of the program, many rotamers are eliminated and, in some cases, the GMEC conformation is reached, in which case the final structure is saved. In case the structure is not yet uniquely defined, a one-time rotamer pruning based on the volume overlap is used. By this elimination, rotamers of the same residue with a volume overlap exceeding 90% are replaced by the rotamer of this subgroup having the best interaction with the so far fixed template. Only After this elimination step, the system is reinitialized and, with the

remaining rotamers, again injected in the DEE cycle searching for new dead-ending rotamers.

4.6. End Phase Routines

The previously described set of DEE and optimizations are very powerful, but nevertheless it is possible that a too large number of rotamers is still present to be tackled by brute force combinatorial techniques. We have developed two additional techniques to solve this problem. These methods are outlined in **Subheadings 4.6.1.** and **4.6.2**.

4.6.1. Divide and Conquer Routines (DAC)

The remaining set of residues is divided into subsets A and B. Although the rotamers of set B are kept as single rotamers, those of subset A are combinatorially unified into superrotamers $\mathfrak{R}$. Subsequently one by one all $\mathfrak{R}$ are temporarily fixed by considering them as part of the template. Each of the fixed superrotamers may lead to a list of dead-ending rotamers in the B set. Each time this list is memorized. The intersecting rotamers of all these lists resulting from fixing the superrotamers are dead-ending rotamers and may be removed from the list of left rotamers (proof and effectiveness is illustrated in **ref. *31***). It is clear that given the combinatorial nature of this superrotamer, combining more than two residues into one superrotamer is practically impossible.

4.6.2. Combinatorial Buildup Assisted by DEE

This combinatorial routine generates all possible side-chain combinations for a growing cluster of residues, starting from the root residue being the one with the least rotamers left. Instead of simply exploring the full combinatorial tree, it is attempted to predict the conformation for the remaining rotatable residues at each specific rotamer combination for the current residue cluster. A fast and reliable way is using the conventional DEE routines for single residues. The result of such an attempt is either that all remaining residues have only one rotamer left or not. The remaining residues become uniquely defined the algorithm passes to the next cluster node, whereas in the latter case the current cluster is enlarged with one more residue for combinatorial enumeration. Selecting the next residue to be added is done by searching the residue with the highest *interaction number*. This quantity is defined as the number of nonzero interactions with the already-clustered residues divided by the number of rotamers of the residue. By recursively calling this routine, the tree is completely explored. When all combinations of the cluster have been explored, the algorithm terminates. The GMEC conformation is identified as the cluster element that yielded the lowest energy of the total protein.

4.7. Storage of the GMEC Structure

After execution of the DEE, and the final phase of the program, all rotatable residues have only one rotamer left and the final GMEC structure is saved in Brookhaven PDB format. This structure might contain small, short contacts due to the discretization of the rotamer library. To alleviate these repulsions, the structure is subjected to 100 steps of steepest descent energy minimization. In this step, the backbone is kept fixed.

4.8. Automation of the Method

The current implementation of the DEE package is completely automated. The program takes the PDB-formatted file as input. It is possible to include water molecules and hetero atoms, although in this study they were stripped off. Missing hydrogen atoms were generated. In this study the effect of including the heme group was tested. Disulphides, prolines, glycines, and alanines are kept unchanged and form the template structure, together with the main-chain backbone and the C_β atoms. The user defines the objects, being the residues he or she wants to model, the used rotamer library, and steps to be taken around the χ angles of user-selected residue types. The next phase starts the actual DEE, removal of redundant rotamers and the final clustering algorithm. At users' request, a detailed output is provided during each rotamer elimination step. Finally, the resulting structure is subjected to 100 steps steepest descent energy minimization with fixed main-chain atoms in order to alleviate minor short contacts. Thanks to the procedural language of the modeling package, all these steps can be edited off line and a single input file submitted to the Brugel package for batch execution.

Finally, the original X-ray structure and the modeled structure are evaluated in terms of volume overlap, difference in χ angles, root-mean-square deviation (RMSD), and scoring quality of the side chains. A typical output is given in **Figs. 11** and **12**, and explained in **Subheading 5.**

5. Evaluation of the Method

5.1. The Protein Test Set

In the current work we used a representative test set of 22 proteins retrieved from the PDB *(29)*. The proteins are listed in **Table 5**, together with the total number of residues in the protein, the number of rotatables, the resolution, the presence of hetero groups, and the reference to the structure.

5.2. Use of the Volume Overlap Criterion Instead of RMSD or Dc

In literature a wide variety of criteria is used to validate the correctness of the side-chain prediction. In general, the accuracy of the prediction is evalu-

```
A1 LYS % max asa: Cryst=  33.4 Msa=  35.9  Phi= -      Psi= 148.5
crystal chis:  177.0   179.4   167.7   167.7
closest rotl: -172.1   175.3   180.0   180.0 rms= 0.45
msa     chis: -172.1   175.3  -180.0   180.0 rms= 0.45 %overlap= 84.7
msa+rs  chis: -173.5   176.9  -180.0  -179.6 rms= 0.40 %overlap= 85.4
========================================================================
A2 GLU % max asa: Cryst=  45.3 Msa=  46.8  Phi=-135.1  Psi= 153.1
crystal chis:   66.5  -175.7   -84.9
closest rotl:   69.8  -179.0   120.0          rms= 0.36
msa     chis:   69.8  -179.0   120.0          rms= 0.36 %overlap= 87.4
msa+rs  chis:   69.0  -179.5   113.5          rms= 0.27 %overlap= 91.3
========================================================================
A4 TYR % max asa: Cryst=  17.8 Msa=  12.1  Phi= -80.5  Psi= 129.8
crystal chis:  -66.4   103.9
closest rotl:  -66.5    96.6                 rms= 0.10
msa     chis:  -76.5   116.6                 rms= 0.57 %overlap= 81.6
msa+rs  chis:  -75.5   115.0                 rms= 0.51 %overlap= 84.0
========================================================================
A5 LEU % max asa: Cryst=   0.0 Msa=   0.0  Phi= -62.1  Psi= 150.1
crystal chis:  -63.5   175.1   178.9  -178.1
closest rotl:  -64.9   176.0    60.0    60.0 rms= 0.04
msa     chis:  -64.9   176.0    60.0    60.0 rms= 0.04 %overlap= 98.8
msa+rs  chis:  -64.3   175.7    59.6    61.8 rms= 0.03 %overlap= 99.7
========================================================================
A6 VAL % max asa: Cryst=   1.3 Msa=   0.0  Phi=-137.2  Psi= 140.4
crystal chis:   62.9  -178.9   178.6
closest rotl:   69.3    60.0    60.0          rms= 0.12
msa     chis:   69.3    60.0    60.0          rms= 0.12 %overlap= 92.2
msa+rs  chis:   67.2    62.9    55.3          rms= 0.08 %overlap= 94.3
========================================================================
A7 LYS % max asa: Cryst=  29.8 Msa=  29.7  Phi= -83.9  Psi= 131.2
crystal chis:  -61.2   176.9  -128.5   166.9
closest rotl:  -68.9  -178.4   -60.0   180.0 rms= 0.90
msa     chis:  -68.9  -178.4  -180.0    60.0 rms= 1.37 %overlap= 68.3
msa+rs  chis:  -69.1  -178.2  -180.0    60.7 rms= 1.38 %overlap= 69.1
========================================================================
A8 LYS % max asa: Cryst=  74.8 Msa=  45.1  Phi= -67.3  Psi= -25.4
crystal chis: -168.3   166.6    61.2   172.0
closest rotl: -172.1   175.3    60.0   180.0 rms= 0.33
msa     chis: -104.0    74.6    60.0   180.0 rms= 3.64 %overlap= 18.8
msa+rs  chis:  -98.1    66.1    64.2   177.5 rms= 3.68 %overlap= 18.2
========================================================================
A9 SER % max asa: Cryst=  71.5 Msa=  70.6  Phi= -70.4  Psi= -53.1
crystal chis:   58.8  -179.8
closest rotl:   64.7    60.0                 rms= 0.09
msa     chis:   64.7    60.0                 rms= 0.09 %overlap= 92.7
msa+rs  chis:   61.6    69.3                 rms= 0.07 %overlap= 94.9
========================================================================
A10 ASP % max asa: Cryst=  31.3 Msa=  34.3  Phi=-116.0  Psi= -11.3
crystal chis:   61.9    10.5
closest rotl:   63.7     0.0                 rms= 0.14
msa     chis:   63.7     0.0                 rms= 0.14 %overlap= 94.0
msa+rs  chis:   62.2     5.8                 rms= 0.08 %overlap= 95.8
```

Fig. 11. Typical analysis results for the starting residues of the protein Scorpion neurotoxin (PDB code 2sn3) of a side-chain placement experiment. The structure before and after the steepest descent minimization is compared with the X-ray structure. The first line lists the considered residue in the protein, for the crystal and the modeled structure the percentage of buried surface area as compared with the maximal possible ASA of the residue type in extended form together with the φ and ψ angles. The second line lists the experimentally determined X-ray χ angles. The third line lists the library rotamer closest to the experimental value with the χ angles and the RMSD as compared with the X-ray structure. The fourth and fifth line list, respectively, the χ angles as observed in the modeled structure before and after the energy minimization step. In addition, the RMSD value and percentage volume overlap versus the X-ray structure are added.

```
Nonbonded ener: Ori=   -646.9
                DEE=   -689.7
Rms 10 % Max ASA side chains heavy atoms =   0.69
Rms 25 % Max ASA side chains heavy atoms =   0.68
Rms all side chains heavy atoms =    1.06
Overlap 10 % Max ASA side chains heavy atoms =    89.7
Overlap 25 % Max ASA side chains heavy atoms =    89.8
Overlap all side chains heavy=    88.0
```

res	#<10%A	wrong	%	#<25%A	wrong	%	#tot	wrong	%
arg	1.	0.	100.0	1.	0.	100.0	1.	0.	100.0
lys	-	-	-	-	-	-	2.	0.	100.0
asp	1.	1.	0.0	1.	1.	0.0	6.	2.	66.7
glu	1.	0.	100.0	1.	0.	100.0	6.	1.	87.3
his	-	-	-	2.	0.	100.0	3.	0.	100.0
phe	3.	0.	100.0	4.	0.	100.0	4.	0.	100.0
tyr	6.	0.	100.0	8.	0.	100.0	9.	0.	100.0
trp	1.	0.	100.0	1.	0.	100.0	1.	0.	100.0
asn	1.	0.	100.0	3.	0.	100.0	9.	2.	77.8
gln	1.	0.	100.0	1.	0.	100.0	2.	0.	100.0
ser	-	-	-	-	-	-	15.	9.	60.0
thr	1.	0.	100.0	1.	0.	100.0	6.	0.	100.0
ile	2.	0.	100.0	2.	0.	100.0	2.	0.	100.0
leu	-	-	-	2.	0.	100.0	3.	0.	100.0
val	4.	1.	75.0	6.	1.	83.3	8.	2.	75.0
met	-	-	-	-	-	-	-	-	-
all	22.	2.	90.9	33.	2.	93.9	77.	16.	79.2

Fig. 12. Typical score analysis output of the Brugel package for the protein with PDB code 1rga. The first two lines list the nonbonded energy for the X-ray structure as compared to the DEE side chain placement program. The three following lines list the RMS. difference between the X-ray and the modeled structure for less than 10% and 25% solvent accessibility, respectively, and the RMSD for all side chains. The total volume overlap for the same three accessibility classes is given in the next three lines. The next table lists for each possible residue type, and for the three defined accessibility classes, the number of modeled residues, the number of wrongly predicted residues, and the percentage of correctly modeled residues in each ASA class. The final line summarizes the information for all residues present in the modeled protein.

ated for two classes of residues — the solvent exposed and buried side chains. As seen from the literature the definition of buried residue is rather vague. Some groups use the slightly extended definition of Miller et al. ***(32)***. This definition states that buried residues have less than 10% of their maximal

accessible solvent area exposed. Others ***(5,23,33–39)*** consider a more permissive threshold of 20–40% accessible surface area exposed to the solvent. There are mainly two methods used for the evaluation of the correctness of the modeled structure. The first — and most used — method is the RMSD between the modeled and the experimental X-ray structure. The second method is the comparison of the side-chain dihedral angles between the calculated and the X-ray observed values. There is a considerable variation in the definition of the tolerance on $\chi 1$, $\chi 2$ separately or on the combined $\chi 1 + \chi 2$ used in the evaluation process (from 20–40°).We have discussed recently ***(6)*** the difficulties encountered when using either the RMSD and $\Delta\chi$ method. In order to evitate these problems, a method was needed to evaluate the correctness of modeled side chains that is sensitive to spatial errors or functional interactions, but insensitive to alternate fitting of the electron density map. The real space fit for a side-chain evaluates how well the calculated electron density map of the model coincides with the observed electron density map. Similar to the real space fit ***(40)*** and also inspired by the work of Schiffer et al. ***(41)*** we propose to use the Van der Waals volume overlap to evaluate how well the volume of the predicted side-chain volume overlaps with the observed X-ray structure. In a similar fashion, as the real space fit, the calculated volume overlap becomes independent of individual χ angle comparisons and evaluates the side chain as a whole entity. A systematic analysis of a series of proteins allowed us to define the percentage threshold when a modeled side chain may be considered as correctly positioned ***(6)***. This value was determined to be 70% volume overlap between the modeled and the X-ray structure.

5.3. Calculation of the Volume Overlap

Provided the experimental structure is present, we are able to calculate the side chain volume overlap of the calculated and experimental structure. To this end, an algorithm was implemented allowing the computation of the volume overlap between any pair of objects. Objects are defined as any user-defined atom collection of the protein. The computation of the volume overlap between the observed (X-ray) and predicted side-chain conformation goes as follows and is applied to each predicted residue. In the first place, two objects are made containing the side-chain atoms from the X-ray and the modeled structures. Second, we generate a cubic lattice with a 0.5 Å mesh size encompassing the Van der Waals envelopes of each objects. A logical AND operation between the two objects selects the overlapping points of the two objects. The overlapping volume is then obtained by multiplying the number of points with the volume of the lattice unit cube. To optimize these calculations, we orient the cubic lattice along the principal axes of the object containing the atoms of the X-ray observed side chain. As a consequence, lesser grid points are needed to encom-

pass fully the Van der Waals envelopes of the side chains thereby increasing the computational speed. **Figure 11** shows a typical output of the analysis phase for the scorpion neurotoxin protein (PDB code 2sn3).

5.4. Scoring Analysis

It is a well-known fact that solvent exposed residues are less well predicted as compared to buried residues. We have shown ***(6)*** that extending the definition of Miller et al. ***(32)*** for buried (core) residues from less than 10% of the maximal possible accessible surface area to 25% makes only a marginal difference in our prediction quality. In addition most other investigators use this extended view too, facilitating comparison between different methods. The maximal ASA for each type of amino acid is calculated using the Survol algorithm ***(42)***, being part of the Brugel package ***(24)***, on the basis of extended dipeptide units built as terminally blocked amino acids Acetyl-X-NHCH3 ***(1)***. A water probe radius of 1.4 Å. was used throughout all these calculations. A high overall volume overlap of around 90% between the modeled and X-ray structure is observed for the core residues. In case the PDB file indicates the existence of multiple conformations for a residue, the alternative with the highest occupancy factor is selected for comparison with the modeled side chain. An example of the typical Brugel output produced as scoring analysis is shown in **Fig. 12**.

6. Performance

The current DEE implementation in the Brugel package has outperformed the first DEE implementation by several orders of magnitude the computational needs for the prediction of the side-chain orientation in a fixed backbone. At the same time, the accuracy of the rotamer library increased significantly the quality of the prediction ***(6)***. Application of the simple HETR criterion induced a tenfold execution time reduction for medium-sized proteins of around 120 residues. Subsequent introduction of the HETR for rotamer pairs and the usage of residue rotamer pairs lists induced a time gain between 3 for small proteins (50 residues) to a factor of 22 for large proteins (3app with 325 residues). In addition to this increase in accuracy and decrease in computational time, the current optimizations decreased also significantly the memory requirements of the modeling package.

The execution time for each major step in the side-chain placement program is depicted in **Fig. 13**. The initialization time grows almost linearly with the number of residues in the protein (this correlation plot is not shown). As we observe from **Fig. 14**, the total execution time is roughly linearly dependent with the total number of residues in the studied protein. A few exceptions to this rule are also observed and are due to the exceptionally large number of

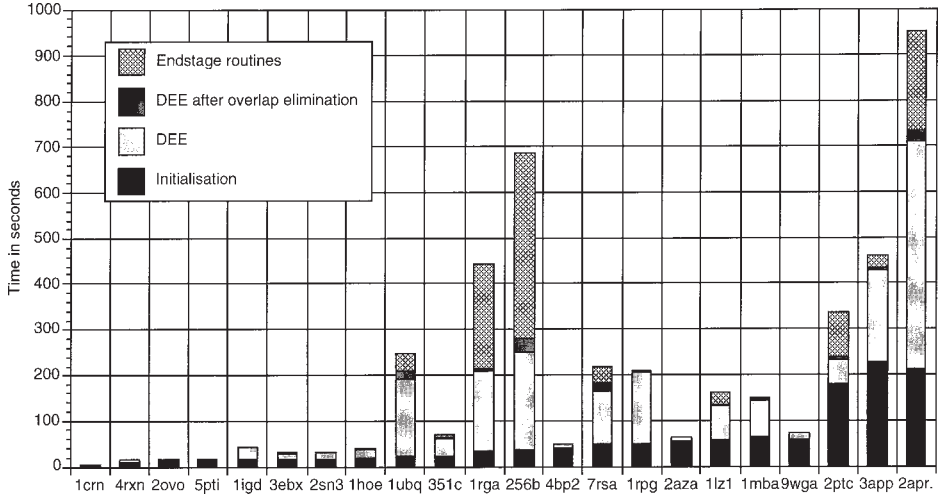


Fig. 13. Plot of the contribution of each phase in the total execution time of the DEE program for each of the studied proteins. The different gray scales indicate, respectively, the initialization time, the DEE phase, the reapplication of the DEE algorithm after the elimination of highly overlapping rotamers, and finally the execution time of the end-stage routines.

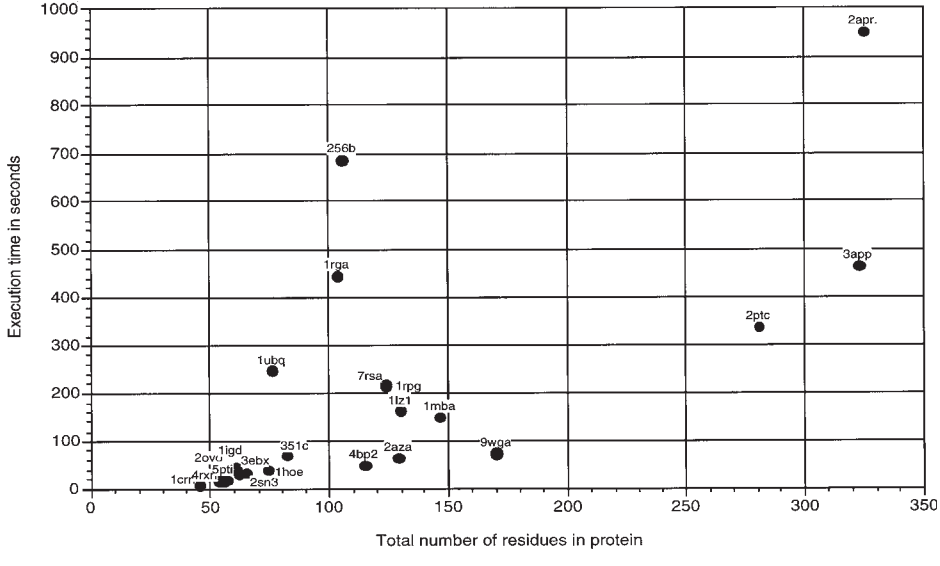


Fig. 14. Dependency of the total execution time versus the total number of residues in each of the 22 studied proteins. Each of these proteins carries its PDB code.

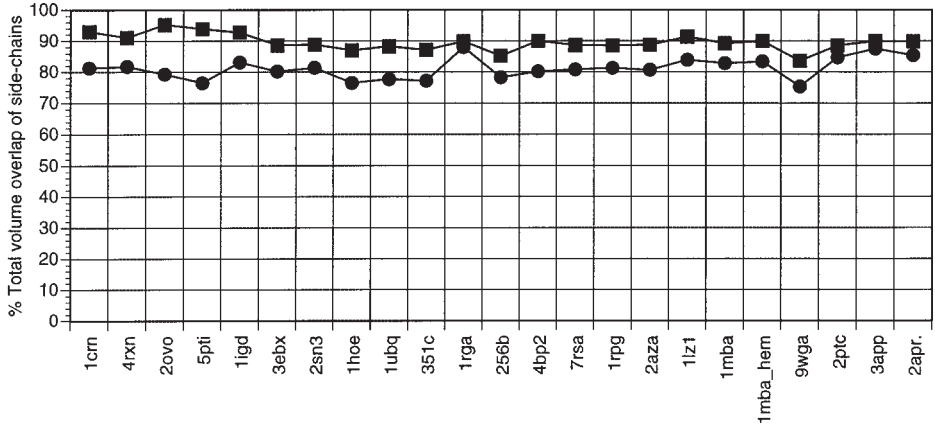


Fig. 15. Plot of the percentage overlap of all modeled side chains for each of the studied proteins. The filled square symbols indicate residues with a solvent accessibility of less than 25% of their maximum ASA value, whereas the filled circles indicate the results for all the side chains in the protein.

rotamers left before starting the end-stage routines. The exclusion of the heme group in cytochrome B562 (structure 256b) in the calculation could partially explain the longer end-stage phase. Inclusion of hetero groups in myoglobin (structure 1mba) indeed reduces the conformational space and accelerates the elimination of dead-ending rotamers. A similar effect is observed for the protein wheat germ agglutinin (PDB code 9wga, only one chain of the dimer has been modeled). One monomer of the dimeric form contains 170 residues, but only 82 rotatable residues. The unusual high Gly contents (40 residues or 23.4% of the total number of residues) and Cysteine bridges (32 residues or 18.7% of the total number of residues) makes the conformational space highly constrained by the absence of rotatable side chains. This is observed in the very effective execution of the DEE algorithm. The main reason is that by modeling in a constrained environment, the interacting rotamers are fixed in an early stage of the DEE process, thereby increasing the elimination power of the DEE and FEE processes. The drawback of this speed increase is a somewhat less accurate side-chain prediction (*see* **Fig. 15**). This exemplifies that such proteins would benefit even more from using a very detailed rotamer library.

In a recent review article ***(3)***, an alternative method developed by Bower et al. ***(43)*** is referred to as one of the most accurate methods for side chain prediction. Interestingly this method uses an MDRL in contrast to our DEE algorithm, which uses an MIRL. Bower et al. compare the average RMSD results per residue and per structure with the work of Holm and Sander ***(55)*** and Koehl and Delarue ***(58)***. The results of processing the results of our 22

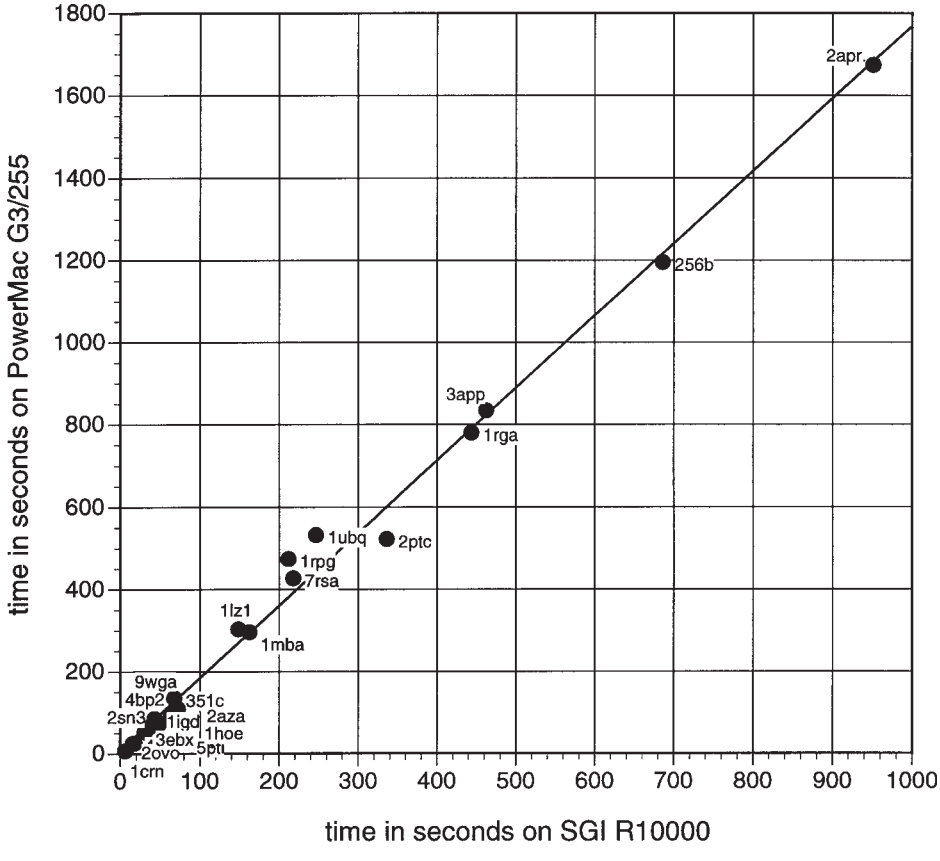


Fig. 16. Comparison of the total execution time of the complete DE-based side-chain prediction for a series of proteins, varying in size up to 325 residues, for two types of computers. The *x*-axis lists the total CPU time on a Silicon Graphics R10000 type of computer, whereas the *y*-axes shows the execution time on a Apple PowerMac G3/244. The slope of the scatterplot is 1.76.

proteins in the same way is collected in **Table 6**. As seen from the table, our method compares favorably to these three other methods. It is also observed that in the study of Bower et al. the scoring results using an MIRL are a lot worse. Both studies ***(6,43)*** demonstrate clearly that the choice and quality of the rotamer library is a keystone prerequisite for an accurate side-chain placement result. However, using a poorly defined MIRL will always score worse than a highly optimized MDRL. On the other hand, in the design of new sequences compatible with a given main chain ***(12)*** the usage of MDRL might be a better choice. Although not only is the number of rotamers at each residue position is limited, one also searches for sequences with a very high probabil-

Table 6
Study Processing Results

Study	Koehl and Delarue *(38)*	Holm and Sander *(33)*	Bower et al. *(43)*	DEE
Number of residues	subset	subset	36048	1944
Average RMSD (Å) per residue	1.33	1.25	1.25	0.98
Average RMSD (Å) per structure	2.01	1.96	1.93	1.60

RMSD of the side-chain atoms using four different methods. The measurements include all side-chain atoms beyond C_β, and are corrected for crystallographically symmetrical residues (Asp, Asn, Glu, Phe, Tyr).

ity for a given main-chain fold. In contrast with this, nature may allow very low rotamer probabilities in a particular protein, in which case an MIRL is more successful, and better suited in homology modeling work.

The Brugel modeling package has been ported to the Apple PowerMac series, without severe problems. (M. De Maeyer has ported the Brugel package, except for the graphical part, onto the Power Macintosh family of computers. This work has been carried out in a private collaboration with the company Beagle bvba.) All FORTRAN 77 and C code has been compiled under the macintosh programmers workshop (MPW) shell using the Absoft compilers *(44)*. No special optimizations were included to accelerate the execution of the code. Tests on the G3 family of PowerMacintosh for the same series of proteins (*see* **Fig. 16**) reveal that the high-end Silicon Graphics R10000 workstation version of the program is only 1.76 times faster compared to the G3. We are confident that current mathematical methods in protein engineering will also become available in the more affordable personal computer series.

7. Additional Reading

The current implementation of the DEE method has its major application field in the area of homology modeling. Recently, other fields of applications for the DEE algorithm have emerged *(9–13)*. Besides the DEE method to predict the side-chain conformation on a fixed backbone, other computational methods have been developed. These methods have been surveyed in two articles *(45,46)*. We summarize the major methods in this field: genetic algorithm *(47,48)*, simulated annealing *(49,50)*, lowest-energy conformation searching *(41)*, systematic search procedures in the context of the backbone atoms combined with extensive local energy minimizations *(23,43,51)*, local three-dimensional homology modeling *(52)*, combined sequence and side-

chain conformation network ***(53)***, Monte Carlo simulation ***(33,54–56)***, approaches where clusters of residues are examined instead of the whole protein ***(16,35,47)***, a self-consistant field method to iteratively refine a conformational matrix of protein side chains ***(57)***, segment matching method ***(58)***, the dead-end elimination method ***(8)***, and the DEE± and fuzzy-end elimination theorem ***(5,6,11,12,19,20,21,28)***.

Homology modeling also forms the first step in the complete prediction of protein folds. A review article putting all these tools in the perspective of the prediction of protein folds is worth reading and forms an excellent introduction in this emerging field ***(59)***.

Acknowledgments

J. Desmet thanks the Research Foundation of the KULeuven, the "Vlaams Instituut voor bevordering van het wetenschappelijk-technologisch onderzoek in de industrie" (IWT) for financial support and the Fund for Scientific Research–Flanders (F.W.O.). M. De Maeyer thanks the company Easyware for the kind loan of the G3 PowerMac. M. De Maeyer and I. Lasters are grateful to D. Collen for supporting this work.

References

1. Janin, J., Wodak, S., Levitt, M., and Maigret, B. (1978) Conformation of amino acid sidechains in proteins. *J. Mol. Biol.* **125,** 357–386.
2. Vásquez, M. (1996) Modeling sidechain conformation. *Curr. Opin. Struct. Biol.* **6,** 217–221.
3. Sánchez, R. and Šali, A. (1997) Advances in comparative protein-structure modelling. *Curr. Opin. Struct. Biol.* **7,** *206–214.*
4. Desmet, J., De Maeyer, M., Hazes, B., and Lasters, I. (1992) The dead-end elimination theorem and its use in protein sidechain positioning. *Nature (Lond.)* **356,** 539–542.
5. Tanimura, R., Kidera, A., and Nakamura, H. (1994) Determinants of protein sidechain packing. *Protein Sci.* **3,** 2358–2365.
6. M. De Maeyer, M., Desmet, J., and Lasters, I. (1997) All in one: a highly detailed rotamer library improves both accuracy and speed in the modelling of sidechains by dead end elimination. *Fold. Des.* **2,** 53–66.
7. Leach, A. R. (1994) Ligand docking to proteins with discrete sidechain flexibility. *J. Mol. Biol.* **235,** 345–356
8. Desmet, J., De Maeyer M., and Lasters, I. (1997) Computation of the binding of fully flexible peptides to proteins with flexible side chains. *FASEB J.* **44,** 164–172.
9. Rabijns, A., De Bondt, H. L., and De Ranter, C. (1997) Three-dimensional structure of staphylokinase, a plasminogen activator with therapeutic potential. *Nat. Struct. Biol.* **4,** 357–360.
10. Rashid, A. K., Van Hauwaert, M. L., Haque, M., Siddiqi, A. H., Lasters, I., De Maeyer, M., Griffon, N., Marden, M. C., Dewilde, S., Clauwaert, J., Vinogradov, S.

N., and Moens, L. (1997) Trematode myoglobins, functional molecules with a distal tyrosine. *J. Biol. Chem.* **272,** 2992–2999.
11. Dahiyat, B. I. and Mayo, S. L. (1996) Protein design automation. *Protein Sci.* **5,** 895–903.
12. Dahiyat, B. I. and Mayo, S. L. (1997) De novo protein design: fully automated sequence selection. *Science* **278,** 82–87.
13. Lasters, I., Desmet, J., and De Maeyer, M. (1997) Dead-end based modeling tools to explore the sequence space that is compatible with a given scaffold. *J. Protein Chem.* **16,** 449–452.
14. James, M. N. G., and Sielecki, A. R. (1983) Structure and refinement of penicillopepsin at 1. 8 Å resolution. *J. Mol. Biol.* **163,** 299–361.
15. McGregor, M. J., Islam, S. A., and Sternberg, M. J. E. (1987) Analysis of the relationship between sidechain conformation and secondary structure in globular proteins. *J. Mol. Biol.* **198,** 295–310.
16. Ponder, J. W. and Richards, F. M. (1987) Tertiary templates for proteins. Use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.* **193,** 775–791.
17. IUPAC-IUB Commission on Biochemical Nomenclature (1970) Abbreviations and symbols for the description of the conformation of Polypeptide chains. Tentative rules. (1969) *J. Mol. Biol.* **52,** 1–17.
18. Schrauber, H., Eisenhaber, F., and Argos, P. (1993) Rotamers: to be or not to be? An analysis of amino acid sidechain conformations in globular proteins. *J. Mol. Biol.* **230,** 592–612.
19. Lasters, I., De Maeyer, M., and Desmet, J. (1995) Enhanced dead-end elimination in the search for the global minimum energy conformation of a collection of protein sidechains. *Protein Eng.* **8,** 815–822.
20. Goldstein, R. F. (1994) Efficient rotamer elimination applied to protein sidechains and related spin glasses. *Biophys. J.* **66,** 1335–1340.
21. Lasters, I. and Desmet, J. (1993) The fuzzy-end elimination theorem: correctly implementing the sidechain placement algorithm based on the dead-end elimination theorem. *Protein Eng.* **6,** 717–722.
22. Press, W., Flannery, B. P., Tuekolsky, S. A., and Vetterling, W. T. (1989) in *Numerical Recipes. The Art of Scientific Computing*, Cambridge University Press, UK.
23. Eisenmenger, F., Argos, P., and Abagyan, R. (1993) A method to configure protein sidechains from the mainchain trace in homology modelling. *J. Mol. Biol.* **231,** 849–860.
24. Delhaise, Ph., Bardiaux, M., and Wodak, S. (1984) Interactive computer animation of macromolecules. *J. Mol. Graph.* **2,** 103–106.
25. Wodak, S., De Coen, J. L., Edelstein, S. J., Demarne, H., and Beuzard, Y. (1986) Modification of human hemoglobin by gluthadione. III. Perturbations of hemoglobin conformation analyzed by computer modeling. *J. Biol. Chem.* **261,** 14,717–14,724.
26. Warshel, A. and Levitt, M. (1976) Theoretical studies of enzymatic reactions: dielectric, electrostatic and steric stabilization of the carbonium ion in the reaction of lysozyme. *J. Mol. Biol.* **103,** 227–249

27. Brooks, B. R., Bruccoleri, R., Olafson, D., States, D. J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* **4,** 187–217.
28. Keller, K. A., Shibata, M., Marcus, E., Ornstein, R. L., and Rein, R. (1995) Finding the global minimum: a fuzzy end elimination implementation. *Protein Eng.* **8,** 893–904.
29. Bernstein, F. C., Tasumi, M., Koetzle, T. F., et al. (1977) The Protein Data Bank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.* **112,** 535–542.
30. Desmet, J., De Maeyer, M., and Lasters, I. (1994) The "The Dead, End Elimination" Theorem: A New Approach to the Side, Chain Packing Problem, in *The Protein Folding Problem and Tertiary Structure Prediction* (Merz, K. M. Jr and Le Grands, S. M., eds.), Birkhäuser, Boston, pp. 307–337.
31. Desmet, J., De Maeyer, M., and Lasters, I. (1997) Theoretical and algorithmical optimisations of the dead-end elimination theorem, in *Proceedings of the Pacific Symposium on Biocomputing '97* (Altman, R. B., Dunker, A. K., Hunter, L., and Klein, T. E., eds.) World Scientific, Signapore, pp. 122–133.
32. Miller, S., Janin, J., Lesk, A. M., and Chothia, C. (1987) Interior and surface of monomeric proteins. *J. Mol. Biol.* **196,** 641–656.
33. Holm, L. and Sander, C. (1992) Fast and simple Monte Carlo algorithm for sidechain optimization in proteins: application to model building by homology. *Proteins: Struct. Funct. Genet.* **14,** 213–223.
34. Tufféry, P., Etchebest, C., Hazout, S., and Lavery, R. (1993) A critical comparison of search algorithm applied to the optimization of protein sidechain conformations. *J. Comput. Chem.* **14,** 790–798.
35. Wilson, C., Gregoret, L. M., and Agard, D. A. (1993) Modeling sidechain conformation for homologous proteins using an energy-based rotamer search. *J. Mol. Biol.* **229,** 996–1006.
36. Lee, C. and Subbiah, S. (1991) Prediction of protein sidechain conformation by packing optimization. *J. Mol. Biol.* **217,** 373–388.
37. Laughton, C. A. (1994) Prediction of protein sidechain conformations from local three-dimensional homology relationships. *J. Mol. Biol.* **235,** 1088–1097.
38. Koehl, P. and Delarue, M. (1994) Application of a self-consistent mean field theory to predict protein sidechains conformation and estimate their conformational entropy. *J. Mol. Biol.* **239,** 249–275.
39. Hwang, J. and Liao, W. (1995) Sidechain prediction by neural networks and simulated annealing optimization. *Protein Eng.* **8,** 363–370.
40. Jones, T. A., Zou, J.-Y., Cowan, S. W., and Kjelgaard, M. (1991) Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Cryst.* **A47**, 110–119.
41. Schiffer, C. A., Caldwell, J. W., Kollman, P. A., and Stroud, R. M. (1990) Prediction of homologous protein structures based on conformational searches and energetics. *Proteins: Struct. Funct. Genet.* **8,** 30–43.
42. Alard, Ph and Wodak, S. J. (1991) Detection of cavities in a set of interpenetrating spheres. *J. Comp. Chem.* **12,** 918–922.

43. Bower, M. J., Cohen, F. E., and Dunbrack, R. L. Jr. (1997) Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a new homology modeling tool. *J. Mol. Biol.* **267,** 1268–1282.
44. Absoft, development tools and languages, 2781 Bond Street, Rochester Hills, MI 48309.
45. Šali, A. (1995) Modelling mutations and homologous proteins. *Curr. Opin. Biotechnol.* **6,** 437–451.
46. Vásquez, M. (1996) Modeling sidechain conformation. *Curr. Opin. Struct. Biol.* **6,** 217–221.
47. Tufféry, P., Etchebest, C., Hazout, S., and Lavery, R. (1991) A new appoach to the rapid determination of protein side chain conformations. *J. Biomol. Struct. Dyn.* **8,** 1267–1289.
48. Tufféry, P., Etchebest, C., Hazout, S., and Lavery, R. (1993) A critical comparison of search algorithm applied to the optimization of protein sidechain conformations. *J. Comput. Chem.* **14,** 790–798.
49. Correa, P. E. (1990) The building of protein structures from alpha-carbon coordinates. *Proteins: Struct. Funct. Genet.* **7,** 366–377.
50. Lee, C. and Subbiah, S. (1991) Prediction of protein sidechain conformation by packing optimization. *J. Mol. Biol.* **217,** 373–388.
51. Chinea, G., Padron, G., Hooft, R. W., Sander, C., and Vriend, G. (1995) The use of position-specific rotamers in model building by homology. *Proteins* **23,** 415–421.
52. Laughton, C. A. (1994) Prediction of protein sidechain conformations from local three-dimensional homology relationships. *J. Mol. Biol.* **235**,1088–1097.
53. Kono, H. and Doi, J. (1994) Energy minimization method using automata network for sequence and sidechain conformation prediction from given backbone geometry. *Proteins: Struct. Funct. Genet.* **19,** 244–255.
54. Holm, L. and Sander, C. (1991) Database algorithm for generating protein backbone and sidechain co-ordinates from a Cα trace: application to model building and detection of co-ordinate errors. *J. Mol. Biol.* **218,** 183–194.
55. Lee, C. (1994) Predicting protein mutant energetics by self-consistent ensemble optimization. *J. Mol. Biol.* **236,** 918–939.
56. Shenkin, P. S., Farid, H., and Fetrow, J. S. (1996) Prediction and evaluation of side-chain conformations for protein backbone structures. *Proteins* **26,** 323–352.
57. Koehl, P. and Delarue, M. (1994) Application of a self-consistent mean field theory to predict protein sidechains conformation and estimate their conformational entropy. *J. Mol. Biol.* **239,** 249–275.
58. Levitt, M. (1992) Accurate modelling of protein conformation by automatic segment matching. *J. Mol. Biol.* **226,** 507–533.
59. Dandekar, T. and König, R. (1997) Computational methods for the prediction of protein folds. *Biochim. Biophys. Acta.* **1343**, 1–15.
60. Hendrickson, W. A., Teeter, M. M. (1981) Structure of the hydrophobic protein crambin determined directly from the anomalous scattering of sulphur. *Nature* **290,** 107.

61. Watenpaugh, K. D., Sieker, L. C., Jensen, L. H. (1980) Crystallographic refinement of rubredoxin at 1.2 Å degrees resolution. *J. Mol. Biol.* **138,** 615–633.
62. Bode, W., Epp, O., Huber, R., Laskowski, M. Jr., and Ardelt, W. (1985) The crystal and molecular structure of the third domain of silver pheasant ovomucoid (OMSVP3) *Eur. J. Biochem.* **147,** 387–395.
63. Wlodawer, A., Walter, J., Huber, R., and Sjolin, L. (1984) Structure of bovine pancreatic trypsin inhibitor. Results of joint neutron and X-ray refinement of crystal form II. *J. Mol. Biol.* **180,** 301–329.
64. Derrick, J. P. and Wigley, D. B. (1994) The third IgG-binding domain from streptococcal protein G. An analysis by X-ray crystallography of the structure alone and in a complex with Fab. *J. Mol. Biol.* **243,** 906–918.
65. Smith, J. L., Corfield, P. W., Hendrickson, W. A., and Low, B. W. (1988) Refinement at 1.4 Å resolution of a model of erabutoxin b: treatment of ordered solvent and discrete disorder. *Acta Crystallogr. A.* **44,** 357–368.
66. Zhao, B., Carson, M., Ealick, S. E., and Bugg, C. E. (1992) Structure of scorpion toxin variant-3 at 1.2 Å resolution. *J. Mol. Biol.* **227,** 239–252.
67. Pflugrath, J. W., Wiegand, G., Huber, R., and Vertesy, L. (1986) Crystal structure determination, refinement and the molecular model of the alpha-amylase inhibitor Hoe-467A. *J. Mol. Biol.* **189,** 383–386.
68. Vijay-Kumar, S., Bugg, C. E., and Cook, W. J. (1987) Structure of ubiquitin refined at 1.8 Å resolution. *J. Mol. Biol.* **194,** 531–544.
69. Matsuura, Y., Takano, T., and Dickerson, R. E. (1982) Structure of cytochrome c551 from Pseudomonas aeruginosa refined at 1.6 Å resolution and comparison of the two redox forms. *J. Mol. Biol.* **156,** 389–409.
70. Zegers, I., Haikal, A. F., Palmer, R., and Wyns, L. (1994) Crystal structure of RNase T1 with 3'-guanylic acid and guanosine. *J. Biol. Chem.* **269,** 127–133.
71. Lederer, F., Glatigny, A., Bethge, P. H., Bellamy, H. D., and Matthew, F. S. (1981) Improvement of the 2.5 Å resolution model of cytochrome b562 by redetermining the primary structure and using molecular graphics. *J. Mol. Biol.* **148,** 427–448.
72. Finzel, B. C., Weber, P. C., Ohlendorf, D. H., and Salemme, F. R. (1991) Crystallographic refinement of bovine pro-phospholipase A2 at 1.6 Å resolution. *Acta Crystallogr.* **47,** 814–816.
73. Wlodawer, A., Svensson, L. A., Sjolin, L., and Gilliland, G. L. (1988) Structure of phosphate-free ribonuclease A refined at 1.26 Å. *Biochemistry* **27,** 2705–2717.
74. Zegers, I., Maes, D., Dao-Thi, M. H., Poortmans, F., Palmer, R., and Wyns, L. (1994) The structures of RNase A complexed with 3'-CMP and d(CpA): active site conformation and conserved water molecules. *Protein Sci.* **3,** 2322–2339.
75. Baker, E. N. (1988) Structure of azurin from Alcaligenes denitrificans refinement at 1.8 Å resolution and comparison of the two crystallographically independent molecules. *J. Mol. Biol.* **203,** 1071–1095.
76. Artymiuk, P. J. and Blake, C. C. (1981) Refinement of human lysozyme at 1.5 Å resolution analysis of non-bonded and hydrogen-bond interactions. *J. Mol. Biol.* **152,** 737–762.

77. Bolognesi, M., Onesti, S., Gatti, G., Coda, A., Ascenzi, P., and Brunori, M. (1989) Aplysia limacina myoglobin. Crystallographic analysis at 1.6 Å resolution. *J. Mol. Biol.* **205,** 529–544.
78. Wright, C. S. (1990) 2.2 A resolution structure analysis of two refined N-acetylneuraminyl-lactose–wheat germ agglutinin isolectin complexes. *J. Mol. Biol.* **215,** 635–651.
79. Huber, R., Steigemann, W., Kukla, D., et al. (1974) Structure of the complex formed by bovine trypsin and bovine pancreatic trypsin inhibitor. II. Crystallographic refinement at 1.9 Å resolution. *J. Mol. Biol.* **89,** 73–101.
80. James, M. N. G. and Sielecki, A. R. (1983) Structure and refinement of penicillopepsin at 1.8 Å resolution. *J. Mol. Biol.* **163,** 299–361.
81. Suguna, K., Bott, R. R., Padlan, E. A., Subramanian, E., Sheriff, S., Cohen, G. H., and Davies, D. R. (1987) Structure and refinement at 1.8 Å resolution of the aspartic proteinase from Rhizopus chinensis. *J. Mol. Biol.* **196,** 877–900.

13

Classification of Protein Folds

Robert B. Russell

1. Introduction

The classification of three-dimensional (3D) structures now plays a central role in understanding the principles of protein structure, function, and evolution. Classification of new structures can provide functional details through comparison to others, which is of growing importance as X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy can now produce structures in advance of biochemical characterization (e.g., **ref.** ***1***). More generally, structure classifications themselves provide an excellent source of data for analyzes of all kinds.

This chapter presents a strategy for classifying protein structures. Steps in the classification procedure — domains, structural class, folds, superfamilies — are discussed in turn by reference to examples and relevant literature. Methods are discussed for discerning when structural similarities are most likely to indicate an evolutionary and/or functional similarity when sequence similarity is absent. Finally, a review of the most widely used Internet-based classifications is given.

2. Methods

2.1. Secondary Structure

Protein folds are nearly always described in terms of the type and arrangement of secondary-structures (i.e., α-helices and β-strands), thus secondary-structure definition is a good first step in classification. A detailed review of methods for assigning secondary-structure is beyond the scope of this chapter. The reader is directed to references ***(2–4)*** and those therein.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

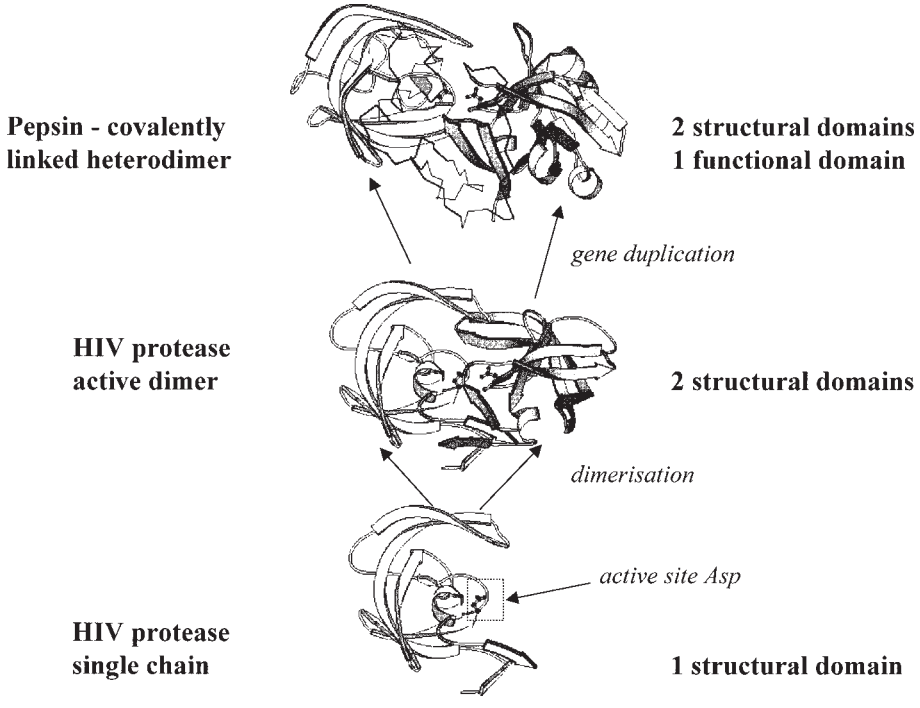


Fig. 1. Molscript ***(68)*** figure showing ambiguities in domain assignment for the aspartyl proteases. The bottom of the figure shows a single subunit from the HIV protease (PDB ***(69)*** code 1hiv-a) dimer shown in the middle of the figure. This dimer is equivalent to the covalently linked heterodimer found in the eukaryotic aspartyl protease (pepsin, PDB code 4pep) shown at the top of the figure. The single subunit corresponds to a structural domain, the homo/heterodimer corresponds to the functional domain.

2.2. Domain Assignment

Domains conveniently divide protein structures into discrete subunits, which are frequently classified separately. Protein domains are usually defined by one or more of the following criteria (*see* **ref. *5*** and references therein):

1. Spatially separate regions of protein chains.
2. Sequence and/or structural resemblance to an entire chain from another protein.
3. A specific function associated with a region of the protein structure.
4. A substructure in another protein that meets one or more of requirements 1–3.
5. Repeating substructures within a single chain meeting one or more of requirements 1–3.

Definitions 2 and 3 do not necessarily agree, as structural units may not be associated with a specific function. Some of the best examples of this struc-

tural/functional domain disagreement can be seen in the trypsin-like serine proteases and the pepsin-like aspartyl (acid) proteases (*see* **Fig. 1**). In both examples the functional domain (i.e., the catalytic domain) consists of two similar structural domains (presumed to be related by gene duplication; e.g., **ref. *6***). For the aspartyl proteases (*see* **Fig. 1**) there is further ambiguity as the retroviral (e.g., HIV) proteases consist of only a single copy of the structural domain that is functionally active as a homodimer, rather than the covalently linked heterodimer found in eukaryotes (e.g., pepsin).

For analysis of the principles of protein structure, use of structural domains is preferable, as these probably fold independently (e.g., each "lobe" of the proteases), and internal pseudo-symmetry (i.e., duplicated domains) can add to the understanding of the fold. It can be difficult to assign structural domains given only sequence data, and functional domains are frequently known in the absence of 3D structure data. It is thus best to consider functional domains during fold recognition/threading studies, where a protein of unknown structure (and often a functional domain) is compared to a database of known structures.

Automated methods have been developed for structural domain assignment, which look for spatially separated compact units ***(7–10)*** or hydrophobic cores ***(11)***. Methods can disagree even for relatively simple cases. A good strategy, adopted by the authors of Class Architecture Topology Homology (CATH) ***(12,13)***, is to combine the results of several algorithms with visual inspection, as often at least one of the methods will be correct. The property of recurrence is also very useful in defining domains. If a fragment of a larger protein is observed in isolation, or in a different domain context, then this adds confidence to the assignment of the segment as an independent folding unit ***(5,14)***.

Domains need not comprise single continuous segments of the polypeptide chain. Domain shuffling during protein evolution means that domains can be inserted into one another ***(15)***, making multisegmented (i.e., discontinuous) domains (*see* example 2 in **Section 3.**).

2.3. Assignment of Structural Class

After assignment of secondary-structures and domains, structural class can be assigned to domains. Structural classes divide proteins according to secondary-structure element content and organization. Globular proteins were first grouped into four classes ***(16)***: all α (or α/α), all β (or β/β), α/β, and α + β. However, a fifth class, small or irregular, is now generally used to group those proteins with few secondary-structures (often containing multiple disulphides or metals).

2.3.1. One Secondary Structure Type: All α or All β

Class assignment is usually straightforward for domains with predominantly α-helices or β-sheets. Small elements of secondary-structure, such as 3^{10} heli-

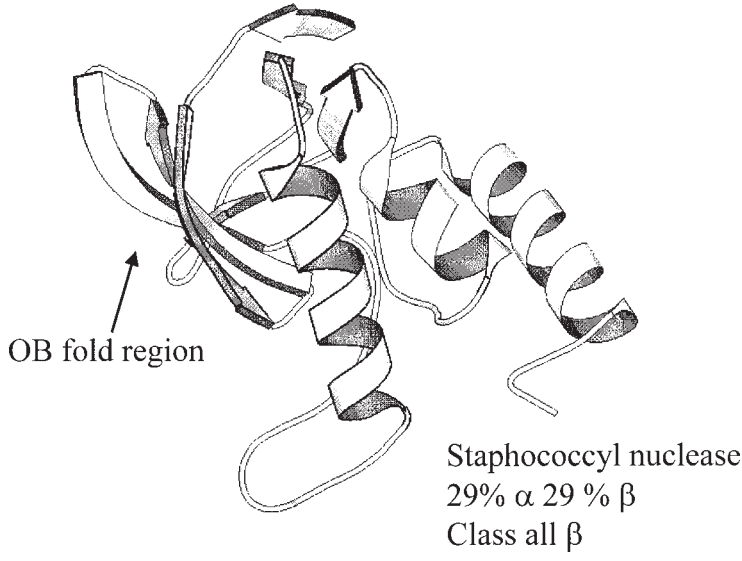


Fig. 2. Molscript ***(68)*** figure showing and example of difficulties in fold class assignment. Staphococcyl nuclease (PDB code 1snc). The protein contains an equal proportion of β-strands and α-helices, but is generally classified as all β because of the β-barrel domain forming the core of the OB fold.

ces, or small β hairpins are usually ignored in the assignment. Class is somewhat subjective, and may be based on the structure of the protein core rather than the abundance of α or β residues. Consider, e.g., staphococcyl nuclease (*see* **Fig. 2**), which contains an equal proportion of residues in α helices and β strands, but is generally classed as all β ***(12,17)*** because the core of the fold comprises an oligonucleotide/oligosaccharide (OB) binding-fold β barrel. Structural similarity may thus affect the assignment of class.

2.3.2. Both Secondary Structure Types: α/β or α + β

Protein domains that contain a mixture of α helices and β sheets are more difficult to classify. Historically ***(16)***, α/β proteins are those containing both α helices and β sheets and where there is an intimate association of helices and strands. In contrast, α + β proteins define those consisting of segregated regions of helix and sheet. More recently, and perhaps most exemplified by the Structural Classification of Proteins (SCOP) database, α/β proteins tend to refer to those structures containing many βαβ units, which consist of two adjacent (e.g., hydrogen bonded) β strands connected by a single a a helix in a right-handed connection, whereas α + β proteins are those not falling easily into this definition. The authors of the CATH database ***(12)*** have done away with the

distinction between α/β and α + β, arguing that it is an architectural distinction, rather than an inherent difference in secondary structure content.

2.3.3. Other Classes

Proteins with few secondary structures form a category of their own. Frequently these proteins are small, with the tertiary structure dominated by multiple disulphide bonds, or one or more metal-binding sites. Other fold classes are used to contain peptides, or multidomain proteins for which no logical single class or domain divisions can be assigned.

Holm and Sander ***(18)*** positioned all structures in the protein database in a high-dimensional, abstract fold space. When multivariate scaling was used to project these positions onto a two-dimensional (2D) density plot, five "attractors" (peaks) were found to cover approx 40% of known folds. These attractors were found to correspond to architectural features: parallel β, β-meander, α-helical, β-zigzag, and αβ-meander. Their, analysis thus provides an alternative means to define the "class" of a protein, although some of the attractors match the traditional classes closely.

2.4. Assignment of Fold

The number, type, connectivity, and arrangement of secondary-structures define the fold of a protein. Frequently, fold similarities are recognized by eye-following structure determination, although there are many papers published following a structure determination reporting a structure/function similarity not noted by the experimentalists (for examples see **refs. *19,20***; for reviews *see* **refs. *21–23***). Fold searching should thus be done with care, and similarities should be considered in a wider context that includes functional similarity.

It is best to compare each domain of a new structure to a database of those already known. Even in instances when the fold is known, such searches can reveal relationships that might be missed. For example, a protein may be easily seen to adopt an immunoglobulin (Ig)-like β-sandwich structure, but a structure with a similar function may be buried in the large group of Ig-like folds.

There are several means of searching protein structure databases with a probe structure (*see* **refs. *21,22*** for reviews). Programs such as SSAP ***(24,25)***, SARF ***(26)***, and STAMP ***(27)*** are available from the respective authors (*see* Appendix at the end of the chapter). It is also possible to run DALI (the engine of the Families of Structurally Similar Proteins [FSSP] database ***[28,29]***) via the World Wide Web (*see* Appendix), and methods similar to the structure comparison technique of Artymiuk et al. ***(30)*** are encoded in QUANTA ***(31)***, and VAST ***(32)*** at the National Center for Biology Information (NCBI) (although VAST comparisons are only currently available for protein structures already

in the database). Structure comparison is also possible within the O crystallographic package via the program Déjà vu ***(33)***. Different methods can give different results, particularly if structural similarity is slight. It is, therefore, prudent to run several algorithms and arrive at a consensus.

A phenomenon to consider during fold assignment is circular permutation, which relates domains that are similar in structure and/or sequence, yet whose N- and C-terminal portions have been exchanged. Permutations are real events in nature (*see* **ref. *34*** and references therein; for a recent example, *see* **refs. *35,36***). Although some structure comparison methods permit matches involving differences in connectivity, few, if any, are able to detect permutations directly.

2.5. Assignment of Superfamily

It is probably impossible to state definitively whether all proteins adopting a similar fold are descended from a similar common ancestor (i.e., related through divergence). For many proteins with similar folds, sequence, structure, or functional arguments suggest divergence from common ancestor; for others, no such conclusion can be drawn. Hence, some classifications distinguish between similarities that are due to divergent evolution, and those that may not be. It is clear that many homologous proteins have simply diverged beyond the point where sequence similarity can be detected. The term *superfamily* is often used in structure classification to refer to groups of proteins that appear to be homologous, even in the absence of significant sequence similarity.

Proteins with the same fold that are not thought to share a common ancestor are often referred to as *analogs* (to distinguish them from homologs), and are thought by many to be the result of a convergence to a stable structure. Although there is little hard evidence, there are some arguments that favor such a convergence. The number of proteins sampled during evolutionary time is vast, despite an estimated low number of possible folds ***(37–40)***, which may be due to restrictions on protein architecture. In addition, recent studies on sequence identity, calculated from structure-based sequence alignments ***(41,42)***, show a bimodal distribution. Although the results are very preliminary, the biomodality may suggest two origins for similarities between protein folds: analogy and homology.

How can analogy and homology be distinguished? A survey of recent literature (e.g., **refs. *38,41–45***) shows that one or more of the following features are often used to deduce a common ancestor (i.e., assign a common superfamily) given a pair of similar 3D structures:

1. Above a certain level of structural similarity, even if sequence similarity is insignificant, one can be largely confident of a homologous relationship ***(32,38,45)***.
2. The conservation of unusual structural features, sometimes outside the common core secondary structure elements. These features include functionally important turn conformations ***(46)***, left-handed βαβ units ***(47)***, or others (e.g., **ref. *43***).

3. Low — but significant — sequence identity as calculated after structure superimposition (i.e., the identity from the structure-based alignment ***(41)***). *See* **ref. *43*** for guide (illustrated by example) to how to calculate an associated statistical significance. It has been suggested ***(41)*** that sequence identities from a structure-based alignment of >12% are more likely to indicate a remote homology. Note also that structure similarities may confirm marginally significant sequence similarities seen prior to 3D structure determination.
4. The presence of key active site residues, even in the absence of global sequence similarity. This is most often applicable to enzymes, for examples, *see* **refs. *20*,*48*–*50***.
5. Sequence similarity bridges, or *transitivity*. Even though two sequences may not be significantly similar to one another, inspection of homologs found in sequence searches with each sequence may reveal a "link" or "sequence bridge" linking the two sequences via significant sequence similarities (e.g., **refs. *49–51***). In other words, if domain A is significantly similar to domain B, and domain C is significantly similar to domain B, then domains A and C can generally be deamed homologous.

2.6. Superfolds, Superfamilies, and Supersites

Certain protein folds are populated by many different superfamilies, suggesting that the fold has arisen many times by convergent evolution. Such folds have been termed *superfolds* ***(38)***. For most of these folds, the core structure is highly symmetrical. Symmetry may imply an easier folding pathway and make convergence more likely than for less symmetrical folds, which often comprise only a single superfamily (e.g., aspartyl proteases). Examples of superfolds include β/α-triace phosphate isomerase (TIM)-barrels, Rossmann-like α/β-folds, ferredoxin-like folds, β-propellers, four-helical up-and-down bundles, Ig-like-β-sandwiches, β-jelly rolls, OB binding-folds, and SH3-like folds. All are adopted by groups of seemingly nonhomologous protein, which perform different functions ***(17*,*38)***.

For some of these superfolds, including the β/α-(TIM)-barrels, Rossmann-like α/β-folds, ferredoxin-like folds, β-propellers, four-helical up-and-down bundles, proteins from different superfamilies show a tendency to bind ligands in a common location (even when the nature of the bound atoms is different). These locations have been termed supersites, as they occur, by definition, within superfolds ***(52)***. Rather than being due to a divergence, supersites are thought to be a property of the protein fold, such as the alignment of nonhydrogen-bonded main-chain atoms, or the "helix dipole" ***(53)***, that dictates the best location for binding non-protein atoms, regardless of evolutionary origin. For some superfolds, it is thus possible to make predictions as to binding-site locations even in the absence of evidence of a common ancestor.

2.7. Predicting Function from Structural Similarity

Proposals have been made to determine large numbers of protein structures with the explicit aim of assigning function ***(54)***. After analysis of all structures

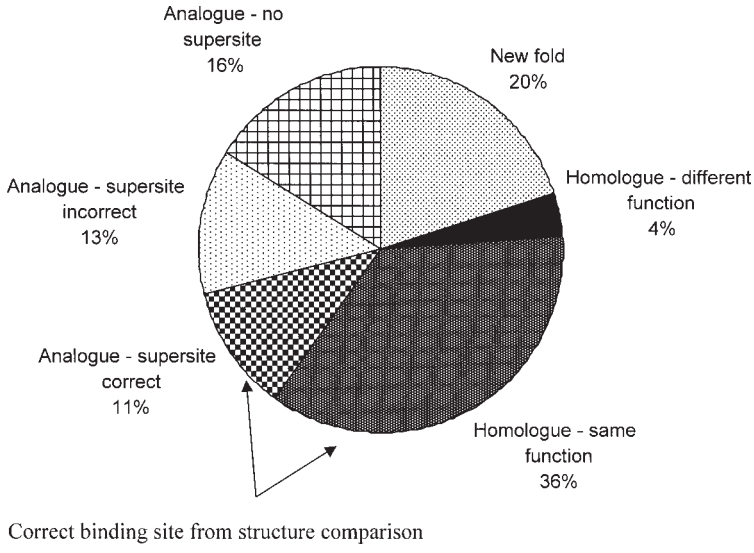


Fig. 3. Pie chart showing how often new structures will have correct binding site information predicted through structure comparison (adapted from **ref. *52***).

within the SCOP database, Russell et al. ***(52)*** estimated the fraction of new structures (ignoring those that are obviously sequence similar to a known structure) that will currently show binding site or functional similarity via structure comparison. This estimate (illustrated by a pie chart in **Fig. 3**) was based on the distribution of homologous and analogous similarities within SCOP, and the fraction of superfolds containing supersites (i.e., how often do analogs have a common binding site?). Currently just under half of new structures will have accurate binding-site information predicted through structure comparison, which highlights the danger of interpreting every structural similarity as an indication of a common function.

3. Two Examples

Two examples of protein structure similarities are described as below. In both instances the similarity was not reported by the crystallographers. Both similarities had clear biological implications that augmented the understanding of protein function following structure determination.

Example 1: β-glucosyltransferase

The structure of β-glucoysltransferease (BGT) was originally reported to contain two domains of similar topology, each reminiscent of a nucleotide binding fold ***(55)***. Subsequent comparison of BGT to other known structures during

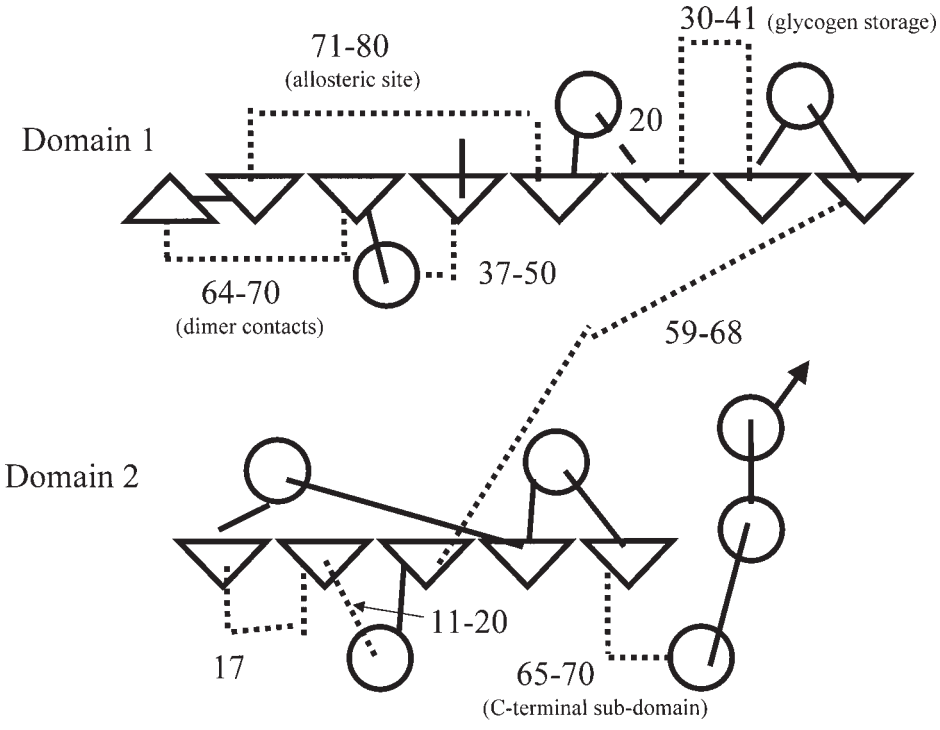


Fig. 4. Molscript ***(68)*** figure showing the similarity between adenylyl cyclase (PDB code 1ab8) and the palm domain from DNA polymerase (PDB code 1dpi). The location of key aspartyl residues is shown (*N.B.*: these are serines in the cyclase domain shown, but are aspartates in the other similar domain forming the active heterodimer). Equivalent regions in the two structures are shown as ribbons/coil, non equivalent regions (including the inserted fingers domain) are shown in C_α trace.

two independent studies revealed a close similarity with glycogen phosphorylase, GPB ***(56,57)***. BGT and GPB differ greatly in length: BGT contains only 351 amino acids compared to GPB's 842. Despite this, 13 β-strands and 9 α-helices are equivalent (**Fig. 4**), and 256 pairs of C_α atoms can be superimposed with a root-mean-square deviation (RMSD) of 3.4 Å ***(56)*** (alternatively, 114 C_α atoms can be superimposed with an RMSD of 1.72 Å ***[57]***). The common fold comprises the entire core BGT structure, with GPB containing numerous long insertions, which appear to modulate function (*see* **Fig. 4**). Superimposition also reveals striking similarities in the active sites of the two enzymes (despite surprisingly few residue identities). The observations lead to the suggestion that BGT and GPB share an ancient common ancestor.

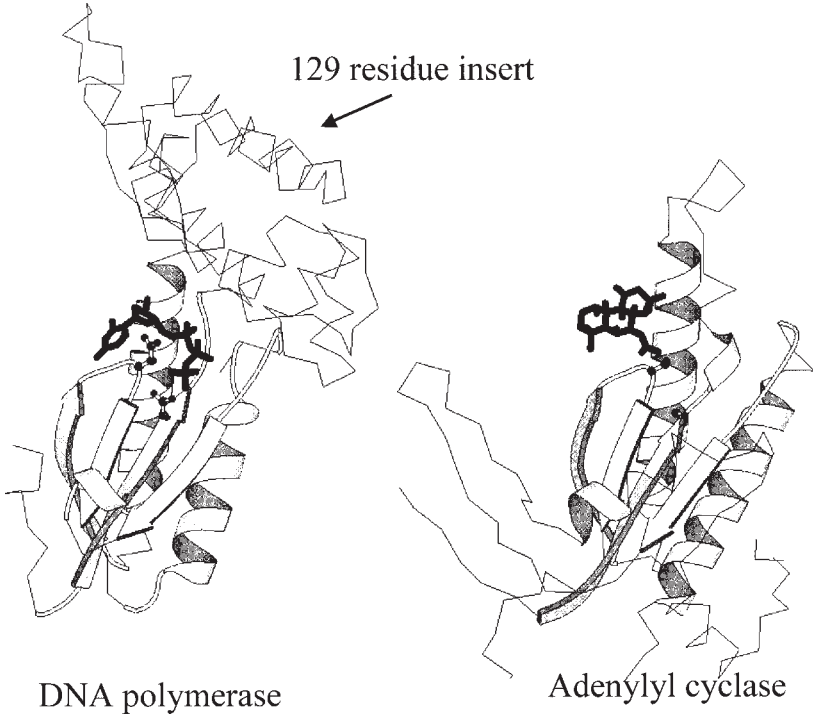


Fig. 5. Topology diagram showing the similarity between bacterial β-glucosyltransferase (BGT; PDB code 1bgu) and glycogen phosphorylase b (GBP; PDB code 1gpb). The figure was adapted from **refs.** ***56*** and ***57***. Dashed lines indicate those regions in the core of the fold where GPB contains very long insertions relative to BGT. Descriptions of the function of each insertion are given where known.

3.2. Example 2: Adenylyl Cyclase

The structure of adenylyl cyclase structure was originally reported to contain a new protein fold ***(58)***. However, subsequent comparison of the structure to the database found a striking similarity with DNA/RNA polymerases ***(20)*** (*see* **Fig. 5**). The core fold adopts the very common ferredoxin-like fold, and although this fold is seen in many proteins, cyclases and polymerases have a similar binding site, a similar reaction mechanism, and both contain key Mg^{2+} binding aspartate residues ***(59)***, known to be critical for polymerase function. The similarity thus provides key insights into the mechanism of the less-well-understood cyclases.

4. Protein Structure Classifications

Several protein structure classification schemes have become available via the Internet over the last 5 yr. Below the relative merits of each are discussed.

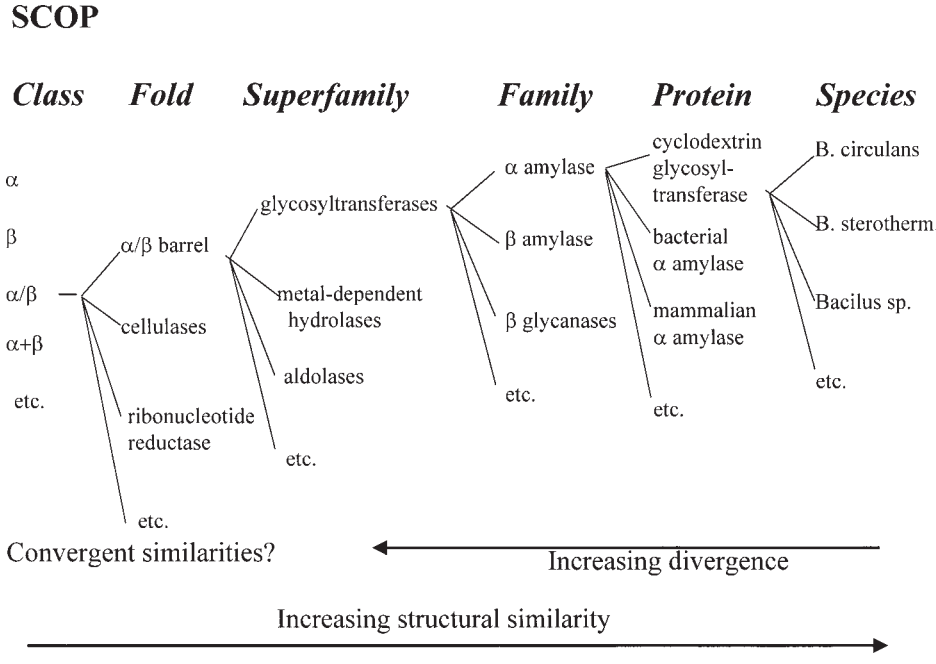


Fig. 6. Example of the classification hierarchy for the SCOP database.

Perhaps the most important general comment is that none of the classifications give a complete picture. Because all have different strengths, it is best to consider as many as possible.

4.1. SCOP

SCOP is maintained by Murzin et al. ***(17)*** in Cambridge, and is an entirely manual classification. The class definitions are after Levitt and Chothia ***(16)***. Proteins are generally divided into functional domains, and these are grouped into a hierarchy consisting of class, fold, superfamily, family, protein, and species. Proteins are put into the same fold if they have a similar core, which is decided by manual analysis. The fold definitions in SCOP are more stringent than the other classifications, and several similar structures are not put into the same fold, sometimes simply to avoid exceedingly large groups of structures (for example, the three layered α–β–α Rossmann-fold like structures; A. G. Murzin, personal communication). The subdivisions within each fold (superfamily, family, protein, and species) group proteins according to the degree of homology. **Figure 6** shows a schematic example of the SCOP classification scheme.

The great strength of SCOP is the very careful assignment of evolutionary relationships, even in the absence sequence similarity. Proteins in the same

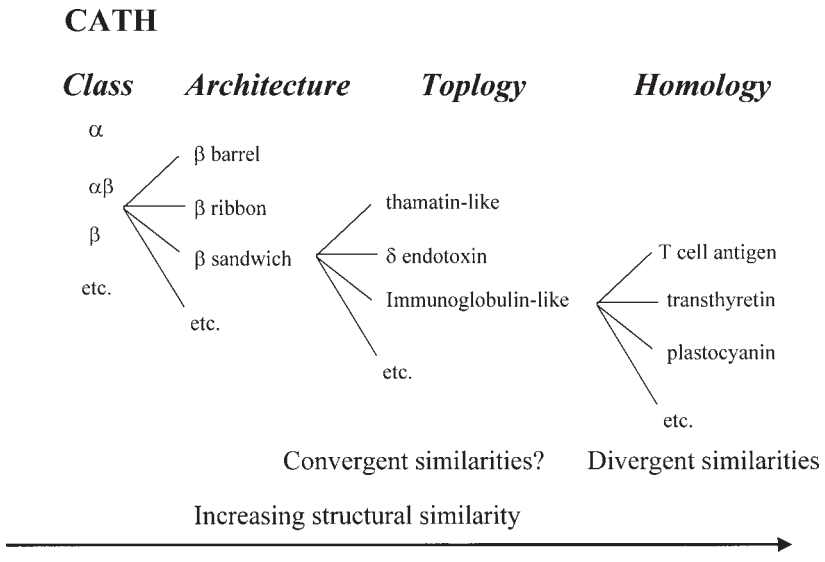


Fig. 7. Example of the classification hierarchy for the CATH database (adapted from **ref.** ***12***).

fold, but in different superfamilies are lacking in evidence for a common ancestor (analogous folds); those in the same fold and the same superfamily show some evidence of a common ancestor, which is often based on the features discussed above.

4.2. CATH

CATH is maintained by Orengo et al. ***(12)*** at University College, London. The classification is partly automated and partly manual, although they work toward a mostly automated system. They classify proteins according to a hierarchy that is similar to SCOP, although with some important differences. The class (C) layer is directly equivalent to that found in SCOP, with the exception that no distinction between α/β and α + β domains is made. The Architecture (A) layer, a unique feature of CATH, is an intermediate between class (C) and fold (or topology, T, in CATH). Protein architecture defines the orientation of the secondary structures composing the fold, independent of the connectivity or direction of secondary structure elements. For example, all protein domains containing a β-barrel regardless of the number of strands forming the barrel, or the connectivity are placed in a single architecture. The extra level to the hierarchy makes browsing classifications somewhat easier, and it makes structure space more continuous than in some of the other classifications. Architecture also encapsulates many of the descriptions often given with newly solved 3D

structures (e.g., the protein contains a β-barrel structure). **Figure 7** shows a schematic example of the CATH classification scheme.

CATH provides excellent peripheral information for every protein structure in the database. Detailed graphical information as to structural motifs ***(60)***, bound ligands ***(61)***, and cross-references to many other data sources are all available.

4.3. FSSP

The FSSP database is provided by Holm and Sander ***(29)*** at the European Bioinformatics Institute (Hinxton, UK). Rather than a classification, FSSP is a list of protein structural neighbors. After each update of the PDB, each new protein is compared to all others using sequence and structure comparison methods. Thus, for each PDB entry, one can obtain a list of sequence and structure neighbors (the results of a search). There are no discrete boundaries discerning similarity from dissimilarity. Frequently, proteins that are not grouped in, e.g., SCOP or CATH, are structural neighbors within FSSP, reflecting weaker matches not always captured by other classifications that may, nevertheless, represent biologically meaningful examples (*see* **ref. *48*** for an example). The main drawback to the FSSP database at present is the lack-of-domain definitions. Multidomain proteins are compared as a whole to the database, meaning that it may be difficult to see similarities involving a rare domain when it is connected to one occurring frequently.

4.4. VAST

VAST is a program for structure comparison written by Bryant and co-workers ***(32)*** at the National Institutes of Health NCBI, and forms the basis for adding structural information to the ENTREZ database facility ***(62)***. Like FSSP, this database is more a list of similarities than a classification, with the same neighboring system found throughout the ENTREZ system. It has the advantages of being updated immediately following protein databank updates, and because of its location in ENTREZ, it contains excellent links to protein and nucleotide sequence data, and to Medline literature references.

4.5. 3Dee

The Protein Domains Database (PDB) (Barton et al., European Bioinformatics Institute, in press), provides a set of carefully defined protein domains for the entire protein databank, a structural classification and links to SCOP and other databases. In addition to providing a graphical view of superimpositions, 3DEE also provides the unique ability to view different domain assignments.

5. Notes

1. Domain assignment: Repeating units in a structure need not define individual domains, as many single domain structures possess internal symmetry, e.g., the

β-trefoils, which contain three similar trefoil motifs that form a single domain. It is unlikely that the motifs could fold or be functionally active in isolation.

2. Class assignment: Assign class based on the core structure. For example, if a protein contains a β-barrel with numerous helical insertions, then it is usually best classified as all-β.
3. Fold assignment: If you have similarities involving separate domains, attempt to extend them by adding domains. For example, BGT (*see* **Section 3**), was originally commented to have two Rossmann fold domains ***(55)***. However, both structural domains can be superimposed on glycogen phosphorylase ***(56,57)***, indicating an ancient common ancestor.
4. Superfamily assignment:
   a. The structure with the highest degree of structural similarity to a probe structure may not necessarily be the best candidate for superfamily or functional similarity (*see* adenylyl cyclase and DNA polymerase ***(20)*** in the NCBI–VAST database ***[63]***). Partly this can be due to limitations in the structure comparison method.
   b. Even homologous protein structures can have different functions (e.g., **refs. *43,64,65***). Consider, e.g., the similarity between sonic hedgehog (a factor) and DD carboxypeptidase (an enzyme) ***(65)***.
   c. A common binding-site location is not sufficient to group proteins into the same superfamily, as some proteins appear to show binding-site similarity in the absence of homology (e.g., the α/β-barrels; *see* **refs. *52,66***).
5. Classifications:

SCOP *Advantages*

a. Classification is done manually, and with careful consideration of the literature.
b. Includes classifications for structures for which no coordinates are publicly available.
c. Evolutionary classification is better than any other system.
d. Interactive interface to local copy of protein databank (via RasMol (67]).

SCOP *Limitations*

a. Groupings at the fold level are fairly stringent, meaning that similar structures are often not grouped together. Note that this means that proteins belonging to different folds in SCOP can still show some degree of structural similarity (e.g., Ig folds and cupredoxins).
b. Fold/Superfamily definitions are not static. This is also an advantage, as misclassifications are corrected when more information becomes available.
c. Some folds have been studied in more detail than others.
d. Updates only occur about twice annually.
e. No facility for viewing alignments or superimpositions to date.

CATH *Advantages*

a. Groupings at the fold level are more lenient than in SCOP, and more useful for tasks like the assessment of protein fold recognition.

b. Architecture division makes classification easier to follow.
c. Excellent peripheral resources (e.g., Rasmol, ligand binding, structural characterization, and enzyme-classification annotation).
d. Careful assignment of proteins into domains.

CATH *Limitations*

a. Updates are infrequent.
b. No facility for viewing alignments or superimpositions to date.
c. Domains are often structural, which means that some fold/superfamily similarities are missed (e.g., the trypsin-like serine proteases).

FSSP/DALI *Advantages*

a. Fully automated, and as up to date as the PDB.
b. Provides good interactive interface to view both superimpositions and alignments of structures.
c. Ability to search the PDB with a new structure.
d. Statistical measure provides reliable significance for each similarity.

FSSP/DALI *Limitations*

a. Fully automated, thus can contain some misclassifications owing to lack of human interpretation.
b. Currently, the lack of domain assignments can make classification of multidomain proteins difficult.

VAST/NCBI *Advantages*

a. Fully automated, and as up to date as the protein databank.
b. Excellent crossreferencing to protein databank, protein/DNA sequence and literature data through the Entrez system ***(62)***.
c. Statistical measure provides reliable significance for each similarity.

VAST/NCBI *Limitations*

a. Similarities are detected based on arrangements of secondary-structures, which means some similarities may be missed owing to poor definitions.
b. No domain definitions at present.

Acknowledgments

The author is grateful to Chris Rawlings, David Searls, and Ford Calhoun (SmithKline Beecham) for encouragement, and Mike Sternberg (Imperial Cancer Research Fund) for helpful discussions. Thanks also go to Richard Copley for a detailed proofreading of the manuscript.

Appendix URLs

Structural Classifications

SCOP (MRC/LMB Cambridge, UK): http://scop.mrc-lmb.cam.ac.uk/scop (mirrors around the world)
CATH (University College, London, UK): http://www.biochem.ucl.ac.uk/bsm/cath
FSSP/DALI (European Bioinformatics Institute, Cambridge, UK): http://www2.ebi.ac.uk/dali/fssp/fssp.html
NCBI/VAST (NCBI, NIH, Bethesda, MD): http://www.ncbi.nlm.nih.gov/Structure/vast.html
DDBASE (Department of Biochemistry, Cambridge University, UK): http://www-cryst.bioc.cam.ac.uk/~ddbase/
3DEE (EBI, Cambridge, UK): http://circinus.ebi.ac.uk:8080/3Dee/help/help_intro.html

Algorithms

Data

Protein Databank (PDB): http://www.pdb.bnl.gov/

Secondary Structure Assignment

DSSP: ftp://ftp.ebi.ac.uk/pub/software/unix/dssp/
STRIDE: http://www.embl-heidelberg.de/stride/stride.html

Domain Assignment

DAD algorithm: http://www.icnet.uk/bmm/domains/assign.html calculates domains given a set of coordinates
DOMAK program: http://barton.ebi.ac.uk/ downloadable program for calculating domains

Structure–Database Comparison

DALI: http://www2.ebi.ac.uk/dali/dali.html — compares a query set of protein coordinates to a database of known structures
SSAP: http://www.biochem.ucl.ac.uk/~orengo/ssap.html — information on downloading the SSAP program for protein structure alignment and superimposition.
SARF: http://www-lmmb.ncifcrf.gov/~nicka/run2.html — compare two protein structures from the protein databank.
SARF: http://www-lmmb.ncifcrf.gov/~nicka/prerun.html — download SARF2 program for structure comparison.
STAMP: http://barton.ebi.ac.uk/ download STAMP program for structure comparison.

References

1. Orengo, C. A., Swindells, M. B., Michie, A. D., Zvelebil, M. J., Driscoll, P. C., Waterfield, M. D., and Thornton, J. M. (1995) Structural similarity between the pleckstrin homology domain and verotoxin: the problem of measuring and evaluating structural similarity. *Protein Sci.* **4,** 1977–1983.
2. Colloc'h, N., Etchebest, C., Thoreau, E., Henrissat, B., and Mornon, J. P. (1993) Comparison of three algorithms for the assignment of secondary structure in proteins: the advantages of a consensus assignment. *Protein Eng.* **6,** 377–382
3. Frishman, D. and Argos, P. (1995) Knowledge-based protein secondary structure assignment. *Proteins: Struct. Funct. Genet.* **23,** 566–579.
4. Taylor, W. R. (1992) Patterns, predictions and problems, in *Patterns in Protein Sequence and Structure* (Taylor, W. R., ed.), Springer-Verlag, Berlin.
5. Sternberg, M. J. E., Hegyi, H., Islam, S. A., Luo, J., and Russell, R. B. (1995) Towards an intelligent system for the automatic assignment of domains in globular proteins. *Ismb* **3,** 376–383.
6. McLachlan, A. D. (1979) Gene duplications in the structural evolution of chymotrypsin. *J. Mol. Biol.* **128,** 49–79.
7. Holm, L. and Sander, C. (1994) Parser for protein folding units. *Proteins: Struct. Funct. Genet.* **19,** 256–268.
8. Siddiqui, A. S. and Barton, G. J. (1995) Continuous and discontinuous domains: an algorithm for the automatic generation of reliable protein domain definitions. *Protein Sci.* **4,** 872–884.
9. Sowdhamini, R. and Blundell, T. L. (1995) An automated method involving cluster analysis of secondary structures for the identification of domains in proteins. *Protein Sci.* **4,** 506–520.
10. Islam, S. A., Luo, J., and Sternberg, M. J. E. (1995) Identification and analysis of domains in proteins. *Protein Eng.* **8,** 513–525.
11. Swindells, M. B. (1995) A procedure for detecting structural domains in proteins, *Protein Sci.* **4,** 103–112
12. Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., and Thornton, J. M. (1997) CATH — a hierarchic classification of protein domain structures. *Structure* **5,** 1093–1108.
13. Jones S., Stewart M., Michie A., Swindells M. B., Orengo C., and Thornton, J. M. (1998) Domain assignment for protein structures using a consensus approach: characterization and analysis. *Protein Sci.* **7,** 233–242.
14. Holm, L. and Sander, C. (1998) Dictionary of recurrent domains in protein structures. *Proteins: Struct. Funct. Genet.* **33,** ???–???.
15. Russell, R. B. (1994) Domain insertion. *Protein Eng*. **7,** 1407–1410.
16. Levitt, M. and Chothia, C. (1976) Structural patterns in globular proteins. *Nature* **261,** 552–558.
17. Murzin, A. G., Brenner, S. E., Hubbard, T. J., and Chothia, C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247,** 536–540

18. Holm, L. and Sander, C. (1997) Mapping the protein universe. *Science* **273,** 595–602.
19. Holm, L. and Sander, C. (1997) An evolutionary treasure: unification of a broad set of amiodhydrolases related to urease. *Proteins: Struct. Funct. Genet.* **28,** 72–82.
20. Artymiuk, P. J., Poirette, A. R., Rice, D. W., and Willett, P. (1997) A polymerase I palm domain in adenylyl cyclase*? Nature* **388,** 33–34.
21. Orengo, C. A. (1994) Classification of protein folds. *Curr. Opin. Struct. Biol.* **4,** 429–440.
22. Holm, L. and Sander, C. (1994) Searching protein structure databases has come of age. *Proteins: Struct. Funct. Genet.* **19,** 165–183.
23. Holm, L. and Sander, C. (1997) New structure — novel fold? *Structure* **5,** 165–171.
24. Taylor, W. R. and Orengo, C. A. (1989) Protein structure alignment. *J. Mol. Biol.* **208,** 1–22.
25. Orengo, C. A., Brown, N. P., and Taylor W. R. (1992) Fast structure alignment for protein databank searching. *Proteins: Struct. Funct. Gene*t. **14,** 139–167
26. Alexandrov, N. N., Takahashi, K., and Go, N. (1992) Common spatial arrangements of backbone fragments in homologous and non-homologous proteins. *J. Mol. Biol.* **225,** 5–9.
27. Russell, R. B. and Barton, G. J. (1992) Multiple sequence alignment from tertiary structure comparison. Assignment of global and residue confidence levels. *Proteins: Struct. Funct. Gene*t. **14,** 309–323.
28. Holm, L. and Sander, C. (1993) Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.* **233,** 123–138.
29. Holm, L. and Sander, C. (1994) The FSSP database of structurally aligned protein fold families. *Nucleic Acids Res*. **22,** 3600–3609. *See* also *Nucleic Acids Res.* **24,** 206–210.
30. Mitchell, E. M., Artymiuk, P. J., Rice, D. W., and Willett, P. (1990) Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.* **212,** 151–166.
31. Quanta. Molecular Simulations, San Diego, CA.
32. Gibrat, J.-F., Madej, T., Bryant, S. H. (1996) Surprising similarities in structure comparison. *Curr. Opin. Struct. Biol.* **6,** 377–385.
33. Kleywegt, G. J. and Jones, T. A. (1997) Detecting folding motifs and similarities in protein structures. *Methods Enzymol.* **277,** 525–545.
34. Russell, R. B. and Ponting, C. P. (1998) Protein fold irregularities that hinder sequence analysis. *Curr. Opin. Struct. Biol.* **8,** 364.
35. Murzin, A. G. (1998) Probable circular permutation in the flavin-binding domain. *Nat. Struct. Biol.* **5,** 101.
36. Liepinsh, E., Kitamura, M., Murakami, T., Nakaya, T., and Otting, G. (1997) Pathway of chymotrypsin evolution suggested by the structure of the FMN-binding protein from *Desulfovibrio vulgaris*. *Nat. Struct. Biol.* **4,** 975–979.
37. Chothia, C. (1992) One thousand families for the molecular biologist. *Nature* **357,** 543–544.

38. Orengo, C. A., Jones, D. T., and Thornton, J. M. (1994) Protein superfamilies and domain superfolds. *Nature* **372,** 631–634.
39. Blundell, T. L. and Johnson, M. S. (1993) Catching the common fold. *Protein Sci.* **2,** 877–883.
40. Crippen, G. M. and Mariov, V. (1995) How many protein folding motifs are there? *J. Mol. Biol.* **252,** 144–151.
41. Russell, R. B., Saqi, M. A. S., Sayle, R. A., Bates, P. A., and Sternberg, M. J. E. (1997) Recognition of analogous and homologous protein folds. Analysis of sequence and structure conservation. *J. Mol. Biol.* **269,** 423–439.
42. Jones, D. T. (1997) Progress in protein structure prediction. *Curr. Opin. Struct. Biol.* **7,** 377–387.
43. Murzin, A. G. (1993a) Sweet tasting protein monellin is related to the cystatin family of thiol proteinase inhibitors. *J. Mol. Biol.* **230,** 689–694.
44. Russell, R. B. and Barton, G. J. (1994) Structural features can be unconserved in proteins with similar folds. An analysis of side-chain to side-chain contacts, secondary structure and accessibility. *J. Mol. Biol.* **244,** 332–350.
45. Holm, L. and Sander, C. (1997) Decision support system for the evolutionary classification of protein structures. *Intel. Syst. Mol. Biol.* **5,** 140–146.
46. Swindells, M. B. (1993) Classification of doubly wound nucleotide binding topologies using automated loop searches. *Protein Sci.* **2,** 2146–2153.
47. Murzin, A. G. (1995) A ribosomal protein module in EF-G and DNA gyrase, *Nat. Struct. Biol.* **2,** 25–26.
48. Holm, L. and Sander, C. (1995) DNA polymerase β belongs to an ancient nucleotidyltransferase superfamily. *Trends Biochem. Sci*. **20,** 345–347.
49. Holm, L. and Sander, C. (1997) An evolutionary treasure: unification of a broad set of amidohydrolases related to urease. *Proteins: Struct. Funct. Gene*t. **28,** 72–82.
50. Holm, L. and Sander, C. (1997) Enzyme HIT. *Trends Biochem. Sci*. **22,** 116.
51. Park J., Teichmann S. A., Hubbard T., and Chothia C. (1997) Intermediate sequences increase the detection of homology between sequences. *J. Mol. Biol.* **273,** 349–354.
52. Russell, R. B., Saseini, P. D., and Sternberg, M. J. E. (1998) Supersites within superfolds. Binding site similarity in the absence of homology. *J. Mol Biol.* **28,** 903.
53. Hol, W. G., van Duijnen, P. T., and Berendsen, H. J. (1978) The alpha-helix dipole and the properties of proteins. *Nature* **273**. 443–446.
54. Pennisi, P. (1998) Taking a structured approach to understanding proteins. *Science* **279,** 978–979.
55. Vrielink, A., Ruger, W., Driessen, H. P., Freemont, P. S. (1994) Crystal structure of the DNA modifying enzyme beta-glucosyltransferase in the presence and absence of the substrate uridine diphosphoglucose. *EMBO J.* **13,** 3413–3422
56. Holm, L. and Sander, C. (1995) Evolutionary link between glycogen phosphorylase and a DNA modifying enzyme. *EMBO J.* **14,** 1287–1293.
57. Artymiuk, P. J., Rice, D. W., Poirette, A. R., and Willett, P. (1995) β-Glucosyltransferase and phophorylase reveal their common theme, *Nat. Struct. Biol.* **2,** 117–120.

58. Zhang, G., Liu, Y., Ruoho, A. E., and Hurley, J. H. (1997) Structure of the adenylyl cyclase catalytic core. *Nature* **386,** 247–253.
59. Tesmer, J. J., Sunahara, R. K., Gilman, A. G., and Sprang, S. R. (1997) Crystal structure of the catalytic domains of adenylyl cyclase in a complex with Gsα.GTPγS. *Science* **278,** 1907–1916.
60. Laskowski, R. A., Hutchinson, E. G., Michie, A. D., Wallace, A. C., Jones, M. L., and Thornton, J. M. (19??) PDBsum: a Web-based database of summaries and analyses of all PDB structures. *Trends Biochem. Sci.* **22,** 488–490.
61. Wallace, A. C., Laskowski, R. A., and Thornton, J. M. (1995) LIGPLOT: a program to generate schematic diagrams of protein-ligand interactions. *Protein Eng.* **8**,127–134.
62. Schuler, G. D., Epstein, J. A., Ohkawa, H., and Kans, J. A. (1996) Entrez: molecular biology database and retrieval system *Methods Enzymol.* **266**, 141–162.
63. Bryant, S. H., Madej, T., Janin, J., Liu, Y., Ruoho, A. E., Zhang, G., and Hurley, J. H. (1997) A polymerase I palm in adenylyl cyclase? *Nature* **388,** 34.
64. Murzin, A. G. (1993b) Can homologous proteins evolve different enzymatic activities? *Trends Biochem. Sci*. **18,** 403–405.
65. Murzin, A. G. (1996) Structural classification of proteins: new superfamilies. *Curr. Opin. Struct. Biol.* **6,** 386–394.
66. Farber, G. K. and Petsko, G. A. (19??) The evolution of a/b barrel enzymes. *Trends Biochem Sci* **15,** 228–234.
67. Sayle, R. A. and Milner-White, E. J. (1995) RASMOL Biomolecular Graphics for all. *Trends Biochem. Sci.* **20,** 374.
68. Kraulis, P. J. (1991) Molscript: a program to produced detailed and schematic plots of protein structures. *J. Appl. Cryst.* **24,** 946–950.
69. Bernstein, F. C., Koetzle, T. F., Williams, G. J., Meyer, E. E., Brice, M. D., Rodgers, M. D., Kennard, O., Shimanouchi, T., and Tasumi, M. (1977) The Protein Data Bank: a computer-based archival file for macrormolecular structures. *J. Mol. Biol.* **112,** 535–542.

14

Modeling Transmembrane Helix Bundles by Restrained MD Simulations

Mark S. P. Sansom and Leo Davison

1. Introduction

Integral membrane proteins are a major challenge for protein–structure prediction. It is estimated that about a third of genes code for membrane proteins ***(1)***, and yet high-resolution structures are known for only a handful of these. Furthermore, technical problems of protein expression and crystallization suggest that an explosive expansion in the number of membrane–protein–structure determinations is still in the future. In this chapter, attention is restricted to the major class of membrane proteins, i.e., those formed by bundles of transmembrane (TM) α-helices. Prediction methods also exist for those membrane proteins (e.g., porins and some bacterial toxins) that are formed by β-barrels (Kay Diederichs, personal communication; also *see* website: `http://loop.biologie.uni-konstanz.de/~kay/om_topo_predict2.html`). However, these methods are not applicable to the majority of membrane proteins and so are not discussed here.

One approach to modeling TM domains is based on two-stage ***(2)*** membrane protein folding (*see* **Fig. 1**). In the first stage of folding, TM regions are inserted into the membrane and form α-helices as they are inserted. Such TM helices lie approximately perpendicular to the plane of the lipid bilayer. The TM helices then aggregate within the plane of the membrane during the second stage of folding to form a TM helix bundle, which is the membrane-spanning domain of the protein. Mimicking this model of the folding process, structure prediction may also proceed via two main stages: (1) prediction of TM secondary-structure and topology, i.e., of the location of helices within the sequence and of their orientation (up/down) relative to the bilayer plane; followed by (2) prediction of how the TM helices pack together within the bilayer plane to form a TM helix bundle.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

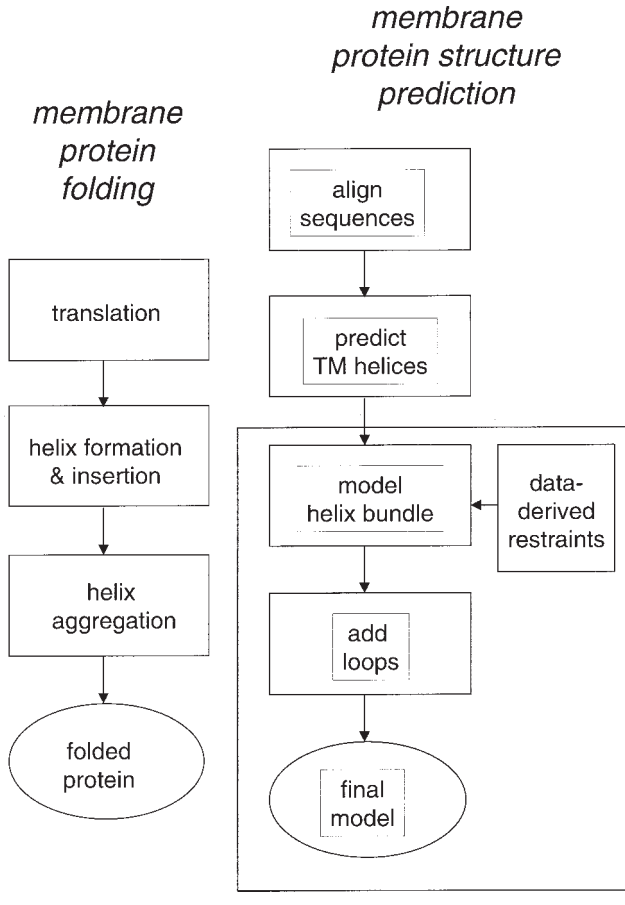


Fig. 1. Flow diagrams of two-stage folding model (left-hand side) vs modeling process (right-hand side) for a TM helix bundle. The gray box encompasses those stages of modeling that are the main topics of this chapter.

One problem with prediction of membrane protein structures is the small number (approx 20) of high-resolution structures that are known. This seriously impedes development of rules or empirical potentials to predict the packing of TM helices. Thus, *ab initio* prediction of TM helix packing in the absence of additional experimental data remains difficult in all but the simplest cases. However, restraints on possible modes of TM helix packing within a given membrane protein (or family of membrane proteins) may be obtained, either from analysis of multiply aligned sequences and/or from analysis of experimental protein chemistry and mutagenesis data. Furthermore, advances in electron microscopy (EM) mean that for an increasing number of membrane

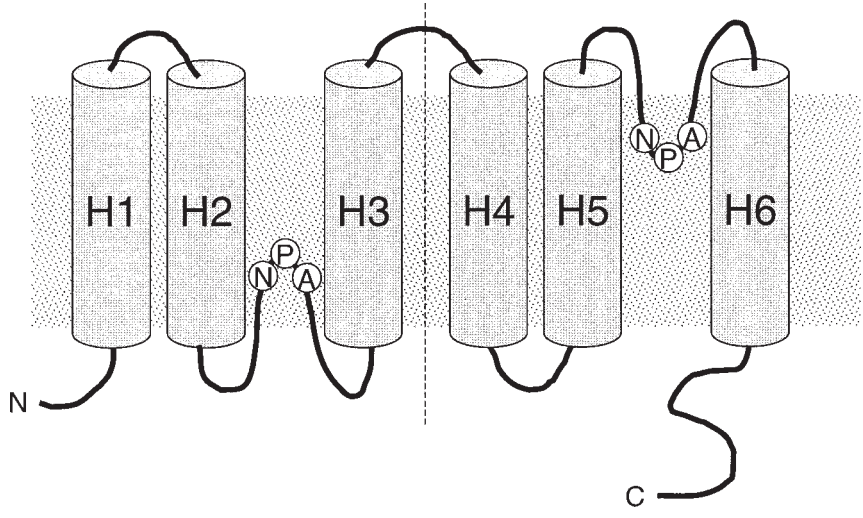


Fig. 2. Proposed topology of Aqp (*see* **ref. *6***). The six predicted TM helices (H1–H6) are shown superimposed on the bilayer (gray, stippled). The two loops containing the NPA motifs are shown folded back into the bilayer region.

proteins, low-resolution (9–6 Å) structures are available, which can provide restraints on the positions and orientations of TM helices. A number of computational techniques may be used to model packing of TM helices subject to such restraints. In this chapter we describe a simulated annealing/molecular dynamics (SA/MD) procedure, which is relatively simple to implement using standard modeling/simulation packages such as Xplor ***(3)***.

At this stage it is important to stress the role of prediction of TM helix bundles. In other than the simplest cases, such methods are unlikely to yield a unique model of a given membrane protein. Instead, they yield a number of alternative possible models, which differ from one another to some extent depending on the nature and number of restraints employed. Such models are not definitive structures. However, they do integrate available structural data and extend the effective resolution of such data. An important role of TM helix bundle models is to provide a conceptual framework facilitating design of further experiments to probe the structure–function relationships of membrane proteins.

As an illustration of the methods to be described, we consider modeling the TM helix bundle of an aquaporin (Aqp). The aquaporins are a family of integral membrane proteins that transport water across cell membranes ***(4,5)***. Secondary structure and topology prediction studies, supported by a range of experimental data ***(6–8)***, suggest that the Aqp monomer consists of six TM helices (*see* **Fig. 2**). Recently this has been confirmed by low-resolution (6 Å)

EM images of two-dimensional (2D) crystals of Aqp ***(9–11)***. In combination with sequence analysis, such EM images provide powerful restraints on the packing of the six TM helices within Aqp models.

2. Sequence Analysis

The first stage of modeling a TM helix bundle is to predict the positions within the sequence of the TM helices. This may be achieved using a single protein sequence. However, it seems that the accuracy of secondary structure prediction is improved if multiple aligned sequences for a number of related membrane proteins are used. Furthermore, alignment of multiple sequences is essential if sequence periodicity analysis (*see* below) is to be employed to derive restraints for helix bundle modeling. Standard alignment techniques may be readily applied to membrane proteins ***(12)***. Note that one may wish to pass through sequence alignment and TM helix prediction twice, in that it is advisable (and physically reasonable) to use higher gap penalties within predicted TM helices than in the interhelical loops. Methods for prediction of secondary structure are covered elsewhere in this volume ***(13)*** and so is not discussed in any detail here. Several secondary structure prediction methods are available for membrane proteins, mainly as Web-based tools (*see* **Table 1**). Because of the small number of three-dimensional (3D) structures for integral membrane proteins it is difficult to be certain of the absolute and relative accuracies of these methods. However, it is unlikely that prediction of TM secondary structure is likely to be less than that of secondary structure prediction in general ***(14)***, and so an accuracy of approx 80% may be reasonably assumed. As with all such predictions, a problem for subsequent modeling is that the "ends" of the helices are predicted less accurately. Comparisons of different prediction methods applied to the same sequence, and of related sequences analyzed using the same method reveal substantial variability in this respect.

We illustrate TM helix prediction with the analysis of human Aqp1 using the methods listed in **Table 1**. They vary in the number of TM helices predicted between five (PHDHTM) and seven (MEMSAT and TOPPRED2). However, the consensus appears to be six TM helices. The major discrepancies cluster around the first NPA sequence motif, which may correspond with the suggestion that this forms a loop that folds back into the membrane. From the EM images (*see* below; ***[9,10]***), there appear to be six TM helices. Thus, this example shows that even though the overall success rate for TM helix prediction is approx 85%, problems may occur for an individual protein. Furthermore, even though the PHDtopology, DAS, and TMAP methods all predict six TM helices, they disagree with respect to the start/end positions of those helices in the sequence. In particular, PHDtopology seems to predict rather shorter TM helices than some of the other methods (*see* **Fig. 3**).

Table 1
TM Secondary Structure and Topology Prediction Methods

Program	Method	Website	Refs.
MEMSAT	Statistical tables plus expectation maximization	http://globin.bio.warwick.ac.uk/~jones/memsat.html	*(55)*
TMAP	Multiple sequence alignments	http://www.embl-heidelberg.de/tmap/tmap-info.html	*(56,57)*
PHDHMT	Neural network	http://www.embl-heidelberg.de/predictprotein	*(58)*
PHDTOPOLOGY	Neural network plus dynamic programming	http://www.embl-heidelberg.de/predictprotein	*(51)*
TOPPRED2	Hydrophobicity analysis plus the "positive inside" rule	http://www.biokemi.su.se/~server/toppred2	*(59,60)*
DAS	Dense alignment surface	http://www.biokemi.su.se/~server/DAS	*(61)*

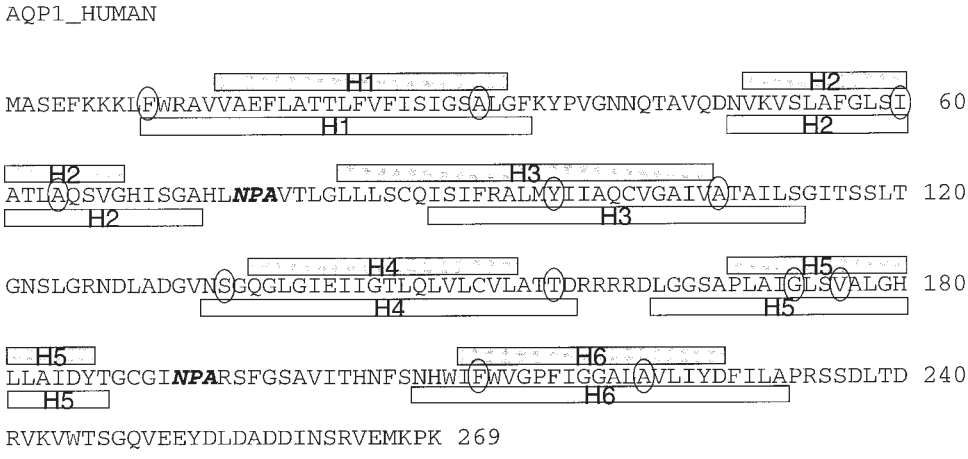


Fig. 3. Topology prediction for human Aqp1. The six TM helices predicted by PHDtopology (gray boxes above the sequence) and by a combination of TMAP and MEMSAT (white boxes below sequence) are shown superimposed on the human Aqp1 sequence. The NPA motifs are highlighted in ***bold italics***. For each helix, residues defining the conserved, inward-facing surface are indicated by a surrounding ellipse (0; *see* also **Table 3**).

3. Restraints for Modeling

Before attempting to pack together the predicted TM helices, restraints on their positions and orientations are needed. These may be divided into: (1) knowledge-based restraints, (2) restraints derived from sequence analysis, and (3) restraints based on experimental data. Such restraints may be used to determine starting configuration(s) for helix-packing simulations, and restrict interhelix motions in such simulations.

In principle, restraints derived from statistical analyzes of known structures of integral membrane proteins may prove to be the most valuable for guiding helix-packing simulations. However, the paucity of membrane protein structures means that, at present, such restraints are relatively weak. For example, an analysis of 45 TM helices (making 88 helix-packing interactions) in *three* independent membrane–protein structures ***(15)*** has revealed that TM helices pack against their neighbors in the sequence. This considerably reduces the number of possible packing arrangements that need to be considered. It would also seem reasonable that, on grounds of "compactness," the number of TM helix–helix interactions should be maximized. However, the latter consideration is difficult to be certain of on theoretical grounds, and may be biased by the classes of membrane protein for which crystallographic structures have been determined.

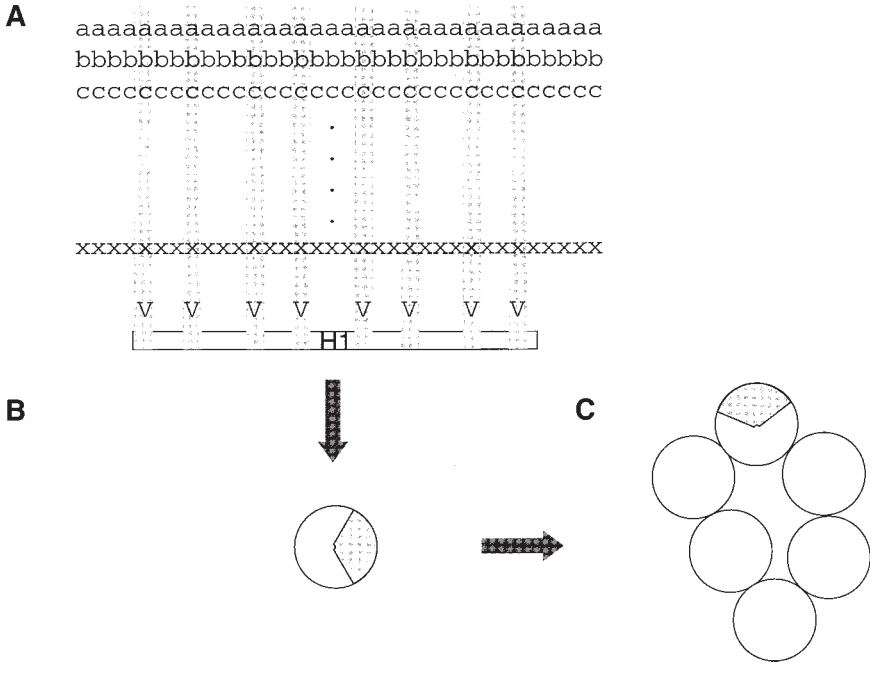


Fig. 4. Periodicity analysis of aligned multiple sequence of TM helices. **(A)** The aligned sequences for a predicted TM helix. The vertical gray stripes represent the most variable (V) residue positions in the helix. **(B)** The helix seen end on, with the gray sector representing the variable face of the helix. **(C)** The same helix as part of a TM helix bundle, with the variable (gray) face of the helix pointing out into the surrounding hydrophobic environment made up of lipid acyl tails.

The second class of restraints, based on sequence analysis, provide information on relative helix orientations within a bundle. A number of analyses ***(16,17)*** of multiply aligned sequences of those membrane proteins for which a 3D structures is known suggest that residues within the interior of the helix bundle are more highly conserved than those of the exterior (lipid-facing) surfaces of the helices. This is seen as a periodicity in conservation/variability of aligned residues along the length of a predicted TM helix (*see* **Fig. 4**). Such periodicity can be detected via Fourier analysis as in PERSCAN ***(17)***. Application of such methods to multiply aligned sequences of homologous TM helices for a family of membrane proteins can provide powerful restraints on which face of a predicted helix should point out toward the surrounding lipid environment and which face should point in toward the center of the helix bundle.

The final class of restraints arises from experimental data on a given membrane protein. These may be subdivided into “hard” restraints, derived from

low-resolution structural data (EM and solid-state nuclear magnetic resonance [NMR]), and "soft" restraints, derived from e.g., chemical labeling and site-directed mutagenesis experiments. EM data, typically at 9–6 Å resolution, can restrain the positions of the centers and long axes of TM helices. Solid-state NMR approaches ***(18,19)***, although in their infancy, can provide distance restraints between pairs of atoms, located either within the same TM helix (thus reinforcing secondary-structure predictions) or in different TM helices, thus restraining possible packing interactions for a helix pair. Site-directed mutagenesis, in particular, the substitution of a selected residue by cysteine, may be combined with protein chemistry. For example, a cysteine may be reacted with probe reagents or labeled with spectroscopic reporter groups, providing information on lipid-exposed residues of helices, e.g. Alternatively, pairs of cysteine residues may be introduced, and patterns of disulphide-bridge formation analyzed in terms of helix–helix interactions present within the intact protein.

For a number of intensively studied membrane proteins, e.g., rhodopsin, a plethora of data exist. In this case, the interactions of the TM helices are quite well defined ***(20)*** and restrained Monte Carlo methods may be used to obtain a unique (or near unique — *see* below) model of the packing of the TM helices. However, for most membrane proteins there are considerably fewer experimental data, and so the resultant models are inevitably rather less precise.

We now consider the restraints available for Aqp1. As discussed earlier, secondary structure prediction studies and experimental topology data ***(6,7)*** suggest six TM helices, with their N- and C-termini on the intracellular face of the membrane. This is supported by Fourier transform infrared spectroscopic studies, which suggest a high α-helical content, with the helices approximately perpendicular to the bilayer plane (and hence membrane spanning), but with an average tilt of 20–25° relative to the bilayer normal ***(21)***. Periodicity analysis of multiply aligned sequences for the predicted TM helices yields a surprisingly clear-cut assignment of interior and exterior surfaces for each of the six helices (***[22]***; *see* **Fig. 3**). Unfortunately, there are not many protein chemistry–mutagenesis data that place restraints on Aqp models, other than the suggestion that the two NPA-containing loops are involved in the water-transport mechanism and may be inserted "back into" the membrane. Significantly, low-resolution EM images provide powerful restraints on the positions and orientations of the six TM helices. Taken together, the various restraints are sufficient to enable prediction of a relatively small number of possible models for the Aqp1 TM helix bundle.

4. Restrained MD Simulations

The following sections provide a description of the use of simulated annealing (SA)/molecular dynamics (MD) (implemented using Xplor ***[3]***) for gener-

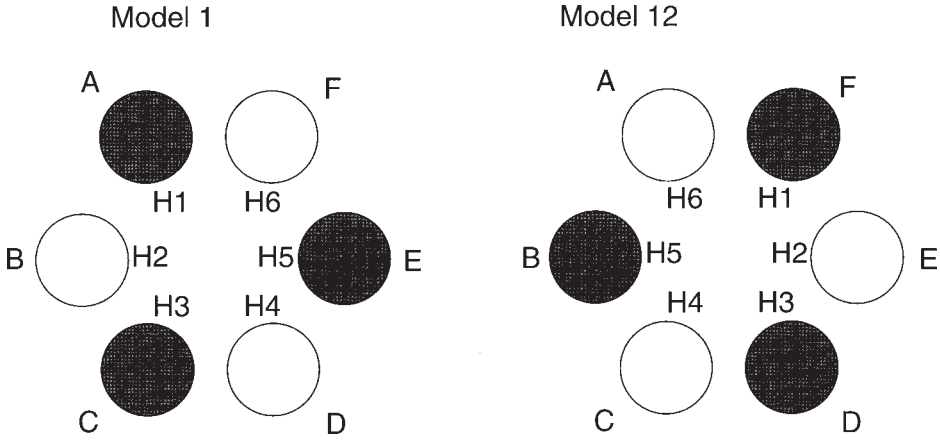


Fig. 5. Possible C_α templates for Aqp models. The circles represent the initial positions of the idealized α-helices. Filled circles have their C-terminus closest to the viewer, empty circles their N-terminus. The templates for Models-1 and -12 (*see* **Table 2**) are shown.

ating TM helix bundle models. This is illustrated via application to modeling a six-TM helix bundle for Aqp1, using the restraints described earlier.

4.1. C_α Templates

The first stage of SA/MD is to define a C_α template. This provides an initial model of the TM helix bundle as a set of idealized helices made up of just C_α atoms. The starting positions and orientations of these idealized helices embody the initial assumptions and the restraints on the models. For a simple helix bundle, e.g., pores formed by symmetrical assemblies of *N* identical TM helices, then an initial C_α template is straightforward to devise, provided that rotational symmetry about the central pore axis is assumed. If, as is the case for Aqp-1, a low-resolution EM structure is available, this may be used to define the C_α template. However, if one is interested in a complex membrane protein (e.g., a transporter molecule with 12 TM helices) and no EM images are available, then it may be necessary to use a Monte Carlo search method ***(23,24)*** to generate a family of possible C_α templates that are compatible with the restraints.

For Aqp1, the C_α template was generated as follows. From the EM images it is evident that the six helices lie at the corners of an irregular hexagon. Thus, in the C_α template, six idealized helices were positioned on a regular hexagon with an interaxial separation of 9.4 Å between adjacent helices (*see* **Fig. 5**). The helices were oriented such that their residue conserved faces (*see* above

Table 2
C_α Templates for Aqp Models

Model	A	B	C	D	E	F	
1	H1	H2	H3	H4	H5	H6	Clockwise
2	H6	H1	H2	H3	H4	H5	Clockwise
3	H5	H6	H1	H2	H3	H4	Clockwise
4	H4	H5	H6	H1	H2	H3	Clockwise
5	H3	H4	H5	H6	H1	H2	Clockwise
6	H2	H3	H4	H5	H6	H1	Clockwise
7	H1	H6	H5	H4	H3	H2	Counterclockwise
8	H2	H1	H6	H5	H4	H3	Counterclockwise
9	H3	H2	H1	H6	H5	H4	Counterclockwise
10	H4	H3	H2	H1	H6	H5	Counterclockwise
11	H5	H4	H3	H2	H1	H6	Counterclockwise
12	H6	H5	H4	H3	H2	H1	Counterclockwise

The 12 possible C_α templates are defined in terms of which helix in the model (H1–H6) corresponds to which helix (A– F) in the EM images (*see* **Fig. 5**). The "clockwise" models have helices H1–H6 arranged in a clockwise manner when looking down on the model from extracellular toward intracellular. The boxed models have helices A, C, and E of the EM images with their N-termini inside the cell.

and **Fig. 3**) were pointing toward the center of the bundle. In all C_α templates the N- and C-termini were intracellular. Furthermore, in accordance with the analysis of, e.g., **ref. *15*** and with the EM images, C_α templates had the helices placed at the apices of the hexagon in either a clockwise or counterclockwise fashion (*see* **Fig. 5**), i.e., sequence-adjacent helices were spatially adjacent. Thus, 12 C_α templates were possible (*see* **Table 2**), and each of these was used as a starting model for generation of an ensemble of 25 structures by models using SA/MD.

4.2. Implementing Restraints

Three classes of restraint have been used: (1) *intra*helix distance restraints, to maintain α-helicity of the TM segments, (2) *inter*helix distance restraints, to maintain helix orientations identified by periodicity analysis, and (3) "target" restraints on helices, to maintain the positions and orientations seen in the EM images. All three classes of restraint may be implemented by adding terms to the potential energy function used in the MD simulations:

$$E = E_{\mathrm{COVALENT}} + E_{\mathrm{NONBONDED}} + E_{\mathrm{RESTRAINT}} \quad (1)$$

Restraint energies, $E_{\mathrm{RESTRAINT}}$, may take a variety of forms. However, in the current application distance restraints were used. Distance restraints may act

Table 3
Interhelix Restraints — Model 7

Helix *i*		Helix *j*		Distance (Å)
H1	F10	H4	T157	12.2
H2	I60	H5	V176	14.2
H3	Y97	H6	A223	12.9
H4	S135	H1	A32	12.2
H5	G173	H2	A64	14.2
H6	F212	H3	A108	12.9

Restraints were applied between the C_β atoms of listed pairs of residues, during stage 2 of SA/MD.

either between pairs of atoms within a model structure (as is the case for both the intrahelix and the interhelix distance restraints) or between a pair of corresponding atoms in a model structure and a "target" structure, where the latter represents, e.g., a low-resolution structure derived from EM images. For example, a distance restraint between atom *i* and atom *j* may take the form:

$$E_{\text{RESTRAINT}} = K(d_{ij} - d_{\text{TARGET}})^2 \quad (2)$$

where d_{ij} is the distance between the two restrained atoms and d_{TARGET} is the target distance for the restraint. The scale factor *K* balances the experimentally derived restraints and the remainder of the energy function.

Intrahelix distance restraints were used to maintain α-helical geometry. They were between the carbonyl O of residue *i* and the amide H of residue *i* + *4*. Target distances for these restraints were based α-helical H-bonding geometries observed in crystal structures of proteins ***(25)***. Interhelix distance restraints were between the C_β atoms of inward-facing residues (as defined by periodicity analysis — *see* above) of helices on opposite sides of the hexameric bundle. Such restraints oriented the sequence-conserved face of each helix toward the center of the bundle. Appropriate distances for such restraints (*see* **Table 3**) were derived in an interactive fashion via construction of preliminary models. A number of SA/MD studies suggested that the pattern of such restraints (i.e., the pairs of side chains that are restrained) was more important than the exact value of the d_{TARGET} employed.

"Target" restraints (*see* **Fig. 6**) were derived from low resolution EM images . These images provided approximate coordinates for 6 rods (labeled A–F, *see* **Fig. 5**) of (presumably α-helical) density, sectioned 7 Å above and below the midplane (z = 0, where the z-axis is perpendicular to the plane of the bilayer) of the membrane. For each of Models-1 to -12 (i.e., each C_α template, *see* **Table 2**) the midpoint of the C_α atoms of an N-terminal and C-terminal section

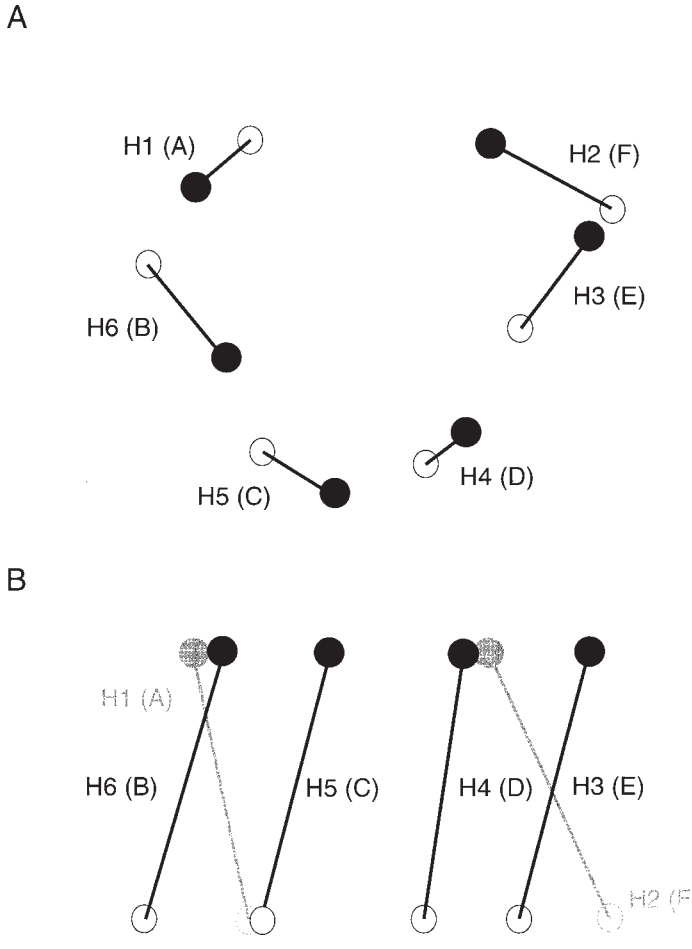


Fig. 6. Target restraints derived from EM images of Aqp1. Each rod of density in the EM image (A–F) is represented by the center of the density at $z = -7$ Å (open circles) and the center of the density at $z = +7$ Å (filled circles). The view in **A** is down a perpendicular to the bilayer, from outside toward inside the cell. The view in **B** is perpendicular to that in **A**, with the extracellular side of the membrane at the top of the diagram. The target restraints are shown for Model 7, for which density rod **A** corresponds to helix H1, rod **B** to H6, and so on, as indicated.

of each TM helix was restrained to minimize their distances from the corresponding sections of density. For example, for Model-1, the midpoint of the C_α atoms of residues 6 to 12 of H1 was restrained to the $z = -7$ Å section of rod A and the midpoint of the C_α atoms of residues 16–22 of H1 was restrained to the $z = +7$ Å section of rod A. In this way, helices H1–H6 were restrained to lie in

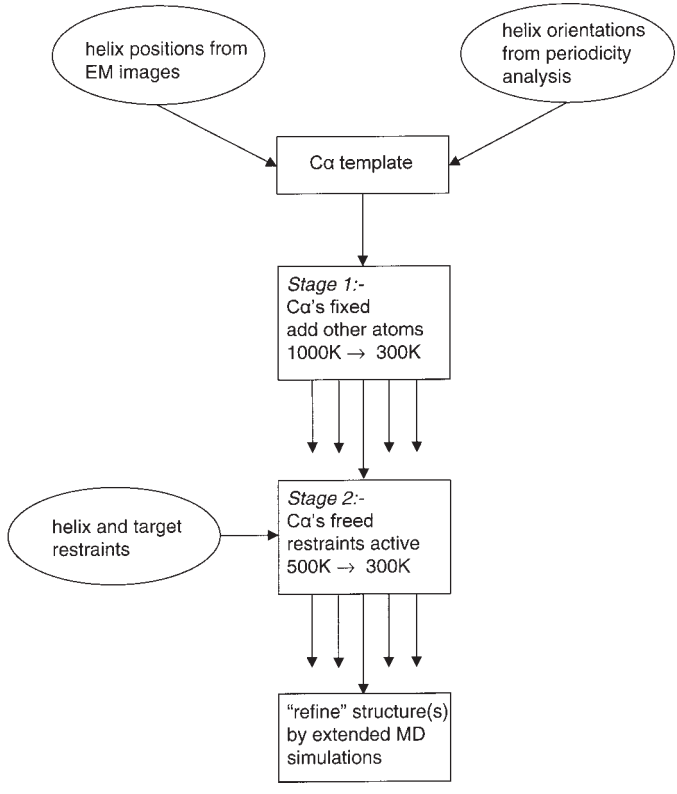


Fig. 7. Flow diagram of SA/MD. From each C_α template, stage 1 yields, e.g., 5 structures and stage 2 yields 5×5 structures.

the positions of rods A–F (with the correspondences as defined in **Table 2**) in each of the models 1–12.

4.3. Two Stage SA/MD Protocol

Having defined restraint terms to be added to a potential energy function, an MD simulation protocol is needed to search for helix-bundle geometries compatible with the restraints. Simulated annealing protocols ***(26)*** have been used with some success in protein modeling ***(27)***. For modeling TM helix bundles we have employed simulated annealing via restrained molecular dynamics (SA/MD; **Fig. 7**). This is based on methods for NMR structure determination ***(28)***. An early use of this method was the successful prediction of helix packing within the GCN4 leucine zipper helix dimer ***(29,30)***. For membrane proteins, a related approach has been to predict the structure of a homodimer of TM helices of glycophorin ***(31)***, and a homopentamer of TM helices of phospholamban ***(32)***.

In SA/MD the temperature of the simulation is used to control the sampling of different conformations. By starting at a high temperature and then progressively decreasing the temperature to 300 K, larger changes in conformation are possible at the start of the simulation, whereas toward the end of the simulation only much smaller changes occur. Application of SA/MD to ion channel models and related membrane proteins has been described in detail in several papers ***(33–39)***. The starting point of stage 1 of SA/MD is a C_α template (*see* above), which embodies underlying assumptions concerning the nature of the TM helix bundle. The other backbone and side-chain atoms are superimposed on the C_α atoms of the corresponding residue. These atoms "explode" from the C_α atoms, the positions of which remain fixed throughout stage 1. Annealing starts at 1000 K, during which weights for bond lengths and bond angles, and subsequently for planarity and chirality, are gradually increased. A repulsive van der Waals term is slowly introduced after an initial delay. Once the scale factors of these components of the empirical energy function reach their final values, the system is cooled from 1000–300 K, in steps of 10 K and 0.5 ps. During this cooling the van der Waals radii are reduced to 80% of their standard values in order to enable atoms to "pass by" one another. Electrostatic terms are *not* included during stage 1. Typically five structures are generated for each Cα template, corresponding to multiple runs of the process with different random number seeds.

Structures from stage 1 are each subjected to five molecular dynamics runs, e.g., (stage 2), resulting in an ensemble of $5 \times 5 = 25$ final structures from a single Cα template. Initial velocities are assigned corresponding to 500 K. The distance restraints are introduced at this point, and the Cα positional constraints are removed. Also during stage 2 electrostatic interactions are introduced into the potential energy function. On reaching 300 K, a 5-ps burst of constant temperature dynamics is performed, followed by 1000 steps of conjugate gradient energy minimization.

An important practical consideration is how to implement the foregoing methods. For SA/MD the program XPLOR ***(3)***, developed to implement MD simulations for X-ray and NMR determination of protein structures, is flexible and easy to use. However, there are a number of other MD programs (e.g., CHARMM ***[40]***, AMBER ***[41,42]***, GROMOS ***[43,44]***, and GROMACS ***[45]***), which could be used in principle.

The application of SA/MD to generate an ensemble of 25 structures for Model-1 of Aqp1 is illustrated in **Fig. 8**. Note the variation in structures within the final ensemble (*see* **Fig. 8C**). Analysis of such variation provides an indication of the extent to which the final structure is remains underdetermined by the restraints. To proceed further with modeling, it may be necessary to select a single structure from such an ensemble. It is difficult to define a "best"

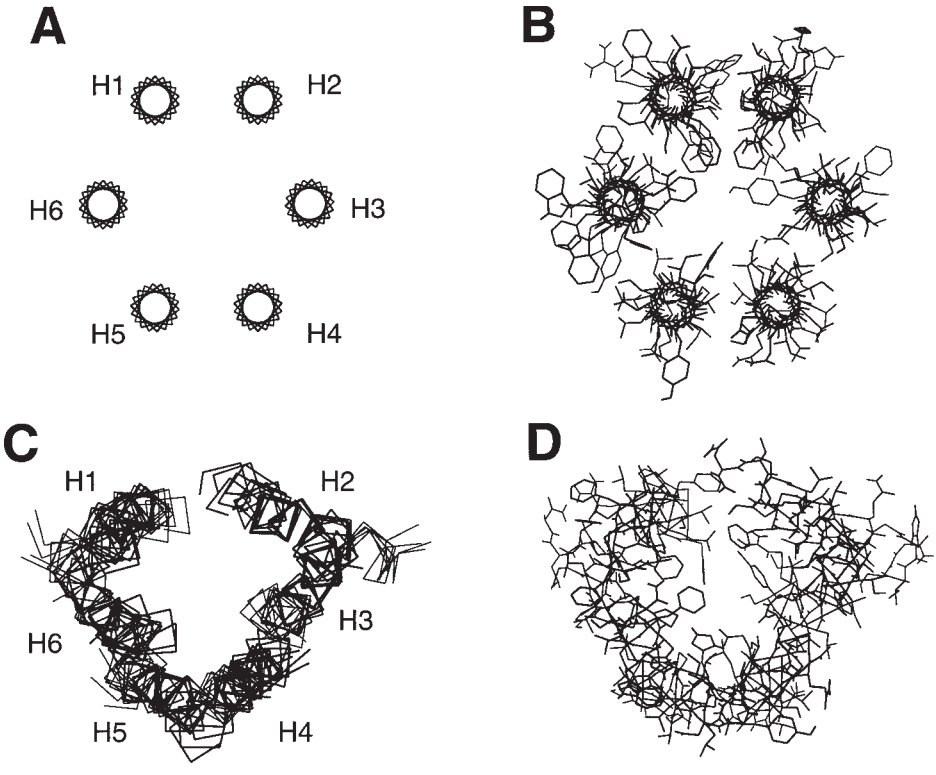


Fig. 8. Stages of SA/MD (illustrated for Model-7). **(A)** C_α template. **(B)** Structure from stage 1. **(C)** Superimposed C_α traces of 5 structures from the ensemble of 25 structures generated by stage 2. **(D)** — A single structure from the final ensemble. Diagrams were drawn using MOLSCRIPT ***(62)*** .

way in which to do this. One might take the structure that has the lowest root-mean-square deviation from the ensemble average structure. An alternative is to take the structure that best satisfies the distance restraints.

4.4. Refining Initial Models

The models described so far consist only of a TM helix bundle. However, for Aqp (and many other membrane proteins), mutagenesis and other data indicate that the interhelix loops may play important functional roles. It is therefore desirable to include at least a preliminary model of such interhelix loops. Other ways in which one might wish to refine TN helix bundle models include incorporation of water within transbilayer "pores," and embedding of models in a lipid bilayer.

Modeling surface loops of proteins is a nontrivial task. It is particularly difficult for membrane proteins because there are few experimental structures on which to base such models. However, a number of methods are available for modeling loops ***(46)*** and we briefly discuss one of them as applied to Model-7 of Aqp1. Examination of the predicted topology for Aqp1 (*see* **Fig. 3**) reveals that the loops between H1 and H2, and between H4 and H5 are relatively short. It therefore is feasible to model these. Furthermore, it has been suggested on the basis of site-directed mutagenesis and related studies ***(6,7)*** that the loops containing the NPA motifs fold back into the bilayer (the "hourglass" model) and probably contribute to the water-permeation pathway. Density in the EM images, approximately in the center of the helix bundle, has been tentatively identified with these NPA-loops ***(9,10)***. So, these two NPA-loops were also included in the model, using the EM suggestion of their locations as weak restraints. The loop between H3 and H4 is rather long and there are no restraints that may be applied. Therefore, it has not (yet) been incorporated into the Aqp1 model.

For the short interhelical loops (which are believed to lie on the surface), the only restraint is the (modeled) position of the two helix termini to which the loops are attached. For the reentrant NPA-loops a weak "target" restraint may be applied to take into account the suggestions from the EM images. No secondary structure (e.g., intrahelical) restraints were applied to the loops. Although one might restrain backbone dihedrals of loops to values from analysis of experimental structures or from conformational searching, this has not been done in the current study. The approach adopted was as follows. The C_α coordinates of one selected structure from each ensemble of models (i.e., one structure each for Models-1 to -12) was used as a new C_α template. C_α template coordinates for the H1–H2 and H4–H5 loops were calculated by taking evenly spaced points along a vector linking the C_α of the C-terminus of H1 (or H4) and the C_α of the N-terminus of H2 (or H5). Coordinates for each C_α atom of the loop were generated using a Gaussian distribution (standard deviation = 1 Å) centered on the corresponding point on the C-terminus-to-N-terminus vector. For the NPA-loops a similar procedure was used. Evenly spaced points along a V-shape projecting into the center of the helix bundle (with the apex of the V in the position approximately indicated by the EM images) were used. The new C_α template was then input to a further run of the SA/MD procedure. During stage 2, the NPA-containing loops were restrained to lie close to "target" coordinates derived from the published EM images ***(10)***. The H1–H2 and H4–H5 loops were not restrained.

An example of a model derived by this approach (Model-7) is shown in **Fig. 9**. It can be seen that the two NPA-loops are folded back into the center of the six helix bundle in a quasisymmetrical fashion, as suggested in the original

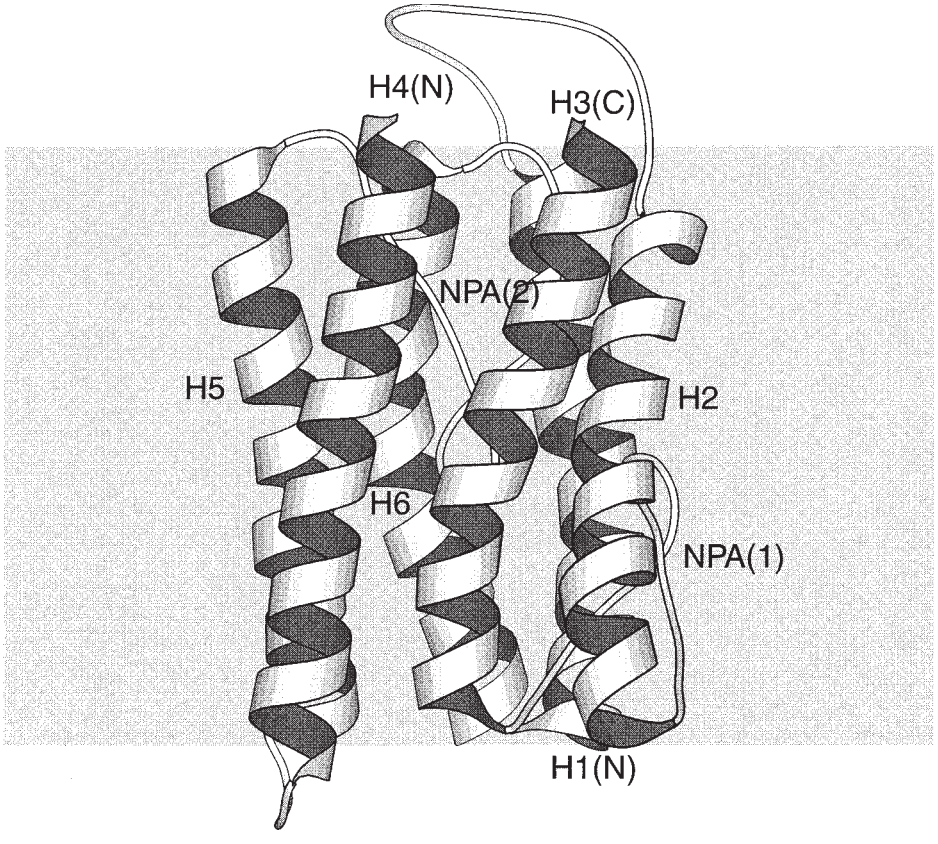


Fig. 9. Model-7 of the Aqp1 TM helix bundle, including the H1–H2, H4–H5, and NPA-containing loops. The view is perpendicular to the plane of the bilayer, with the extracellular face of the membrane at the top of the diagram.

"hourglass" model ***(6)***. Both NPA-loops adopt a turn conformation in their central region. The H1–H2 and H4–H5 loops lie on surface of the molecule. The shorter H4–H5 loop adopts a turn conformation, whereas the longer H1–H2 loop is rather more irregular. Note that loops were only added to 9 of the 12 models defined in **Table 2** (Models-1, -3, -4, -6, -7, -8, -9, -10, and -12). Examination of the other models (Models-2, -5, and -11) revealed that the NPA-loops could not be added to them in a manner compatible with the restraints. Thus, a total of $25 \times 9 = 225$ possible structures for the Aqp1 TM helix bundle plus loops were produced.

The problem remains of how to rank the nine models (and how to rank the 25 structures within each model ensemble). This is problematic. For example, mean-force pairwise residue potentials, which might be used to "score" mod-

els, are as yet not possible for integral membrane proteins. In the absence of a "structural" score for membrane protein models, all nine models must be presented as candidate structures for Aqp1, which may be used to aid design of further experiments. One may examine the models as to their possible functional implications. In particular, Aqp is believed to form a water-selective pore. Therefore, the dimensions of the pore running through the center of the (loop-containing) models are functionally relevant. Pore radius profiles may be evaluated using, e.g., HOLE ***(47,48)***. Such analysis suggested that Model-7 was a functionally plausible candidate for the Aqp1 structure. In particular, it had a continuous pore through the center of the molecule, with a minimum radius only a little less than that of a water molecule (1.6 Å).

Having identified a possible pore structure, it may be "refined" by more extended MD simulations. The SA/MD simulations were performed *in vacuo.* However, it is likely that complex, anisotropic environment (lipid bilayer plus water) may affect the conformations of interhelix loops, e.g. Two possible approaches may be taken to the refinement of the Aqp1 model by extended MD simulations. One is to add water molecules within the central pore and as approximately hemispherical caps at the two ends of the molecule and then run *in vacuo* MD simulations of combined the (pore + water) system. The alternative is to embed the Aqp1 model in a lipid bilayer, solvate with water within the pore and on either face of the bilayer, and then run nanosecond MD simulations of the (pore + water + bilayer) system. The latter is preferable, and is within the range of current computer power. However, this takes us away from prediction per se, in the direction of simulations based on model structures, and are not discussed further. The interested reader is referred to a number of reviews on this topic ***(49,50)***.

5. Notes

In this section we discuss some of problems of this approach to modeling TM helix bundles. The main difficulty is that of verifying this (or any) method, because of the lack of a sufficient number of high-resolution structures for integral membrane proteins. This situation will only improve as further structures are determined. However, at current rates of progress, it may be awhile before there is a sufficiently large database of membrane protein structures to permit development of knowledge-based approaches to prediction for these membrane proteins. In the meantime, perhaps the only way to proceed is to apply structure prediction methods to those membrane proteins for which low-resolution structural data are available, and to see how such models fare as higher-resolution structural data emerge.

Secondary structure prediction and topology predictions appear to be quite reliable for the TM helices ***(51)***. However, from the perspective of modeling

membrane protein folds, a problem lies is the imprecision of prediction of the termini of TM helices. It seems unlikely that purely sequence-based approaches will make further progress with this problem. A possible solution may lie in analysis of extended MD simulations of (TM helix + water + bilayer) systems ***(52,53)***, which will provide improved physicochemical insights into the factors driving TM regions to adopt a helical conformation. In particular, we need to know more about the conformational preferences of amino acids and in the "interfacial" region that lies between the hydrophobic bilayer core and the bulk water facing a bilayer.

A further problem in modeling TM helix bundles is the "softness" of the restraints employed to orient the helices relative to one another. Although periodicity analysis of aligned TM sequences may enable identification of the inner/outer faces of TM helices, one cannot be certain how strongly to apply a restraint derived from such analysis. This may improve as such analysis is applied to a greater number of membrane proteins. However, it is not only those restraints derived from sequence analysis that are "soft." By definition, *low-resolution* EM images do not provide accurate positions for helices. In terms of "hard" restraints, the best hope for the future may lie in solid-state NMR methods. Results from such studies ***(18,19)*** suggest that accurate estimates of distances between pairs of side-chain atoms may be obtained, which would greatly increase one's confidence in restrained SA/MD models.

The main problem with SA/MD as a technique lies in the complexity of setting up (multiple) C_α templates. As the number of TM helices increases (e.g., many transport proteins seem to contain approx 12 TM helices ***[54]***), so does the number of possible C_α templates. This presents the difficulty of having to generate large numbers of possible models, and of having to evaluate and rank even larger numbers of structures. Even though SA/MD is quite fast (e.g., running `Xplor` on a Silicon Graphics R1000K 195-MHz processor takes approx 25 min of computer time to generate a single Aqp structure), with a large number of possible C_α templates the size of the problem is daunting. A solution is to use the restraints to preselect more probable C_α templates using, e.g., a Monte Carlo search method that treats TM helices as rigid rods or cylinders ***(23,24)***. However, there is a problem with such an approach, i.e., that of "mirror images." If helices are approximated as rods or cylinders, then for a given set of distance restraints there will always be at least two optimum structures, which are mirror images of one another. Once an all-atom model is generated, then if the distance restraints are sufficiently exact, this problem should disappear. However, it is unlikely that even restraints from solid-state NMR data will fully resolve this problem. Possibly the only way in which one may confidently choose between two possible TM helix bundles related to one another by (approximate) mirror image symmetry is on the basis of EM images.

If such images are not available, than at least two possible models generated by MC searches will have to be converted to all atom models by SA/MD and subsequently refined by extended MD simulations as we discussed.

Acknowledgments

This work was supported by the Wellcome Trust. The authors thank Richard Law and Ian Kerr for the sequence analysis and secondary-structure predictions of Aqp1.

References

1. Walker, J. E. and Saraste, M. (1996) Membrane protein structure. *Curr. Opin. Struct. Biol.* **6,** 457–459.
2. Popot, J. L. and Engelman, D. M. (1990) Membrane protein folding and oligomerization: the two-state model. *Biochemistry* **29,** 4031–4037.
3. Brünger, A. T. (1992) *X-PLOR Version 3.1. A System for X-ray Crystallography and NMR*, Yale University Press, New Haven, CT.
4. Chrispeels, M. J. and Agre, P. (1994) Aquaporins: water channels in plant and animal cells. *Trends Biochem. Sci.* **19,** 421–425.
5. Engel, A., Walz, T., and Agre, P. (1994) The aquaporin family of membrane water channels. *Curr. Opin. Struct. Biol.* **4,** 545–553.
6. Jung, J. S., Preston, G. M., Smith, B. L., Guggino, W. B., and Agre, P. (1994) Molecular structure of the water channel through aquaporin CHIP: the hourglass model. *J. Biol. Chem.* **269,** 14,648–14,654.
7. Preston, G. M., Jung, J. S., Guggino, W. B. and Agre, P. (1994) Membrane topology of aquaporin CHIP: analysis of functional epitope-scanning mutants by vectorial proteolysis. *J. Biol. Chem.* **269,** 1668–1673.
8. Bai, L., Fushimi, K., Sasaki, S., and Marumo, F. (1996) Structure of aquaporin–2 vasopressin water channel. *J. Biol. Chem.* **271,** 5171–5176.
9. Walz, T., Hirai, T., Murata, K., Heymann, J. B., Smith, B. L., Agre, P., and Engel, A. (1997) The three-dimensional structure of aquaporin–1. *Nature* **387,** 624–627.
10. Cheng, A., van Hoek, A. N., Yeager, M., Verkman, A. S. and Mitra, A. K. (1997) Three-dimensional organization of a human water channel. *Nature* **387,** 627–630.
11. Heymann, J. B., Müller, D. J., Mitsuaoka, K. and Engel, A. (1997) Electron and atomic force microscopy of membrane proteins. *Curr. Opin. Struct. Biol.* **7,** 543–549.
12. Higgins, D. and Taylor, W. R. (2000) Multiple sequence alignment, in *Protein Structure Prediction: Methods and Protocols* (Webster, D. M. and Walker, J., eds.), Humana Press, Totowa, NJ, pp. 1; 1–18.
13. Rost, B. and Sander, C. (2000) Third generation of secondary structures, in *Protein Structure Prediction: Methods and Protocols* (Webster, D. and Walker, J., eds.), Humana Press, Totowa, NJ, pp. 5; 1–24.
14. Russell, R. B. and Barton, G. K. (1993) The limits of protein secondary structure prediction accuracy from multiple sequence alignment. *J. Mol. Biol.* **234,** 951–957.

15. Bowie, J. U. (1997) Helix packing in membrane proteins. *J. Mol. Biol.* **272,** 780–789.
16. Komiya, H., Yeates, T. O., Rees, D. C., Allen, J. P., and Feher, G. (1987) Structure of the reaction centre from *Rhodobacter sphaeroides* R-26 and 2.4.1: symmetry relations and sequence comparisons. *Proc. Natl. Acad. Sci. USA* **85,** 9012–9016.
17. Donnelly, D., Overington, J. P., Ruffle, S. V., Nugent, J. H. A., and Blundell, T. L. (1993) Modelling a-helical transmembrane domains: the calculation and use of substitution tables for lipid-facing residues. *Protein Sci.* **2,** 55–70.
18. Watts, A., Ulrich, A. S., and Middleton, D. A. (1995) Membrane-protein structure: the contribution and potential of novel solid-state NMR approaches. *Mol. Memb. Biol.* **12,** 233–246.
19. Smith, S. O., Ascheim, K., and Groesbeek, M. (1996) Magic angle spinning NMR spectroscopy of membrane proteins. *Q. Rev. Biophys.* **4,** 395–449.
20. Donnelly, D. and Findlay, J. B. C. (1994) Seven-helix receptors: structure and modelling. *Curr. Opin. Struct. Biol.* **4,** 582–589.
21. Cabiaux, V., Oberg, K. A., Pancoska, P., Walz, T., Agre, P., and Engel, A. (1997) Secondary structures comparison of aquaporin-1 and bacteriorhodopsin. A Fourier transform infrared spectroscopic study of two-dimensional membrane crystals. *Biophys. J.* **73,** 406–417.
22. Sansom, M. S. P., Kerr, I. D., Law, R., Davison, L., and Tielman, D. P. (1998) Modelling the packing of transmembrane helices application to aquaporin-1. *Biochem. Soc. Transac.* 26, 509–515.
23. Herzyk, P. and Hubbard, R. E. (1995) Automated-method for modeling 7-helix transmembrane receptors from experimental-data. *Biophys. J.* **69,** 2419–2442.
24. Son, H. S. and Sansom, M. S. P. (1996) Simulation of packing of transmembrane helices. *Biochem. Soc. Trans.* **24,** 140S.
25. Baker, E. N. and Hubbard, R. E. (1984) Hydrogen bonding in globular proteins. *Prog. Biophys. Mol. Biol.* **44,** 97–179.
26. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983) Optimization by simulated annealing. *Science* **220,** 671–680.
27. Chou, K.-C. and Carlacci, L. (1991) Simulated annealing approach to the study of protein structures. *Protein Eng.* **4,** 661–667.
28. Nilges, M., Clore, G. M., and Gronenborn, A. M. (1988) Determination of three-dimensional structures of proteins from interproton distance data by dynamical simulated annealing from a random array of atoms. Circumventing problems associated with folding. *FEBS Lett.* **239,** 129–136.
29. Nilges, M. and Brünger, A. T. (1991) Automated modelling of coiled coils: application to the GCN4 dimerization region. *Protein Eng.* **4,** 649–659.
30. Nilges, M. and Brünger, A. T. (1993) Successful prediction of the coiled coil geometry of the GCN4 leucine zipper domain by simulated annealing: comparison to the X ray structure. *Proteins: Struct. Func. Genet.* **15,** 133–146.
31. Treutlein, H. R., Lemmon, M. A., Engelman, D. M., and Brunger, A. T. (1992) The glycophorin A transmembrane domain dimer: sequence specific propensity for a right handed supercoil of helices. *Biochemistry* **31,** 12,726–12,733.

32. Adams, P. D., Arkin, I. T., Engelman, D. M., and Brünger, A. T. (1995) Computational searching and mutagenesis suggest a structure for the pentameric transmembrane domain of phospholamban. *Nat. Struct. Biol.* **2,** 154–162.
33. Kerr, I. D., Sankararamakrishnan, R., Smart, O. S., and Sansom, M. S. P. (1994) Parallel helix bundles and ion channels: molecular modelling via simulated annealing and restrained molecular dynamics. *Biophys. J.* **67,** 1501–1515.
34. Kerr, I. D., Doak, D. G., Sankararamakrishnan, R., Breed, J., and Sansom, M. S. P. (1996) Molecular modelling of Staphylococcal d-toxin ion channels by restrained molecular dynamics. *Protein Eng.* **9,** 161–171.
35. Sankararamakrishnan, R., Adcock, C., and Sansom, M. S. P. (1996) The pore domain of the nicotinic acetylcholine receptor: molecular modelling and electrostatics. *Biophys. J.* **71,** 1659–1671.
36. Sansom, M. S. P., Son, H. S., Sankararamakrishnan, R., Kerr, I. D., and Breed, J. (1995) Seven-helix bundles: molecular modelling via restrained molecular dynamics. *Biophys. J.* **68,** 1295–1310.
37. Sansom, M. S. P. and Kerr, I. D. (1995) Transbilayer pores formed by β-barrels: molecular modelling of pore structures and properties. *Biophys. J.* **69,** 1334–1343.
38. Sansom, M. S. P., Sankararamakrishnan, R., and Kerr, I. D. (1995) Modelling membrane proteins using structural restraints. *Nat. Struct. Biol.* **2,** 624–631.
39. Sansom, M. S. P., Kerr, I. D., Smith, G. R., and Son, H. S. (1997) The influenza A virus M2 channel: a molecular modelling and simulation study. *Virology* **233,** 163–173.
40. Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy, minimisation, and dynamics calculations. *J. Comp. Chem.* **4,** 187–217.
41. Weiner, S. J., Kollman, P. A., Case, D. A., Singh, U. C., Ghio, C., Alagona, G., Profeta, S., and Weiner, P. (1984) A new force field for molecular mechanical simulation of nucleic acids and proteins. *J. Am. Chem. Soc.* **106,** 765–784.
42. Pearlman, D. A., Case, D. A., Caldwell, J. W., Ross, W. S., Cheatham, T. E., Debolt, S., Ferguson, D., Seibel, G., and Kollman, P. (1995) Amber, a package of computer-programs for applying molecular mechanics, normal-mode analysis, molecular-dynamics and free-energy calculations to simulate the structural and energetic properties of molecules. *Comp. Phys. Commun.* **91,** 1–41.
43. Berendsen, H. J. C., Postma, J. P. M., van Gunsteren, W. F., DiNola, A., and Haak, J. R. (1984) Molecular dynamics with coupling to an external bath. *J. Chem. Phys.* **81,** 3684–3690.
44. Hermans, J., Berendsen, H. J. C., van Gunsteren, W. F., and Postma, J. P. M. (1984) A consistent empirical potential for water-protein interactions. *Biopolymers* **23,** 1513–1518.
45. Berendsen, H. J. C., van der Spoel, D., and van Drunen, R. (1995) GROMACS: A message-passing parallel molecular dynamics implementation. *Comp. Phys. Commun.* **95,** 43–56.
46. Bruccoleri, B. (2000) Ab initio loop modeling and its application to homology modeling, in *Protein Structure Prediction: Methods and Protocols* (Webster, D. and Walker, J., eds.), Humana Press, Totowa, NJ, pp. 11; 1–18.

47. Smart, O. S., Goodfellow, J. M., and Wallace, B. A. (1993) The pore dimensions of gramicidin A. *Biophys. J.* **65,** 2455–2460.
48. Smart, O. S., Breed, J., Smith, G. R., and Sansom, M. S. P. (1997) A novel method for structure-based prediction of ion channel conductance properties. *Biophys. J.* **72,** 1109–1126.
49. Sansom, M. S. P. (1998) Models and simulations of ion channels and related membrane proteins. *Curr. Opin. Struct. Biol.* **8,** 237–244.
50. Tieleman, D. P., Marrink, S. J., and Berendsen, H. J. C. (1997) A computer perspective of membranes: molecular dynamics studies of lipid bilayer systems. *Biochim. Biophys. Acta* **1331,** 235–270.
51. Rost, B., Fariselli, P., and Casadio, R. (1996) Topology prediction for helical transmembrane proteins at 86% accuracy. *Protein Sci.* **5,** 1704–1718.
52. Shen, L., Bassolino, D., and Stouch, T. (1997) Transmembrane helix structure, dynamics, and interactions: multi-nanosecond molecular dynamics simulations. *Biophys. J.* **73,** 3–20.
53. Woolf, T. B. (1997) Molecular dynamics of individual α-helices of bacteriorhodopsin in dimyristoyl phosphatidylcholine. I. Structure and dynamics. *Biophys. J.* **73,** 2376–2392.
54. Henderson, P. J. F. (1993) The 12-transmembrane helix transporters. *Curr. Opin. Cell Biol.* **5,** 708–721.
55. Jones, D. T., Taylor, W. R., and Thornton, J. M. (1994) A model recognition approach to the prediction of all-helical membrane protein structure and topology. *Biochemistry* **33,** 3038–3049.
56. Persson, B. and Argos, P. (1994) Prediction of transmembrane segments utilising multiple sequence alignments. *J. Mol. Biol.* **237,** 182–192.
57. Persson, B. and Argos, P. (1997) Prediction of membrane protein topology utilizing multiple sequence alignments. *J. Protein Chem.* **16,** 453–457.
58. Rost, B., Casadio, R., Fariselli, P., and Sander, C. (1995) Prediction of helical transmembrane segments at 95% accuracy. *Protein Sci.* **4,** 521–533.
69. von Heijne, G. V. (1992) Membrane protein structure prediction. Hydrophobicity analysis and the positive inside rule. *J. Mol. Biol.* **225,** 487–494.
60. Claros, M. G. and von Heijne, G. (1994) Toppred-II — an improved software for membrane-protein structure prediction. *Comput. Appl. Biosci.* **10,** 685–686.
61. Czerzo, M., Wallin, E., Simon, I., von Heijne, G., and Elofsson, A. (1997) Prediction of transmembrane alpha-helices in prokaryotic membrane proteins: the dense alignment surface method. *Protein Eng.* **10,** 673–676.
62. Kraulis, P. J. (1991) MOLSCRIPT: a program to produce both detailed and schematic plots of protein structures. *J. Appl. Cryst.* **24,** 946–950.

15

Predictive Models of Protein-Active Sites

D. Eric Walters

1. Introduction

For many proteins, we do not have enough information to even attempt to predict the three-dimensional (3D) structure. Many drug receptors, e.g., are membrane-bound proteins for which there is not a crystal structure of any homologous protein available. But drug receptors are proteins for which it would be especially useful to know the 3D structure. Fortunately for the drug design problem, it is often sufficient to be able to construct a reasonable *model* of the receptor protein's active site, and there are several ways in which we can construct such models. In this chapter we consider two approaches to constructing binding-site models. Both of these use a common starting point: a series of ligands for which binding (or other biological activity) has been measured. This structure–activity series serves as a template around which the active site model is built. The two methods differ only in the ways in which the binding site is represented (graphical surface or atoms).

2. Theory

There are several assumptions implicit in the foregoing approach:

1. All of the ligands are binding to a common site on the protein.
2. Biological activity is proportional to ligand–protein affinity.
3. All of the ligands bind in low-energy conformations (not necessarily the global minimum, but a reasonable local minimum).

3. Materials

The most important step in constructing a protein binding-site model is the selection and preparation of the structure–activity series. There are several guidelines to be followed:

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

1. There should be good reason to expect that all of the compounds in the structure–activity series act at the same site. If the measured bioactivity is a receptor-binding assay, there may be little or no doubt. If the measured activity is a response several steps downstream from the receptor binding event, this may be more difficult to prove. Suppose you are measuring smooth muscle contraction. Are all of the compounds acting as agonists at a single type of adrenergic receptor, or are multiple receptor types involved, or are some of the compounds blocking metabolism or reuptake of neurotransmitter that is already present?
2. It is important to have as much structural diversity as possible in the structure–activity series, so that much of the nature of the binding site can be explored. A parent structure with a simple series of methyl-ethyl-propyl-fluoro-chloro-bromo substitutions at a single site will only provide information about a limited region of the receptor site.
3. In order to get models that are quantitative in nature (e.g., able to make some sort of prediction about binding of new ligands), it is necessary to have ligands with a broad range of activities or affinities. This is, after all, an interpolation method.
4. All of the ligands must be placed in low-energy conformations. Conformational analysis is a complex topic ***(1,2)***. This means that you have a choice of many different approaches. It also means that no matter what method you choose, there will be those who think you did it wrong. Molecular mechanics force-field calculations are usually sufficiently accurate to identify local minima and to calculate *relative* conformational energies within a few kcal/mol. If you are dealing with relatively rigid molecules (only a few rotatable bonds), you may be able to systematically explore all of the accessible conformational space. For more flexible ligands (six or more rotatable torsions), there are other approaches (Monte Carlo sampling, molecular dynamics, buildup procedures, and others) that can be used. Modern molecular modeling software packages usually provide several different approaches to the conformational analysis problem, and it is up to you to select a method suitable for your particular data set. If you are dealing with a structurally diverse data set, you may be able to select one ligand with relatively few accessible conformers, then search for conformers of your more flexible ligands that can superimpose well onto your first ligand.
5. For each ligand in the structure–activity series, charge distribution must be calculated and partial atomic charges must be assigned to the individual atoms. This can be done using methods ranging from quick, approximate ones such as that of Gasteiger and Marsili ***(3)***, to very computationally intensive *ab initio* calculations with large basis sets. However, the receptor-modeling methods we are discussing are quite approximate and probably do not justify the use of *ab initio* calculations. We generally use semiempirical methods such as Stewart's Molecular Orbital Package (MOPAC) at the PM3 or AM1 level.
6. Finally, the ligands must be superimposed in a way that produces some common pattern of shape and charge distribution. There are programs that attempt to do this automatically and systematically. We have a great deal of respect for the ability of the human brain to recognize visual patterns, so we prefer to manually

superimpose structures on a computer graphic workstation. We select an initial compound on the basis of its high affinity (it should be one of the best-fitting ligands in the active site) or because it has few conformations to choose from (adapt the more flexible ligands to the shape of a less flexible one). Because electronic forces act over greater distances than van der Waals forces, we use space-filling representations with surfaces colored according to the electrostatic potential. We first match regions of positive or negative potential, then try to maximize steric overlap while maintaining electronic overlap.

4. Methods

The optimally superimposed set of ligands, in low-energy conformations, is the starting material for building models of protein binding sites.

4.1. Method 1: Computed Graphic Surfaces

The simpler (and perhaps more abstract) way to make a model of the binding site from our assembled ligands is to build a graphic surface over the superimposed ligands. Anthony Nicholls' Graphical Representation and Analysis of Surface Properties (GRASP) program ***(4)*** is particularly well suited to this task. Hahn and Rogers ***(5,6)*** have written specialized software to do this as well. Many other molecular modeling programs are also able to construct such a surface.

The next step is to color the surface on the basis of electronic properties. This usually highlights important electronic interactions as well as hydrogen bond donors and acceptors. It is assumed that the receptor surface is, for the most part, complementary to the ligand surface. Electrostatic potential can be calculated using all ligands, or using one or a few of the most active ligands. Using all the ligands may show you the absolutely essential features, whereas using the most active ligands may show you secondary interaction sites that can increase affinity.

4.1.1. Example

Several years ago, we constructed a model of a receptor site for high-potency sweeteners ***(7)***. Sweet taste is apparently mediated by G protein-coupled receptors ***(8)***, although the receptors have not yet been identified. We chose a series of five ligands (shown in **Fig. 1**) on which to base our model. This set had considerable structural diversity: two aspartic acid derivatives, one arylurea–aspartyl compound, and two arylguanidine–acetic acid derivatives. Sweetness potencies covered a range of three orders of magnitude. We were willing to assume that all of these compounds act at a common receptor because all five compounds (and other active analogs in each series) have several features in common:

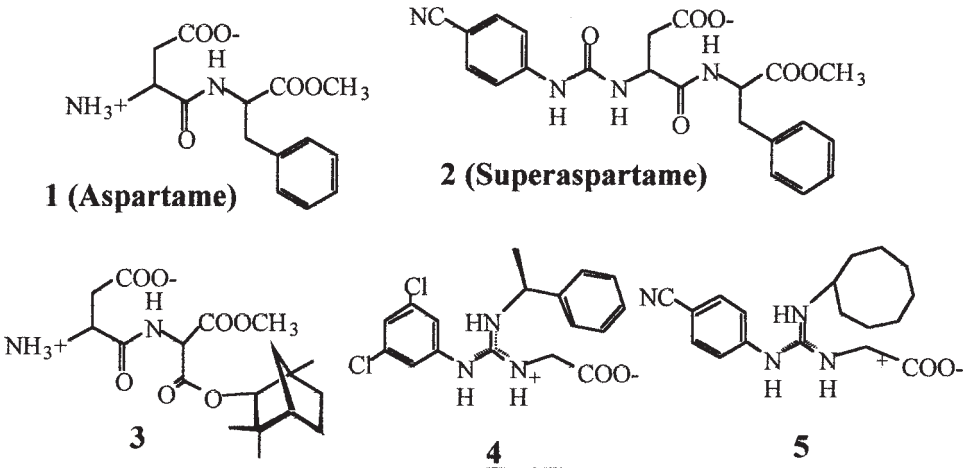


Fig. 1. Five potency sweet-tasting compounds used in constructing a surface model for a sweet-taste receptor.

1. An ionizable carboxylate group is required for activity.
2. Each compound has two or more polar N–H groups (amine, urea, amide, guanidinium).
3. Each compound has a large hydrophobic substituent, and in each structural class, potency depends on the size and shape of the hydrophobic group.
4. The most potent analogs have an aryl ring with strongly electronegative substitution.

Partial atomic charges were calculated for all five compounds using a semiempirical method intermediate neglect of differential overlap (INDO/S). Structures were calculated with carboxylate, amino, and guanidinium groups in their ionized states.

We first carried out conformational analysis on the dipeptide aspartame, and found about 150 low-energy conformers. This did not help us much in finding a starting point. However, the arylguanidines have only five or six low-energy conformers due to partial conjugation between the aryl ring and the guanidinium group. We were able to identify a single conformer of the arylguanidines that could match well with low-energy conformers of the more flexible dipeptides. In each case it was straightforward to superimpose carboxylate groups, two different N–H hydrogens, and a hydrophobic substituent. For the most potent compounds, we could also superimpose the aryl rings.

We chose to manually superimpose the five structures in a standard molecular-modeling software package (Quanta, **ref. *9***). Structures were modeled with electrostatic potential surfaces displayed. We initially superimposed regions

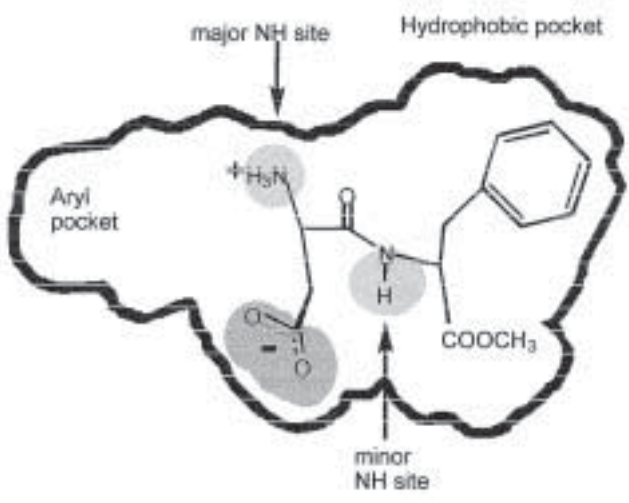


Fig. 2. Schematic representation of the surface model derived for the sweet-taste receptor active site.

with large electrostatic potential (carboxylate and N–H regions), then adjusted to maximize steric overlap for all five compounds.

Once the five structures were superimposed, a surface was calculated over the whole set. The surface was then colored according to the calculated electrostatic potential, based on the partial atomic charges of the five compounds. **Figure 2** shows a schematic representation of the surface.

4.1.2. Testing the Model

The model was tested in several ways. First, it was found that the model is useful in evaluating possible analogs. In general, compounds that have carboxylate and N–H groups in the right locations with something hydrophobic in the hydrophobic region have a high likelihood of having a sweet taste. The potency increases as more of the hydrophobic pocket is filled. Furthermore, structures that extend beyond the boundary of the hydrophobic pocket lose potency rapidly, indicating that we have done a reasonable job of mapping out this space. Second, we found that compounds that have negative potential rather than positive potential around the main N–H site (occupied by the amino group in **Fig. 2**) often had a strong bitter taste. Third, we used the model to correctly predict which stereoisomer in a racemic mixture was responsible for the sweet taste ***(10)***. Finally, we were able to identify compounds lacking one or more important binding features for receptor model binding, which acted as competitive antagonists for the sweet-taste receptor ***(11)***

4.2. Method 2: Atom-Based Models

The idea of constructing receptor-site models from atoms (or better yet, amino acid fragments) is an appealing one. After all, this is what real receptor sites are made of. The problem lies in the incredible number of degrees of freedom involved when we do not know anything about the three-dimensional structure. Which atoms or amino acids should we use? Where should we put them? How should we orient them?

Nevertheless, there have been several attempts to construct such models on the basis of one or more active ligands. One such approach is Vedani's `Yak` program ***(12)***, in which pseudoreceptors are constructed by the placement of amino acid side chains. Here we describe a method (Genetically Evolved Receptor Models, GERM) which we have devised for making atom-based, receptor-site models ***(13)***.

The GERM method uses atoms rather than amino acid fragments for two reasons. First, we have no basis for deciding what specific amino acid to use in any part of the receptor model. How are we to choose between aspartate and glutamate? Thus, we use atoms rather than molecules as our building blocks; these atoms are selected from the types of atoms that are typically found in proteins. Second, if we use atoms instead of fragments, we have only three degrees of freedom for each building block (*location* in *xyz* space); with fragments, we would need three more degrees of freedom to describe the *orientation* of the fragment.

Still, construction of a receptor model is a high-level combinatorial problem. Suppose we construct a shell of 60 atoms around our set of aligned ligands. These atoms represent the layer of receptor atoms that contact the ligands. If each of these atoms is chosen from a set of 15 different atom types, there are about 4×10^{70} different models. The GERM method uses a genetic algorithm ***(14)*** to find good models in a reasonable amount of time. The genetic algorithm is a method for rapidly searching through highly multidimensional space; it does not guarantee that the user will ever find the absolute best solution, but it very efficiently locates many very good solutions by mimicking the biological evolutionary processes of recombination, mutation, and natural selection.

Once again, the starting point is a series of compounds for which biological activity has been measured. We used a set of 22 high-potency sweeteners (compounds 1, 2, 4, and 5 of **Fig. 1**, and compounds 6–23 of **Fig. 3**). As described in Method 1, all compounds were modeled in low energy conformations and partial atomic charges were calculated using semiempirical methods. Compounds were superimposed manually using the Quanta program. Eleven of the compounds (2, 5, 6, 8, 9, 11, 13, 16, 19, 20, 23), encompassing a broad range of structural types as well as a broad range of biological activities (sweetness

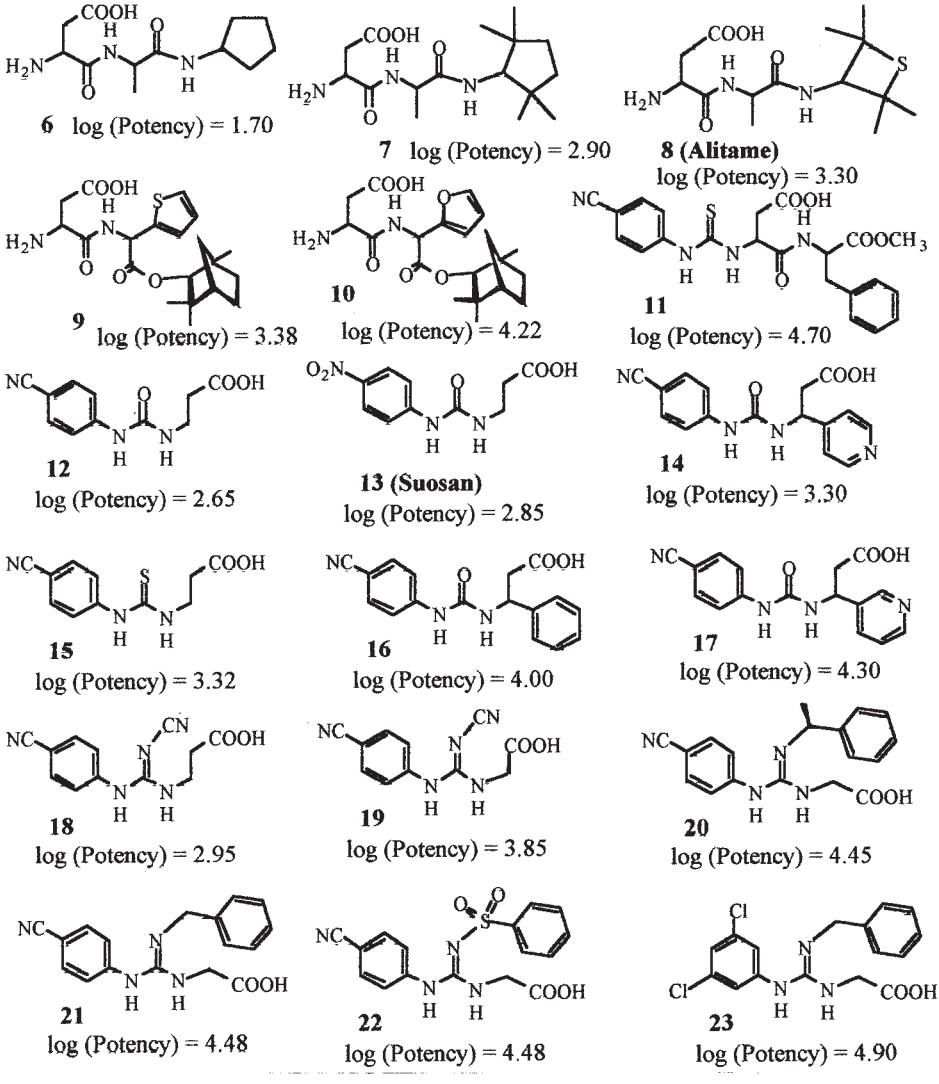


Fig. 3. Additional high potency sweet-tasting compounds used in constructing atom-based models for the sweet-taste receptor.

potencies) were selected as the training set. These compounds were used by the GERM program as the template around which a shell of 60 atoms was constructed. The program was run using the 11 structures as input. The final result is a set of models for which there is a high statistical correlation ($r = 0.94$)

between *calculated* ligand–receptor model binding and *experimentally measured* bioactivity. Each model is an array of atoms forming a shell around the ligand set, with a specific atom type at each of the positions.

4.2.1. Testing the Model

The first test of the model was to calculate ligand–receptor model-binding energies for the compounds omitted from the training set, and to extrapolate predicted potencies from these energies. Average error for these compounds was 0.44 log unit. This is considered an excellent result, considering the magnitude of error commonly encountered in measuring sweetness potencies with a human taste panel ***(15)***. This level of accuracy is considered useful in the context of a ligand design application, as well. The model can be used to screen large numbers of potential synthetic target molecules, allowing the selection of those that have the highest probability of success.

The result of the GERM calculation is a population of hundreds or thousands of models, all of which have a high correlation between calculated binding and measured bioactivity. Comparison of the models across the population ***(16)*** shows that, in some parts of the receptor model, a single atom type is highly conserved, whereas there is high variability in other regions. The most conserved positions correspond to the most important features for receptor recognition. In this series, e.g., in multiple runs of the GERM program, there were the following consistently observed features:

1. One or two positively charged atoms adjacent to the carboxylate groups of the high-potency sweeteners.
2. Negatively charged atoms adjacent to the major N–H group site.
3. A series of hydrophobic atom types around the hydrophobic pocket.

In a related study ***(17)***, we used a series of HIV protease inhibitors as the training set, and compared the calculated receptor models with the actual active site of the protease. The calculated models incorporated most (but not all) of the important features of the active site.

5. Notes

1. There are automated methods for structure alignment, such as Kearsley and Smith's Steric and Electrostatic ALignment (SEAL) program ***(18)***. We prefer manual alignment, especially for fine adjustment of the alignment, because the human brain can visually absorb and process pattern information in ways that are difficult to incorporate into computer programs. Depending on your problem, you may wish to try automated methods, especially as a source of alternative alignment ideas.
2. The statistician George E. P. Box is credited with the statement "All models are wrong, some are useful." Certainly all models created using these methods will

have errors. It is important to test such models, e.g., by docking other analogs into them and seeing whether the model is consistent with all known structure–activity results. Often, when a new analog is made, it will indicate parts of the model that are wrong. The model has to be modified, refined, and retested. In our experience, after just a few rounds of model refinement, the models are often quite reliable in predicting whether or not new analogs will be active.

3. Both methods implicitly build a completely closed-surface model, although we know from X-ray crystallographic studies that many binding sites leave some portion of the ligand exposed to solvent. For computed-surface models, you may discover some regions of your surface are sensitive to steric violation (compounds extending beyond the surface have substantially diminished activity). Other regions may be insensitive to substitution, and these may represent solvent-exposed parts of the ligand. The atom-based GERM method allows for solvent-exposed regions by allowing a "null" atom type. Parts of the receptor model can have no atom at all if this gives a better model. Again, this may point to solvent-exposed areas.
4. In the case of the atom-based GERM models, we have built models around a series of ligands for which the actual protein structure was known. Twelve HIV protease inhibitors were superimposed in their protease-bound conformations and model sites were generated. These were then compared with the protease. Most of the active site functional groups were present in the final models. In particular, the hydrophobic side chains, active site aspartates, bound water molecule, and hydrogen bond donors were reliably reproduced; hydrogen bond acceptors were often missed.

References

1. Dodziuk, H. (1995) *Modern Conformational Analysis.* VCH Publishers, New York.
2. Leach, A. R. (1996) *Molecular Modelling Principles and Applications*. Longman, Harlow, England.
3. Gasteiger, J. and Marsili, M. (1980) Iterative partial equalization of orbital electronegativity — a rapid access to atomic charges. *Tetrahedron* **36,** 3219–3228.
4. Nicholls, A., Bharadwaj, R., and Honig, B. (1993) GRASP: graphical Representation and Analysis of Surface Properties. *Biophys. J.* **64,** A166.
5. Hahn, M. (1995) Receptor surface models. 1. Definition and construction. *J. Med. Chem.* **38,** 2080–2090.
6. Hahn, M., and Rogers, D. (1995) Receptor surface models. 2. Application to quantitative structure-activity relationships. *J. Med. Chem.* **38,** 2091–2102.
7. Culberson, J. C. and Walters, D. E. (1991) Development and utilization of a three-dimensional model for the sweet taste receptor, in *Sweeteners: Discovery, Molecular Design and Chemoreception* (Walters, D. E., Orthoefer, F. T., and DuBois, G. E., eds.), American Chemical Society, Washington, DC, pp. 214–223.
8. Walters, D. E., DuBois, G. E., and Kellogg, M. S. (1993) Design of sweet and bitter tastants, in *Mechanisms of Taste Transduction* (Simon, S. A. and Roper, S. D., eds.), CRC Press, Boca Raton, FL, pp. 463–478.
9. Quanta, Molecular Simulations, Inc., San Diego, CA 92121.

10. Muller, G. W., Madigan, D. L., Culberson, J. C., Walters, D. E., Carter, J. S., Klade, C. A., DuBois, G. E., and Kellogg, M. S. (1991) High potency sweeteners derived from β-amino acids, in *Sweeteners: Discovery, Molecular Design and Chemoreception* (Walters, D. E., Orthoefer, F. T., and DuBois, G. E., eds.), American Chemical Society, Washington, DC, pp. 113–125.
11. Muller, G. W., Culberson, J. C., Roy, G., Ziegler, J., Walters, D. E., Kellogg, M. S., Schiffman, S. S., and Warwick, Z. S. (1992) Carboxylic acid replacement structure–activity relationships in suosan type sweeteners. A sweet taste antagonist. *J. Med. Chem.* **35,** 1747–1751.
12. Snyder, J. P., Rao, S. N., Koehler, K. F., and Vedani, A. (1993) Minireceptors and pseudoreceptors, in *3D QSAR in Drug Design. Theory, Methods and Applications* (Kubinyi, H., ed.), Escom, Leiden, The Netherlands, pp. 336–354.
13. Walters, D. E. and Hinds, R. M. (1994) Genetically Evolved Receptor Models (GERM): a computational approach to construction of receptor models. *J. Med. Chem.* **37,** 2527–2536.
14. Holland, J. H. (1975) *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.
15. DuBois, G. E., Walters, D. E., Schiffman, S. S., Warwick, Z. S., Booth, B. J., Pecore, S. D., Gibes, K., Carr, B. T., and Brands, L. M. (1991) A systematic study of concentration-response relationships of sweeteners, in *Sweeteners: Discovery, Molecular Design and Chemoreception* (Walters, D. E., Orthoefer, F. T., and DuBois, G. E., eds.), American Chemical Society, Washington, DC, pp. 261–276.
16. Walters, D. E. and Muhammad, T. D. (1996) Genetically Evolved Receptor Models (GERM): a procedure for construction of atomic-level receptor site models in the absence of a receptor crystal structure, in *Genetic Algorithms in Drug Design* (Devillers, J., ed.), Academic Press, London, pp. 193–210.
17. Walters, D. E. and Muhammad, T. D. (1998) Genetically Evolved Receptor Models (GERM): a comparison of evolved models with crystallographically determined binding sites, in *Rational Molecular Design in Drug Research* (Liljefors, T., Jørgensen, F. S., and Krogsgaard-Larsen, P., eds.), Alfred Benzon Symposium No. 42, Munksgaard, Copenhagen, Denmark, pp. 101–108.
18. Kearsley, S. S. and Smith, G. (1992) SEAL, the Steric and Electrostatic ALignment method for molecular fitting. *Tetrahedron Comput. Methodol.* **3,** 615–633.

16

Flexible Docking of Peptide Ligands to Proteins

Johan Desmet, Marc De Maeyer, Jan Spriet, and Ignace Lasters

1. Introduction

Computer-simulated ligand binding or docking is a useful technique when studying intermolecular interactions or designing new pharmaceutical products. In general, the purpose of a docking experiment is twofold: (1) to find the most probable translational, rotational, and conformational juxtaposition of a given ligand–receptor pair, and (2) to evaluate the relative goodness-of-fit for different computed complexes. From a computational point of view, these are extremely difficult tasks and a satisfactory general solution to the docking problem has not yet been found. To explain this, let us consider the naïve approach in which a ligand is systematically moved relative to a given receptor. Here, the term "moved" must be understood as the combination of all possible translational, rotational, and conformational changes of the ligand. These operations define the so-called "docking box," i.e., the *a priori* accessible phase space of the ligand, having dimensions $3 + 3 + N$ (three translational, three rotational and N conformational degrees of freedom). While the six topological dimensions already seriously impede a docking simulation, the *a priori* conformational flexibility of the ligand *and* the receptor certainly poses the hardest (and least studied) problem. A large part of this chapter is devoted to a possible solution to this problem.

Another critical point is the evaluation of each considered ligand–receptor structure comprising hundreds or thousands of atoms. The evaluation process is particularly important since it is meant to serve as a discriminator between "correct" and "incorrect" binding modes and, for multiple ligand docking, between binding and nonbinding molecules. Because the ligand binding affinity is directly related to the binding free energy, it is clear that an evaluation function that approaches best the experimental free energy is likely to yield the best

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

results. On the other hand, the evaluation of a single complex using a detailed cost function easily consumes in the order of one CPU-s. Therefore, the evaluation routines often consume the largest share of the total computation time.

In this chapter, we describe a combinatorial method for peptide docking that treats both the receptor and the ligand in a flexible way. The method primarily emphasizes the conformational part of the docking problem, without ignoring the translational/rotational and energetical aspects, however. The search method is basically a combinatorial buildup procedure: in the presence of the receptor protein, the peptide gradually grows both in length and in its number of conformational appearances. The population of computed peptide fragments can be effectively kept within bounds by using their relative binding energy as a filter. These values can be rapidly obtained because of the efficiency of the Dead-End Elimination (DEE) method, which enables fast remodeling of the side chains for each generated peptide configuration ***(1–6)***. This way, peptides of up to 20 amino acid residues can be successfully docked.

We briefly recall the algorithm's working mechanism and we try to rationalize its performance in terms of the ligand- and binding-site constitution, the search strategy (which can be modulated), and the allowed configurational space. Special attention is paid to the computational limitations of the intrinsically combinatorial method. The influence of the aforementioned factors is exemplified by a worked-out experiment on the docking simulation of a viral peptide to the MHC Class I H-2K^b receptor. In general, an excellent agreement can be obtained between the theoretical and the experimental structures, although significant deviations may occur at the level of individual side chains. The method also provides relevant information about local flexibility of both the ligand and the receptor. Finally, we demonstrate and discuss the applicability of the method in cases where the "perfect" backbone of the receptor is unknown.

2. The Peptide-Docking Algorithm

2.1. General Concept

Our docking algorithm falls into the class of combinatorial buildup algorithms. A given peptide sequence is assembled residue by residue in the vicinity of a putative binding site on a receptor. In this way we avoid any conformational bias from a starting structure because the ligand is built from scratch. The central feature of the method is a dynamic repository of peptide fragments which are characterized by their length, rotational/translational offset, main-chain dihedral angles, and interaction energy. During a simulation it is not necessary to store the peptide and receptor side-chain conformations, as they can be rapidly remodeled when needed. The latter is achieved by the DEE method, which allows the accurate positioning of 30–40 side chains in less than one-tenth of a CPU-s. The docking process itself occurs by cyclically

performing a selection, branching, side-chain modeling, and evaluation/storage step (*see* **Subheadings 2.3.1.** and **2.3.2.**). The net result of this process is that the repository gradually grows in the number of fragments with different positions, conformations, and length. This approach allows exhaustive and efficient sampling of the phase space at the same time, as the ligand visits the entire translational, rotational, and conformational space, whereas only energetically favorable fragments are further extended. This means that large "dry" branches of the combinatorial tree can be truncated at an early stage of the docking process. Also, because the DEE-routines take care of the side chains, the sampling of the conformational space can be confined to the peptide main chain. So far, this is the only approach that allows substantial side-chain flexibility for the receptor molecule and full torsional flexibility for the ligand.

2.2. Input Routines

Our docking routines are linked to the core of the commercial modeling package Brugel *(7)*, which comprises the basic routines for energy calculation using the CHARMM molecular mechanics force field *(8)*, rotation around single bonds, dynamic memory allocation, and the input/output of data such as atomic coordinates and diagnostic messages.

Prior to starting a docking experiment, the process must be initialized by reading a configuration script. First, the starting structure of the receptor molecule and the peptide ligand are read. The initial conformation of the peptide ligand is of minor importance, but a starting structure is required to define the connectivities, the bond lengths, and the angles which do not vary during the docking operations. Also, the initial location of the peptide is used as a convenient way to define the position and the boundaries of the translational docking box. The latter is an ellipsoidal volume, the principal axes of which are, by default, aligned with those of the peptide in its initial conformation. The center of the ellipsoid can either be left at the peptide's center of gravity or translated to the C_α-position of the root-residue (the initial building block of the peptide, *see* **Subheading 2.3.1.**). The dimensions and point density of the translational box can be set in a number of ways. The user can opt for a cubic or a radial lattice. In the former case, the number of grid points and their (constant) spacing along each of the principal axes define the translational grid. In the latter case, the grid points are located at the intersection of 26 radial vectors and a user-defined number of spheres with radii 1Δ, 2Δ, 4Δ, 8Δ, …, where Δ is typically chosen at 0.5 Å or 1.0 Å. The cubic lattice has a constant spacing and typically serves to "explore" a complete binding site. In contrast, the radial lattice has the highest density near its center and is therefore the method of choice if the binding site is approximately known. At each translational grid point, the ligand undergoes full, but discrete, rotation. The latter is performed

by a user-defined number of "roll" rotations around the *x*-axis (typically 6, i.e., in steps of 60°), combined with either 14 or 26 joint "pitch" rotations around the *y*-axis and "yaw" rotations around the *z*-axis.

Another important initialization routine is the loading of the rotamer library. This library contains 14,891 entries describing the physically possible combinations of main-chain and side-chain conformations of the 20 natural amino acid types. In contrast to statistical libraries ***(6,9,10)***, the rotamer library in this chapter results from energy parsing of individually generated conformations. This backbone-dependent library, the construction of which has been described earlier ***(11)***, is available on request. In essence, for each residue type there are 47 low-energy main-chain rotamers, and for each main-chain rotamer there are a variable number of backbone-compatible side-chain rotamers. Glycine, proline, and N- or C-terminal residues form an exception and have 125, 35, and 12 main-chain rotamers, respectively. The rotamers are stored in a four-dimensional (4D) array structure, where the dimensions indicate the residue type (1–20), the main-chain conformational type (1–47), the side-chain conformational type (1-[*variable*]), and the side-chain dihedral angle number (1–4), respectively. The separate handling of the main-chain and side-chain conformations is essential within the context of our docking strategy: the DEE-routines require a number of possible side-chain conformations for a given (generated) peptide main-chain conformation (*see* **Subheading 2.3.1.**).

The configuration script must also contain a list of receptor residue numbers that are treated in a flexible way during the docking process. Because the number of flexible receptor residues drastically affects the performance of the algorithm, this selection must be carried out with great care. Typically side chains within a 4 Å thick layer at the surface of the presumed binding site are kept flexible, but a preliminary visual inspection of the receptor structure with considerations about local packing and side-chain orientation is advisable. The limit for convenient docking is about 40 flexible side chains.

Finally, the search path can be controlled in the sense that one can freely select the peptide residue from where the buildup process will start. In addition, the way of building up the peptide, i.e., the order in which N- or C-terminally directed residues are added, can be controlled. It has turned out that the initial building block (the "root" residue) is preferably selected at a residue that may form tight interactions with the receptor.

2.3. Docking Algorithm

2.3.1. Docking of the Root Residue

Because the method is essentially combinatorial, the main problem is to keep the number of peptide fragments within manageable proportions. In a

Table 1
Statistical Data of Peptide Fragments Generated During VSV-8 Docking

Length	Peptide	#Conf	#Accep	%Accep	E_best	ΔE_best	CPU/s	CPU/conf
1	----**Y**---	311,892	920	0.29	–24.4	–24.4	11,358	0.036
2	----Y**Q**--	43,240	2,074	4.80	–43.8	–19.4	4,592	0.106
3	----YQ**G**-	259,250	13,081	5.05	–51.2	–7.4	12,500	0.048
4	----YQG**L**	156,972	289	0.18	–73.9	–22.7	5,444	0.035
5	---**V**YQGL	13,583	1,064	7.83	–82.0	–8.1	679	0.050
6	--**Y**VYQGL	50,008	1,148	2.30	–109.5	–27.5	4,060	0.081
7	-**G**YVYQGL	143,500	11,626	8.10	–120.1	–10.6	9,918	0.069
8	**R**GYVYQGL	139,512	323	0.23	–147.1	–27.0	12,743	0.091
Sum or average:		1,117,957	30,525	2.73		–18.4	61,294	0.055

Column 1: fragment length in number of residues; column 2: fragment sequence in one-letter code (bold indicates added residue); column 3: total number of investigated configurations; column 4: number of accepted configurations on basis of eqn. 2, using *max_tension* = 10 kcal mol^{-1}; column 5: acceptance ratio in% (#accep/#conf × 100); column 6: binding energy of the lowest energy fragment in kcal mol^{-1}; column 7: incremental binding energy, i.e., *E_best* of the given length class minus *E_best* of the class of one residue shorter fragments; column 8: CPU-time (in seconds) required to process all configurations of the given length; column 9: CPU-time per configuration.

typical experiment (*see* **Subheading 3.1.**) there are 79 translations (using a radial grid with three spheres, each having 26 points on their surface, plus the origin), combined with 84 rotations (6 "roll" × 14 "pitch/yaw" rotations), again combined with 47 main-chain rotamers of the root residue, thus in total 311,892 configurations that are systematically processed. The average calculation time required per configuration is about 0.05 CPU-s (*see* **Subheading 3.2.** and **Table 1**). If all these root configurations were to be combined with another 47 rotamers of the next residue, the computational requirements would become unacceptable. Therefore, we have looked for a discriminating criterion that allows a substantial reduction of the number of configurations while not losing "correct" structures. We have found that an absolute energy-based cutoff was not powerful enough in reducing the configurational space. The reduction to a predefined number of energetically top-ranked configurations ***(12)*** was *a priori* discarded because of the different inherent flexibility of the residue types and the possible variation in interactions for a given residue in different topological situations. Our results *a posteriori* confirmed the inappropriateness of such criterion (*see* **Table 1** and **Subheading 3.2.**). The most useful elimination criterion was found to be based on the relative binding energy of peptide configurations of the same length. Concretely, the calculated binding energy of a peptide fragment c, $E_{bind}(c)$, is compared with the lowest binding energy found

so far for all fragments of the same length, $E_{\text{bind}}(lowest(length(c)))$; the difference between both is here defined as the "tension" of a given fragment, $T(c)$:

$$T(c) \equiv E_{\text{bind}}(c) - E_{\text{bind}}(lowest(length(c))) \quad (1)$$

Fragments in a given configuration are rejected if their tension exceeds a predefined maximum value, *max_tension* or, conversely, their binding energy must be lower than that of the best fragment found so far plus the interval *max_tension*:

$$E_{\text{bind}}(c) \leq E_{\text{bind}}(lowest(length(c))) + max_tension \quad (2)$$

Several test experiments have shown that the aimed acceptance ratio of maximally 1/47 for the root residue could always be attained using a *max_tension* value of 10 kcal mol^{-1} (data not shown).

The processing of each of the possible root configurations occurs as follows. First, the root residue is translated to the considered translational grid point (by its C_α-atom), rotated into the appropriate orientation and then the main-chain φ and ψ angles are generated. Next, a quick test on a possible close van der Waals contact between the root main-chain atoms (including C_β) and the fixed atoms of the receptor molecule is performed using a global cutoff energy of + 5 kcal mol^{-1}. If acceptable, the side-chain conformations (as known in the rotamer library) are generated and quickly tested for atomic overlap with the fixed part of the receptor using the same criterion as for the main-chain atoms. Similarly, the receptor side chains are tested for steric compatibility with the root main-chain atoms. The coordinates of the remaining side-chain rotamers of both the peptide root residue and the flexible residues of the receptor are then transferred to the DEE-routines (*see* Chapter 12).

The DEE-routines rapidly provide an answer to two important questions:

1. What is the best possible side-chain arrangement given the position and main-chain conformation of the considered peptide fragment?
2. What is the binding energy of that fragment?

The first problem is addressed by the original DEE-algorithm which has been speed- and memory-optimized for small sets of rotatable side chains and for repeated calls with small variations in main-chain coordinates. The net result is the energetically best possible side-chain conformation of the receptor and the considered root residue. These results allow then the calculation of the total energy, defined as the sum of the side-chain self energy, the side-chain–backbone and the side-chain–side-chain interaction energy. Next, we add to this value the self-energy of the peptide main chain and then we subtract the total energy of the receptor side chains that have been modeled once at the start of the algorithm in the absence of any peptide. This way, we obtain $E_{\text{bind}}(c)$,

which can be interpreted as the direct interaction energy between the root residue and the receptor, including internal strain in both molecules.

Once the entire set of possible root configurations has been processed, it is trivial to search the one with the lowest binding energy and to remove configurations with a too high energy, using **Eq. 2**. Thus, at the end of this routine we have a list of root configurations for which the binding energy falls within a given energy interval above the best one, and for which all data necessary to reconstruct the coordinates is known.

2.3.2. Fragment Extension

In this stage, partial peptide configurations are selected from the fragment repository and are stepwise elongated one residue at a time. The basic handling of these fragments is identical to that of the single-residue root configurations as explained earlier. Still, a number of modifications have been necessary. We recall that this routine essentially comprises three execution stages: (1) a selection–combination stage, (2) a DEE side-chain positioning stage, and (3) an evaluation–storage stage. These stages are executed in a cyclic way until exhaustion of the fragment repository (*see* **Fig. 1**).

In contrast to the systematic exploration of the configurational space for the root residue, this routine each time screens the fragment repository and selects a previously accepted configuration using the criterion lowest tension first. This way, the energetically best or "most probable" fragments are extended first. Extension occurs by generating the coordinates of all possible main-chain rotamers of the next N- or C-end directed residue-to-be-added, as defined in the initialization script. These combinations are then processed individually in the same way as the root configurations: first a rapid precheck occurs on the steric compatibility of the added main-chain atoms and, if this is all right, also on the peptide and receptor side-chain rotamers. From the reduced set of side-chain rotamers the DEE-routines then calculate the energetically most favorable global side-chain arrangement as well as $E_{\text{bind}}(c)$, the binding energy of each new, extended fragment. Finally, the algorithm decides about the acceptability of the elongated fragments by comparing their $E_{\text{bind}}(c)$ with the best binding energy found so far for fragments of the same length, $E_{\text{bind}}(lowest(length(c)))$ (**Eqs. 1** and **2**). Accepted configurations are added to the fragment repository by storing their characterizing properties, i.e., their length, rotational–translational offset, main-chain conformation, and interaction energy.

After each cycle, it is checked whether one of the accepted combinations has yielded a new, lower value for $E_{\text{bind}}(lowest(length(c)))$ and, if so, the previous value is replaced by the newer. As a consequence, such operation may indirectly push some of the previously accepted fragment configurations

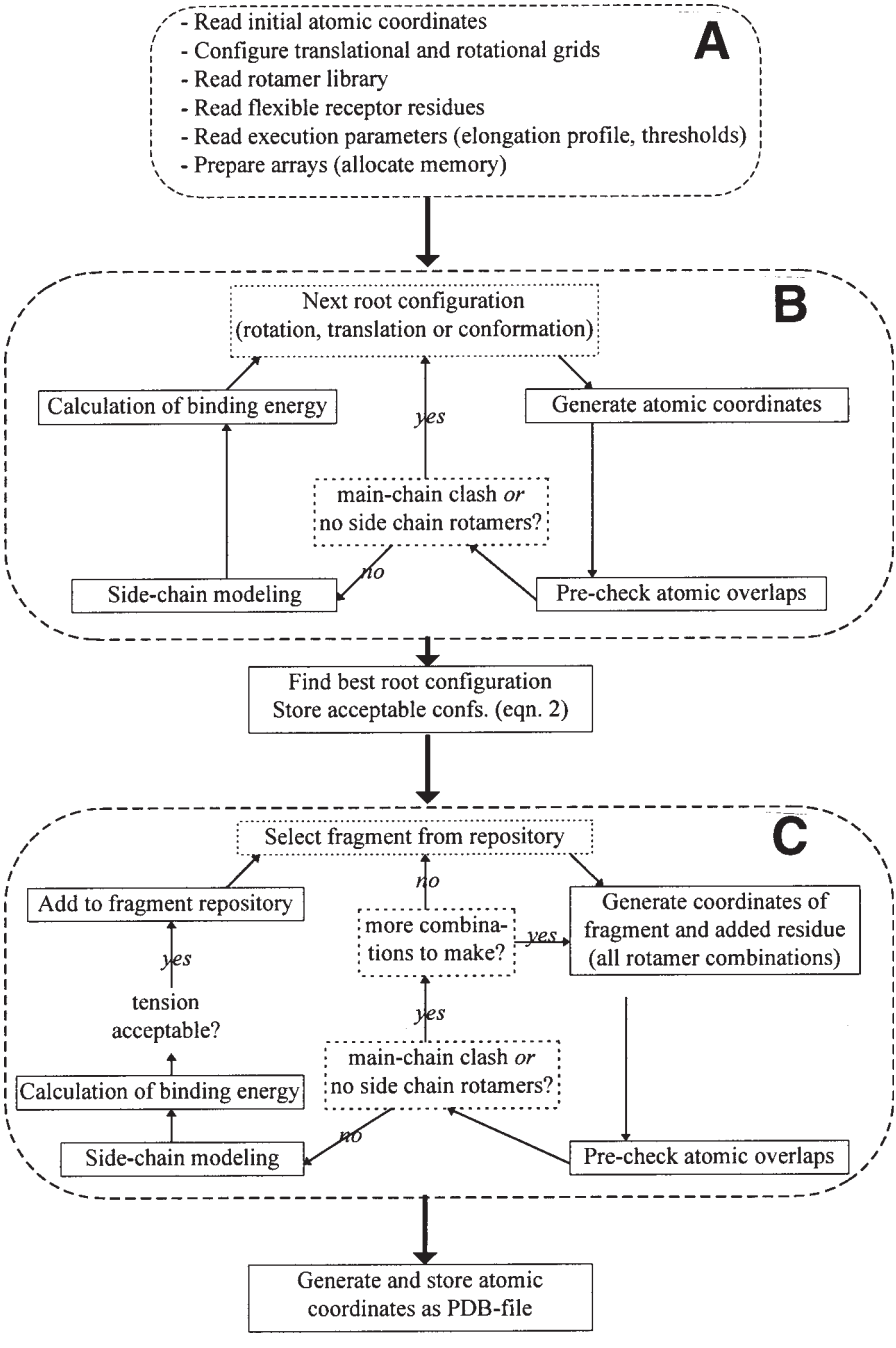


Fig. 1. Flowchart of the docking algorithm. The program consecutively executes the routines labeled (a), then cycles until exhaustion through the routines (b) and then, also until exhaustion through routines (c). All routines are described in **Subheading 2.3.** (a) Initialization routines. (b) Docking of the root residue. (c) Fragment extension.

beyond the acceptance limit (**Eq. 1**). However, rather than definitely removing "knocked-out" configurations, the algorithm leaves them untouched because there is another mechanism by which inactive configurations may be "recovered." Indeed, it happens that the lowest energy configuration of a particular length class appears inextensible. In that case, the algorithm selects the next best configuration as the "lowest." Inextensibility of a lowest energy configuration can sometimes be ascertained immediately after its appearance (it will be directly reselected because it has zero tension), but it is also possible that its "children" (and thus the "parent" as well) are knocked out later due to either the late appearance of a new configuration with a much lower energy, and/or to the fact that the children themselves appear inextensible. In such a case, $E_{\text{bind}}(lowest(length(\text{parent})))$ will *in*crease and it is therefore important that as many as possible "lost" configurations of *length*(parent) are recovered — which would not be possible if they had been removed from the repository. To find out quickly whether the children of a lowest energy configuration are extensible, their tension is artificially set to zero so that they are reselected at once. Such (and other) manipulations of the search path are known as "directed searching" ***(13,14)***. Especially in the early stages of the elongation process, when the sampling of the phase space is still very sparse, the lower bounds of the acceptance intervals can be quite dynamical.

Besides the special treatment of lowest energy configurations, we also apply another element of directed searching. We have found that the efficiency of the search could be drastically improved by retarding a little the selection of longer fragments. Needless in-depth branching of energetically constrained peptide fragments can be prevented by forcing the algorithm to select fragments of length $l - 1$ until at least a good approximation of $E_{\text{bind}}(lowest(l))$ is obtained. In practice, when the fragment repository is screened for the next best configuration, a "penalty" of $(length(c) - 1)extra_tension$ is added to $E_{\text{bind}}(c)$. Here, *extra_tension* is a user-defined parameter that we usually set at $max_tension/2 \leq extra_tension \leq max_tension$. This drives the elongation process toward a search in width (i.e., shortest fragments first) rather than in depth.

2.4. End Stage Routines

The algorithm spontaneously comes to an end when all partial peptide configurations have been processed and only full-length peptides are left. Because these configurations are stored merely by their rotational/translational offset and main-chain dihedral angle values, the algorithm has to retrieve the atomic coordinates by a final application of the DEE-routines to the peptide and receptor rotatable side chains. These structures are then output to disk in order of increasing energy. Also, the history of the docked peptide solutions is given in the form of the tension of the fragments from which they have originated. This allows the user to verify that the combinato-

rial tree was not truncated too drastically. Conversely, when too many parent fragments have a tension close to *max_tension* it means that probably a number of viable fragments have been improperly rejected and a new experiment with a higher cutoff value is advisable.

3. Worked Example

3.1. Experimental Setup

We have tested the docking method on several ligand–receptor systems ***(11)***, one of which is here described in detail with special attention to the practical performance and potential problems. Concretely, we discuss the docking of the octapeptide VSV-8 (RGYVYQGL) to murine MHC class I H-2K^b ***(15–17)***. The following experimental conditions were used.

1. Peptide buildup: Tyr-5 was chosen as the root residue because of its potential to form multiple contacts with the ligand-binding site (Starting from Arg-1 or Leu-8 does not significantly alter the results ***[11]***.) Elongation proceeded first toward the C- and then toward the N-terminal end: ----y--- > ----yq-- > ----yqg- > ----yqgl > ---vyqgl > ---yvyqgl > -gyvyqgl > rgyvyqgl.
2. Translations: The C_α-atom of the root residue (and by this the whole peptide) was systematically translated over 79 grid points that were homogeneously distributed over three spherical shells at distances of 1, 2, and 4 Å from the initial position that was taken from the X-ray structure of the complex. The translational volume (268 Å^3) is approximately one-third of the peptide's molecular volume (953 Å^3).
3. Rotations: At each translational grid point, full rotation was allowed by 6 "rolls" combined with 14 "pitch/yaw" operations.
4. Conformations: For the peptide residues Tyr-3, Val-4, Tyr-5, and Gln-6, the rotamer library provides 47 main-chain conformations; for Gly-2 and Gly-7 there are 125 rotamers and for the N- and C-terminal residues Arg-1 and Leu-8 there are 12.
5. Peptide and receptor side-chain conformations: As explained, the side-chain conformations are rebuilt for each peptide main-chain configuration by the DEE-routines and therefore do not constitute the combinatorial tree. On average, there are 16 side-chain rotamers available for each residue main-chain rotamer. Besides the 8 peptide residues, 28 receptor residues having at least one atom within 4 Å from the peptide in the complex were assigned to be flexible during the docking.
6. Water molecules: This experiment was performed in the presence of nine crystallographically determined buried water molecules that were considered as part of the protein. Experiments in absence of structured water molecules have also been performed but are discussed elsewhere ***(11)***.
7. Other parameters: *max_tension* = 10 kcal mol^{-1}, *extra_tension* = 10 kcal mol^{-1}. This means that fragment configurations were accepted within an interval of 10

kcal mol^{-1} and that the combinatorial search occurred in width, i.e., peptide length class by length class.

8. Other conditions: The experiments were performed on a Silicon Graphics Indigo2 workstation (SGI, Mountain View, CA) equipped with a single R10000-175MHz processor.

3.2. Program Execution

The experiment finally yielded 323 full-peptide configurations within an energy interval of 10 kcal mol^{-1} (*see* **Table 1**). The total time required was 61,294 CPU-s or about 17 CPU-h. The initializations took only 61 s and are ignored in **Table 1**. In total, 1,117,957 peptide fragment configurations have been processed as described in **Subheading 2.3.** and **Fig. 1**. This means that, on average, the algorithm investigated about 20 configurations per second (0.055 s/configuration).

Performing a search in width (length class by length class) also has the elegant side effect of generating a neat output that can easily be analyzed afterward. The statistical results for each class of partial peptides is summarized in **Table 1**. The initial docking of the root residue Tyr-5 required the processing of 311,892 individual configurations. Interestingly, only 920 different configurations had a binding energy within 10 kcal mol^{-1} above the lowest value (–24.4 kcal mol^{-1}). This means that 99.71% of the explored space could be ignored in further extensions. This also shows the importance of selecting a root residue that forms many potential contacts with the receptor and that therefore exhibits a great discriminative power. The side chain of Tyr-5 indeed occupies the deep pocket C in the MHC-receptor-binding site ***(16)***. The straightforward exploration of the phase space for the root residue also consumes a large, but not excessive, share of the total calculation time (11,358 CPU-s or 19%) which indicates that there is still some room for a broader translational sampling.

The combination of the 920 root configurations with the half-exposed Gln-6 led to 2074 low-energy dipeptide configurations. Although each of the root configurations have been combined with 47 main-chain rotamers, after DEE-modeling of the side chains, on average only 2 of them were maintained (4.80%). The addition of the next residue, Gly-7, caused more difficulties: this residue type has no side-chain and 125 main-chain rotamers. This required the investigation of $2074 \times 125 = 259{,}250$ individual configurations. Besides the long calculation time (12,500 s) the acceptance ratio was also quite high (5.05%), leading to 13,081 different tripeptide configurations. However, the next residue Leu-8, which is completely buried into pocket F, yielded acceptable tetrapeptides in only 0.18% of the possible combinations. The extension then proceeded toward the N-terminus by the addition of Val-8. In the crystal

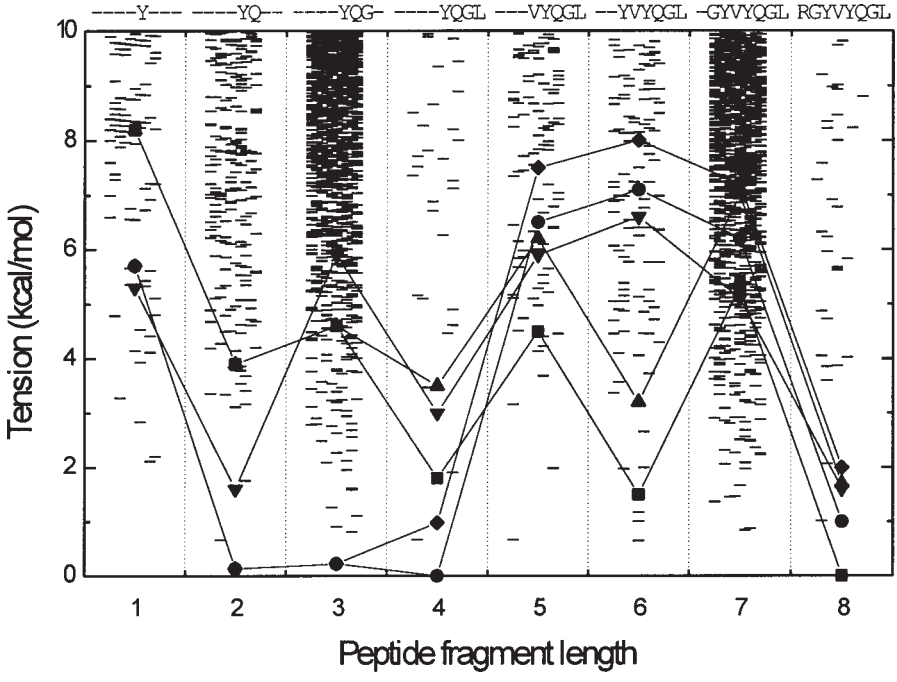


Fig. 2. Energy distribution of docked peptide fragments. The partial peptide configurations ("fragments") that have been accepted during the VSV-8 docking experiment (*see* **Subheading 3**) are shown by plotting their tension (defined by **Eq. 1**) as a function of their length (the number of residues). At the top of the diagram the fragment constitution is indicated. In order not to overload the picture, only 10% of the accepted fragments (selected on a random basis) are shown. For the five best full-length peptides the tensions of the predecessor fragments are indicated by symbols and connected by straight lines. When read from the left to the right, the curves show the "history" of the fragments that eventually lead to one of the five best full-length peptides.

structure, this residue is completely directed toward the solvent and this is reflected in the very high acceptance ratio (7.83%) and the low incremental binding energy (–8.1 kcal mol^{-1}). In contrast, the next residue Tyr-3 shows a low acceptance ratio (2.30%) and a quite specific binding (ΔE_best = –27.5 kcal mol^{-1}); in the crystal structure this residue occupies the shallow pocket D ***(16)***. The addition of Gly-2 shows about the same features as the other Gly at position 7: high acceptance ratio and low incremental binding energy. The final combination with the N-terminal Arg-1 again caused a steep decrease in the number of (full-length) configurations to only 323. This residue also appeared to interact strongly with the binding site, which is in agreement with the experimental observations ***(15,16)***.

Figure 2 shows the energy distribution of the different accepted peptide fragments. Interestingly, each length class shows a comparable profile, i.e., a low density at low tensions and a gradually increasing density at higher tensions. This means that the last-added residue interacts "optimally" for only few configurations, whereas in the majority of the cases the interaction is "suboptimal." This phenomenon has important consequences with respect to fragment extension. In cases where low-energy fragments (from the low-density region) lead to successful extension, the parameter *max_tension* could in principle be kept low which would drastically enhance the performance of the program. On the other hand, it is certainly not guaranteed that the lowest-energy configuration(s) are always extensible up to the full-length peptide. Therefore, a considerable safety margin must be included in *max_tension* to account for "false positives." In this experiment we found two such cases, i.e., Tyr-5 and its neighbor Val-4. Because Tyr-5 (the root residue) can freely translate and rotate and because of its bulky side chain, it may assume several false positive configurations that have a lower binding energy than in the crystal structure. For Val-4, the situation is different. In the crystal structure of the complex this residue is completely solvent oriented, whereas in some of the configurations generated during the experiment it buries its side chain, thereby leading to energetically favorable but inextensible configurations. In **Fig. 2** we also show the "history" of the five best full-length peptide configurations, i.e., the tension of the fragments from which they have originated. It is seen that the best final results do not always descend from the best intermediates; instead, viable intermediates lie in a range of about 4–8 kcal mol^{-1}. Importantly, this band coincides with the low-density region plus the lower part of the high-density region. This explains the success of the method for the current experiment, but at the same time it pinpoints a potential problem when docking peptides that bulge out from the binding site. Then, correct but weakly interacting, intermediate fragments might not stand the competition with false positives, thereby leading to wrong results. We have tested this possibility by the docking of SEV-9 (FAPGNYPAL) to the same H-$2K^b$ receptor ***(11)***. In this complex, the residues Gly-4 and Asn-5 form a β-bulge at about the same position as Val-4 in VSV-8 ***(15)***. Neither of both residues form van der Waals contacts with the receptor. In addition, this peptide lacks Tyr-3, which is replaced by Pro. Also, Tyr-6, which is topologically equivalent to Tyr-5 in VSV-8, forms only half the number of van der Waals contacts with the receptor (8 instead of 16 ***[15]***). Still, the obtained structures were in good agreement with the crystal structure (main-chain root-mean-square deviation [RMSD] = 1.33 ± 0.02 Å), but an explosion of intermediate fragments was observed when making combinations with the least-constrained residues Gly-4 and Asn-5 ***(11)***. This indicates that the main difficulty of this approach is not so much the existence of false positive inter-

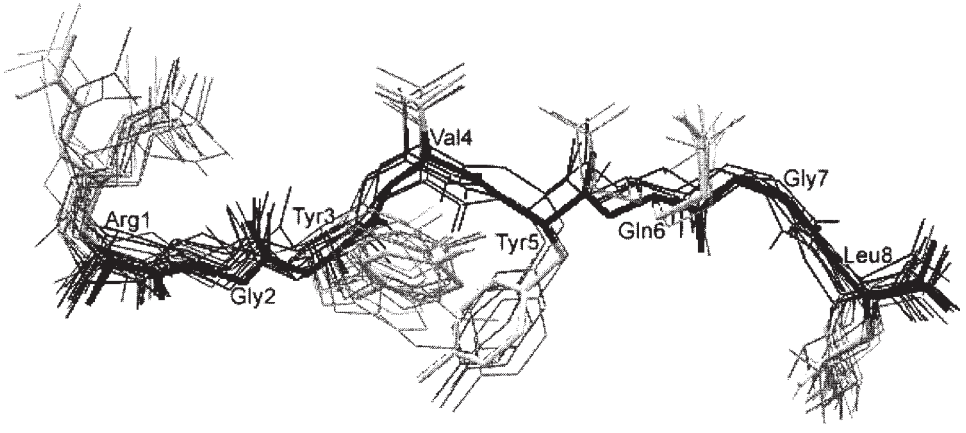


Fig. 3. Drawing of the 43 lowest energy peptides resulting from the VSV-8 experiment. The crystallographically determined structure is presented by the sticks model. Black is used for the main-chain atoms and gray for the side-chain atoms. Only "heavy" (non-H) atoms are shown. The viewpoint is from the "side" of the peptide with the N-terminus on the left. In the complex, the peptide is encompassed by the $\alpha_1\alpha_2$ domain (not shown) with the α_2-helix in front, the α_1-helix at the back, and the β-sheet at the bottom; the upper part of the peptide is solvent accessible.

mediates (the value of *max_tension* = 10 kcal mol^{-1} is sufficiently high to account for this), but rather the computational time and memory requirements when a burst of intermediate structures emerge.

3.3. Resulting Structures

Docking of the VSV-8 peptide to the MHC H-2K^b receptor finally yielded 323 structures within an energy interval of 10 kcal mol^{-1}. Of these, 43 had a binding energy within 5 kcal mol^{-1} above the lowest (–147.1 kcal mol^{-1}) (displayed in **Fig. 3**). Compared with the crystal structure, the lowest energy peptide had a main-chain RMSD of only 0.56 Å. For the 43 best structures, the average RMSD was 0.89 ± 0.27 Å, and for all 323 results it was 1.01 ± 0.39 Å. Although all modeled peptide structures had a sufficiently low RMSD to consider them as correct, a slight degradation in quality was observed for higher-energy structures. The differences were due to subtle translational and/or conformational changes, dispersed over the entire peptide (*see* **Fig. 3**). The anchor residues Tyr-3, Tyr-5, and Leu-8 were correctly packed into their complementary pockets ***(16,17)***, although the side chain of Tyr-3 in general displayed a small coplanar upward shift and the side chain of Leu-8 adopted two different conformational states. Other apparently bistable conformations were observed for Gln-6 and Arg-1 (*see* **Fig. 3**). The side-chain conformation

of the former residue depended on the translational position of the peptide main chain and covaried with a conformational change of the receptor residues Glu-152 and Arg-155. The side chain of the peptide N-terminal residue Arg-1 was the least focused and tended to fold back on the main chain.

At the level of the receptor molecule, 24 of the 28 residue side chains that were kept flexible during the experiment were correctly predicted in the lowest energy structure (data not shown). The four that were deviating were not directly involved in ligand binding and their conformations can be considered as energetically favorable alternative states. For the 43 lowest-energy structures, we occasionally observed transitions for the side chains that were in contact with the peptide. Most of these conformational changes are small (for Thr–143,163 and Tyr-84,59,171). Only three side chains underwent significant transitions, i.e., Lys–66 and, as indicated earlier, Glu-152 and Arg-155. Interestingly, the alternative conformation for the latter two residues has also been crystallographically observed, i.e., in the structure of the same H-2K^b receptor complexed with the nonapeptide SEV-9 ***(15)***.

4. Conclusions

By this method we have opted for a combinatorial buildup procedure that constructs the peptide from scratch in the vicinity of the binding site. In doing so, we basically follow the approach of Moon and Howe ***(12)*** who "grow" a peptide by combining full-residue rotameric templates. However, the usage of the DEE-method has proven to be an elegant way to disconnect the main-chain from the side-chain problem, as first recognized by Leach ***(18)***. By this method one can rapidly obtain the optimal side-chain conformation for each generated peptide main-chain structure. Importantly, the set of side chains is not limited to those of the peptide itself, but can include up to about 40 side chains from the receptor. This allows a considerable degree of flexibility at the level of the receptor molecule. In earlier experiments on SEV-9 docking to an "imperfect" H-2K^b receptor structure, we have observed that minor errors in the receptor main chain (approx 0.7 Å) can be "absorbed" by the flexible treatment of the side chains ***(11)***.

Most other peptide docking methods avoid the combinatorial problem in different ways, either by a combination of energy minimization and Monte Carlo simulation ***(19)*** or by single-residue docking in combination with loop closing ***(20–22)***. In view of the complexity of the energetical landscape defined over the numerous translational, rotational, and conformational degrees of freedom, it is highly questionable whether these approaches are in general capable to find the optimal solution(s). From our experience we feel that a combinatorial approach is the method of choice when dealing with the rugged landscapes encountered with the docking of flexible ligands. On the other hand, each com-

binatorial procedure invariable faces severe computational limitations that can be relieved only in particular conditions and by applying clever search techniques and pruning methods. With respect to peptide docking, we have demonstrated that such techniques can be successfully applied. Concretely, the pruning of branches (partial peptide configurations) by setting energetical boundaries (the parameter *max_tension*) can be accomplished in such a way that (1) the final optimal solutions do not get lost and (2) the computational task remains feasible. Within this respect, three very important conclusions can be drawn from the experimental data presented in **Fig. 2**.

1. Each fragment length class has a comparable energy distribution profile (i.e., a low-density at low tensions and a higher density at high tensions) regardless of its degree of burial. This is a favorable situation, as it means that, if low-tension inextensible fragments occur, they are not very populated. As a consequence, it should be possible to readily "unmask" them as false positives by applying more sophisticated search techniques (research in progress). On the other hand, if the low-density region comprises good, extensible fragments they will rapidly lead to low-energy extended fragments, thereby enabling efficient pruning of this class.
2. Low-energy fragments that lead to full-length peptides appear within a relatively narrow band of 4–8 kcal mol^{-1}. Together with the previous observation, this a posteriori justifies the usage of a limited cutoff range (*max_tension* = 10 kcal mol^{-1}) to keep the number of intermediate fragments within bounds. It also means, at least in the studied example, that the bound peptide does not incorporate considerable local strain.
3. The total number of intermediate fragments remains fairly constant over the different length classes. Perhaps the most important and surprising observation is the fact that the combinatorial buildup does not lead to an explosion of fragments. On the other hand, the combination with solvent oriented residues like Gly7 (class 3), Val4 (class 5), and Gly2 (class 7) does significantly increase the number of fragments. Potential problems related to computational feasibility may therefore occur when consecutive residues are lacking specific interactions with the receptor. Still, if the algorithm succeeds to surmount these "difficult residues" one may expect, in general, a near-linear increase of the number of fragments as a function of their length.

Considering the aforementioned points, we conclude that our peptide docking algorithm is able to produce structures of the complex that are closely matching the crystal structure in a reasonable amount of computing time. The success of the method is most likely due to a combination of (1) the choice of a combinatorial approach, (2) the usage of a very detailed rotamer library, (3) the efficient and reliable pruning of the combinatorial tree on basis of the relative binding energy of peptide fragments, (4) two directed searching techniques,

and (5) the application of the DEE-method to rapidly determine the optimal conformation of the side chains of both the peptide and the receptor molecule.

References

1. Desmet, J., De Maeyer, M., Hazes, B., and Lasters, I. (1992) The dead-end elimination theorem and its use in protein side-chain positioning. *Nature* **356,** 539–542.
2. Lasters, I. and Desmet, J. (1993) The fuzzy-end elimination theorem: correctly implementing the side chain placement algorithm based on the dead-end elimination theorem. *Protein Eng.* **6,** 717–722.
3. Desmet, J., De Maeyer, M., and Lasters, I. (1994) The "dead-end elimination" theorem: a new approach to the side-chain packing problem, in *The Protein Folding Problem and Tertiary Structure Prediction* (Merz, K. Jr. and LeGrand, S., eds), Birkhäuser, Boston, pp. 307–337.
4. Goldstein, R. F. (1994) Efficient rotamer elimination applied to protein side-chains and related spin glasses. *Biophys. J.* **66,** 1335–1340.
5. Lasters, I., De Maeyer, M., and Desmet, J. (1995) Enhanced dead-end elimination in the search for the global minimum energy conformation of a collection of protein side chains. *Protein Eng.* **8,** 815–822.
6. De Maeyer, M., Desmet, J., and Lasters, I. (1997) All in one: a highly detailed rotamer library improves both accuracy and speed in the modelling of sidechains by dead-end elimination. *Fold. Des.* **2,** 53–66.
7. Delhaise, P., Bardiaux, M., and Wodak, S. (1984) Interactive computer animation of macromolecules. *J. Mol. Graph.* **2,** 103–106.
8. Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* **4,** 187–217.
9. Ponder, J. W. and Richards, F. M. (1987) Tertiary templates for proteins. Use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.* **193,** 775–791.
10. Dunbrack, R. L. Jr. and Karplus, M. (1994) Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nat. Struct. Biol.* **1,** 334–340.
11. Desmet, J., Wilson, I. A., Joniau, M., De Maeyer, M., and Lasters, I. (1997) Computation of the binding of fully flexible peptides to proteins with flexible side chains. *FASEB J.* **11,** 164–172.
12. Moon, J. B. and Howe, W. J. (1991) Computer design of bioactive molecules: a method for receptor-based de novo ligand design. *Proteins: Struct. Funct. Genet.* **11,** 314–328.
13. Pearl, J. and Korf, R. E. (1987) Search techniques. *Annu. Rev. Comput. Sci.* **2,** 451–467.
14. Bruccoleri, R. E. (1994) Conformational search and protein folding, in *The Protein Folding Problem and Tertiary Structure Prediction* (Merz, K. Jr. and LeGrand, S., eds.), Birkhäuser, Boston, pp. 125–163.

15. Fremont, D. H., Matsumura, M., Stura, E. A., Peterson, P. A., and Wilson, I. A. (1992) Crystal structures of two viral peptides in complex with murine MHC class I H-2K^b. *Science* **257,** 919–927.
16. Matsumura, M., Fremont, D. H., Peterson, P. A., and Wilson, I. A. (1992) Emerging principles for the recognition of peptide antigens by MHC class I molecules. *Science* **257,** 927–934.
17. Fremont, D. H., Stura, E. A., Matsumura, M., Peterson, P. A., and Wilson, I. A. (1995) Crystal structure of an H-2K^b-ovalbumin peptide complex reveals the interplay of primary and secondary anchor positions in the major histocompatibility complex binding groove. *Proc. Natl. Acad. Sci. USA* **92,** 2479–2483.
18. Leach, A. R. (1994) Ligand docking to proteins with discrete side-chain flexibility. *J. Mol. Biol.* **235,** 345–356.
19. Caflisch, A., Niederer, P., and Anliker, M. (1992) Monte Carlo docking of oligopeptides to proteins. *Proteins: Struct. Funct. Genet.* **13,** 223–230.
20. Rosenfeld, R., Zheng, Q., Vajda, S., and DeLisi, C. (1993) Computing the structure of bound peptides. Application to antigen recognition by class I major histocompatibility complex receptors. *J. Mol. Biol.* **234,** 515–521.
21. Rosenfeld, R., Zheng, Q., Vajda, S., and DeLisi, C. (1995) Flexible docking of peptides to class I major-histocompatibility-complex receptors. *Genet. Anal.* **12,** 1–21.
22. Sezerman, U., Vajda, S., Cornette, J., and DeLisi, C. (1993) Toward computational determination of peptide-receptor structure. *Protein Sci.* **2,** 1827–1843.

17

Geometrical Docking Algorithms

A Practical Approach

Haim J. Wolfson and Ruth Nussinov

1. Introduction

The problem of docking of molecules displaying some level of flexibility is extremely important in computational structural biology.

Most state-of-the-art docking techniques have been designed toward approaching the rigid-docking problem ***(1–17)***. These methods have generally been tested on the so-called "bound docking," where the goal is to predict the docked configuration of a pair of molecules whose coordinates have been extracted from the known three-dimensional (3D) structure of their complex ***(18)***. Yet, in the real-life problem, one needs to predict an a priori unknown structure of a complex, given the structures of the separate molecules. The structures of these have been determined separately ("unbound docking"). Even under the best of circumstances, i.e., if there are no major conformational changes in the separate molecules during their association, we still expect minor conformational variations, especially at the docking interface ***(19)***.

The simplest, and most straightforward, way to extend existing docking methodologies to this realistic unbound case is to relax quantitatively certain constraints, handled as parameters in the docking methodologies ***(19)***. These constraints serve to reject unacceptable, false potential docked configurations. In practice, these constraints enable overcoming two hurdles: on the one hand, they allow a certain, liberal extent of penetration of the ligand into the receptor. On the other hand, they enable relaxing the distance thresholds defining proximity of receptor–ligand molecular surfaces, and hence allow the two molecules to be further apart than they should be if they are to be found in a bound association. The obvious downside of such a strategy is the expected dramatic increase

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

in the number of potential docked solutions. Such an increase inevitably results in making the detection of the correct configuration among the false positives extremely difficult. In general, the run time of computer programs implementing such a strategy increases significantly as compared with their stricter-constraints, entirely rigid-docking counterparts. Hence, in order to realistically implement such a strategy, one should first significantly improve the filtering approach of the rigid-docking techniques. An alternative, in principle superior approach, is to take the molecular flexibility into account a priori. Yet, such an implementation may present even larger hurdles, as it may be expected to involve extensive conformational searches. These may result in very long run times.

In this chapter we first describe geometrically based rigid-body docking algorithms. We review the various algorithms, focusing on their advantages and disadvantages. In particular we discuss our own, computer-vision based approaches to the docking problem. We proceed to address the critical, realistic, issue of surface variability. Furthermore, recognition and binding may involve induced-fit, rather than the classically approached case of lock-and-key association. We therefore go on to describe current techniques handling such induced-fit conformational changes, their advantages, and their disadvantages. In particular, we focus on our hinge-bending, induced-fit, robotics-based docking methodology, and some of the currently obtained results.

The problem of receptor–ligand recognition and interaction has two major components: the first involves the 3D geometrical fitting of the molecules, whereas the second requires that the chemical interactions be optimized ***(1,20)***. In this chapter we confine ourselves to the former.

2. Rigid-Body, Geometry-Based Docking

Geometric complementarity is a central consideration in biomolecular recognition. Tightly matching molecular surfaces between interacting, bound molecules yield considerable areas shielded from the solvent. Such interfaces are essential, as they contribute significantly to the stability of the complex via the hydrophobic effect. Complementary geometry reflects van der Waals interactions, which are very sharp at short distances. Hence, a tightly matching interface is a necessary condition for a stable complex. Thus, in turn, screening potential docked conformations with a reasonable geometric complementarity requirement may rapidly yield a smaller subset, serving as input for detailed physical–chemical–biological examination. For larger receptor–ligand systems, such an approach is far more practical than *ab initio* methods.

Geometric docking is extremely complex, owing to the fact that the computational costs increase exponentially with the degrees of freedom of the molecules. For large molecules, containing hundreds, or thousands, of atoms, the number of

potential conformations is extremely large. Thus, any practical docking approach must apply some constraints. Rigid-body approaches, which freeze all degrees of freedom, except three rotations and three translations of the molecule, are frequently adopted. Nevertheless, even when treated as rigid bodies, a thorough search along each of the six degrees of freedom requires a large number of steps. Focusing on the binding sites reduces significantly the complexity of the problem. However, this necessitates an a priori knowledge of these sites.

There have been several rigid-body geometrically based approaches to docking (e.g., **refs. *1–17***). Fifteen years ago, Kuntz and his colleagues have introduced two basic concepts ***(4)***. First, they suggested a convenient way for representing the negative image of the receptor surface, and of the structure of the ligand. The second concept involved matching of distances between the receptor negative-image "spheres" and the ligand-positive image, either atoms or spheres. Because this approach focuses on the docking of the ligand to the largest cluster of intersecting spheres, it centers on the largest concave regions of the receptor. This works best for small-ligand docking, such as drugs, particularly into enzymes, where the small ligands generally bind in the largest cavities. For larger protein–protein docking, which frequently do not dock into the largest depressions on the molecular surface of the receptor ***(21)***, such an approach may encounter difficulties. Other geometrical methods have positioned the surface atoms of the molecules on 3D grids and matched them by rotating and translating one of the grids, seeking the best fit with the grid of the second molecule (e.g., **ref. *6***). The accuracy of grid-based methods depends on the tesselation of the space, and the number of orientations that are explored. Higher accuracy requires sampling of a large number of orientations. On the other hand, too large a number implies a significant increase in complexity. Hence, although finer, grid-based sampling ensures finding the correct, optimal solution, such an approach is frequently impractical. A significantly faster grid-based approach, which utilizes the Fast Fourier Transform, has recently been suggested ***(11)***.

A few years ago we developed geometrically based approaches for molecular docking ***(12–15,19)***. These approaches ***(12,15)*** adapt and implement the Geometric Hashing, computer-vision based algorithm ***(22)***, or geometrically based variants, which are close in spirit ***(13,14,19)***. The algorithms are extremely efficient and utilize entire molecular surfaces. There is no need to predefine the binding sites, although if such information is available, it speeds the execution of these methods substantially. A reasonable number of docked configurations are produced, and the ranking of these potential solutions is acceptable. Adequate performance is achieved for both large protein–protein docking, and for small molecules docking into large, or smaller receptors.

3. Molecular Surface Representation

The problem of protein–protein recognition and docking contains two critical ingredients. The first is the surface representation. The second is the matching of the surfaces. Accurate representation of the features of the molecular surfaces is extremely important for the docking calculations. Even a cursory look at the shapes of the molecular surfaces immediately reveals that they are highly irregular. This characteriztic complicates the task of an accurate, and concise surface description. The discrete surface representation of Connolly ***(23)*** is convenient to handle. However, for the purpose of docking, the drawback of such a representation is that to adequately describe the molecular surface, tens of thousands of surface points may be needed. Matching these points would result in a combinatorial explosion ***(12)***. Hence, a reduction in the number of these points is essential.

There are a couple of approaches carrying out such a task. The first has been devised by Connolly ***(16)***. Connolly has described an elegant method for choosing "critical points." These describe the most prominent features of the molecular shape, in the form of "knobs" and "holes." This is done via a computation of the local convexity at each of the "regular" Connolly surface dots ***(23)***, and selecting the points representing the local minima/maxima of the shape curvature. We have implemented a variant of this approach in one of our docking techniques ***(13,14)***. In parallel, a second approach to reduce the number of points is that developed by Lin et al. ***(24,25)***. The latter surface representation consists of a sparse set of critical points, nicknamed caps, pits, and belts. These points are the face centers, abstracted from the convex, concave, and saddle areas of the Connolly surface. The positions they occupy are key to the shape of the molecular surface, and they are uniquely and accurately defined. The critical points are computed from the faces composing the molecular surfaces. The centroid of each face is determined, and projected unto the surface. The face centers comprise the critical points. For each point we also compute the surface normal and the patch of surface area that the point represents. The set is divided into subsets containing caps and pits (and belts, which are not utilized in the docking) correspondingly originating from the convex, concave (and saddle) faces. Careful examination has shown that surface atoms have six face centers. This procedure removes subatomic details. We have been able to further reduce the number of points, with little or no deterioration in the quality of the surface representation and its efficacy in the matching ***(25)***. In particular, in addition to the critical points, we have drawn extensively on the surface normals, and most recently also on the surface areas, represented by the points.

4. Docking: Computer-Vision-Based Algorithms

Here we outline briefly the methodology and its rationale. An extensive description has been given ***(22,12,15)***. The current application exploits the critical points and their normals ***(15)***.

The algorithm contains two stages. First, we represent the geometric data of one of the molecules (e.g., the ligand) in a table, which allows fast comparison with the geometric data of the second molecule (e.g., the receptor). In such a way one can significantly speed up the docking algorithm at the expense of additional storage memory. Because molecules undergo rotations and translations, the important feature of this methodology is its rotation and translation invariant representation of the coordinates of the critical points in many different reference frames. This (redundant) rotation and translation invariant representation affords matching avoiding the very time-consuming steps that are followed in exhaustive conformational space searches, such as those carried out utilizing grids. Previously, we have used triplets of surface points to define a reference frame ***(12)***. Currently, we build a Cartesian reference frame for each pair of critical points ***(15)***, and the mean of their normals. The coordinates of other critical points within a certain radius are represented in these reference frames and stored in a hash table.

This procedure is carried out for one of the molecules, generally the ligand, which is the smaller of the two. This is the preprocessing stage. The table is organized to allow a direct access during the next, recognition stage. In practice, the address to each table bin consists of the coordinates of the critical point. The information stored in the bin is the identity of the point (to which atom it belongs), and any additional information, such as the chemical nature of the atom, the type of residue, the type of molecule, and the reference frame. Each critical point is stored in many reference frames, as long as they are based on critical points in its vicinity. It is this redundancy that ensures finding a correct solution, even if it involves only partial matching. This feature is critically important, as we assume that the active site is a priori unknown. Hence, we expect only a partial fit. Another important point to note is that the preprocessing is carried out on only one subset of critical points. In general, we utilize the caps of the ligand. In the recognition stage, we consider the pits of the receptor. These subsets suffice to find good docked configurations.

In the recognition stage, the receptor is scanned, and a similar calculation is carried out for the pits. For each reference frame in the receptor, the coordinates of the critical points in its vicinity are calculated, and the hash table is accessed at the address defined by these. The goal is to find ligand caps that have close-enough coordinates (within an error threshold) in their correspond-

ing reference frames. A match, or a vote, is cast for a ligand reference frame if the coordinates are within that error distance.

In addition, further goodness criteria need to be satisfied, such as similar surface normal directions. One may also add here any further chemical requirements, such as complementarity in charges, hydrophobic interactions, hydrogen bonds, and so on. If a ligand reference frame scores a "large enough" number of votes, it is an indication that a superposition of this reference frame with the corresponding receptor frame will result in a docked configuration containing that same number of matched critical point pairs. Utilizing these receptor–ligand critical point pairs, we can compute the 3D rotation and translation, resulting in the best least-squares fit for the frames and their corresponding matching point pairs.

The next step involves the evaluation of the docked configurations.

5. Filtering and Scoring the Docked Configurations

Filtering and scoring the potential docked solutions is an essential step. There are two goals to this routine. First, it is quite possible that although the molecular surfaces of the receptor and the ligand match well in one region, the molecules interpenetrate at other locations. Such solutions need to be filtered out from the list of docked configurations. For the remaining solutions, the routine needs to rank them, according to some goodness criteria. Ideally, these criteria are geometrical and energetic. Here we describe the first of these, which is geometry based. Within these considerations, we employ angular parameters, overlap check, and scoring of the contacts. The implementation of the realistic surface variability depends on the setting of these parameters.

The first consideration utilized in the filtering of the docked solutions is the direction of the normals ***(13,14,19)***. Although the power of the normals has been employed in matching to reduce the combinatorics, by supplying an additional point for building the reference frames, they may also be utilized in the filtering and in the scoring of the goodness of the matches of the pairs of points. In addition, the torsion angles formed by the two planes, i.e., that of the plane defined by the two critical points and one normal, and the second plane determined by the same critical points and the second normal, may be considered as well. In the Geometric Hashing approach ***(15)***, the torsion angle filtering is employed already in the voting (i.e., table-accessing) step, as described. The solutions passing these criteria ***(13–15,19)*** are evaluated by a scoring routine. In order to assess the interpenetration, the receptor is mapped onto a 3D grid. Interior atoms, exterior atoms, and surface points correspondingly designate interior, exterior, and surface voxels ***(14)***. The scoring routine next transforms all ligand atoms by the transformation computed for the matching receptor–ligand point pairs, and maps the ligand atoms onto the same grid. If a ligand

atom falls into a voxel designated as interior voxel, the solution is rejected. The remainder of the solutions are scored and ranked. The quality of a scoring function can be judged by its effectiveness in the filtering and the goodness of its ranking. Ideally, the lower the root-mean-square deviation (RMSD) between the docked and the crystal complex ***(18)***, the higher the ranking that solution would be awarded. The ranking score is computed by awarding surface contact and penalyzing overlaps. Additionally, we may award patches of connected matching surface points and apply a simple hydrophobicity filter ***(19)***.

6. Incorporating Surface Variability: The Bound and Unbound Cases

Because of surface variability, worse fitting of the molecular surfaces of the receptor and of the ligand is expected to occur. However, in reality, the situation would be alleviated by conformational displacements, particularly of surface residues, to optimize the shape complementarity and the intermolecular interactions. These considerations dictate weaker constraints in accepting and in retaining docked configurations. On the other hand, the difficulty is that even for moderately sized protein molecules such a relaxation in determining what should be considered as complementary shape can already result in a very large number of solutions. In particular, the problem may be expected to be more severe if the structures have been determined separately. For example, docking of an immunoglobulin with its antigen (1hfm – 1lym, i.e., IG*G1 fv fragment – lysozyme; or 2hflLH – 1lyz, i.e., IG*G1 fab fragment – lysozyme), with the same definition of shape complementarity as that applied to many protein crystal complexes ***(14)***, results in tens of thousands of docked configurations.

If one scans the obtained complexed conformations, however, it becomes immediately evident that many of these are rather similar, and represent virtually the same docked solution. Hence, many docking approaches adopt clustering schemes. Each cluster represents one docked solution. The difficulty, however, is in the definition of what constitutes a cluster. That is, how different can two docked solutions be and still be considered to represent the same complexed configuration? Frequently the thresholds for the clustering are rather intuitive. A logical strategy may then involve a trade-off: on one hand, to ensure obtaining and retaining the correct docked conformation, we are more liberal in the overlap-filtering constraints. On the other hand, to substantially reduce the number of solutions, we apply generous clustering. To assess the extent of the thresholds that should be applied both in the intermolecular penetrations and in the clustering scheme, we have superimposed the crystal structures of the same molecules when in the bound, and in the unbound configurations ***(19)***. We have inspected the RMSDs of residues that are on the surface in both

Table 1
The Complexes (i.e., Bound Cases) Used for the Rigid Body Protein–Protein Docking

Complex	PDB	Receptor name	Ligand name	Res. in Å
1	1cho	Alpha-chymotrypsin 1-146 (E)	Alpha-chymotrypsin 149-245 (E)	1.8
2	1fdl	IG*G1 fab fragment (LH)	2-Lysozyme (Y)	2.5
3	1tec	Thermitase eglin-c (E)	Leech (I)	2.2
4	1tgs	Trypsinogen (Z)	Pancreatic secretory trypsin inhibitor (I)	1.8
5	2hfl	IG*G1 fab fragment (LH)	Lysozyme (Y)	2.5
6	2kai	Kallikrein a (A13)	Bovine pancreatic trypsin inhibitor (I)	2.5
7	2mhb	Hemoglobin α chain (A)	β chain (13)	2.0
8	2ptc	Beta-trypsin (E)	Pancreatic trypsin inhibitor (I)	1.9
9	2sec	Subtilisin carlsberg (E)	Genetically engineered N-acetyl eglin-c (I)	1.8
10	2sni	Subtihisin novo (E)	Chymotrypsin inhibitor (I)	2.1
11	2tgp	Trypsinogen (Z)	Pancreatic trypsin inhibitor (I)	1.9
12	3hfm	IG*G1 fab fragment (LH)	Lysozyine (Y)	3.0
13	4cpa	Carboxypeptidase	Potato carboxypeptidase a inhibitor (I)	2.5
14	4hvp	HIV-1 protease chain A	Chain B	2.3
15	4sgb	Senine proteinase (E)	Potato inhibitor pci-1 (I)	2.1
16	4tpi	Trypsinogen (Z)	Pancreatic trypsin inhibitor (I)	2.2
17	1abi	Hydrolase alpha thrombin (II)	Chain L	2.3
18	1acb	Hydrolase alpha-chymotrypsin (E)	Eglin C (I)	2.0
19	1cse	Subtilisin carlsberg (E)	Eglin C (I)	1.2
20	1tpa	Anhiydro-trypsin (E)	Arypsin inhibitor (I)	1.9
21	2sic	Subtilisin (E)	Subtilisin inhibitor (I)	1.8
22	5hmg	Influenza virus hemagglutinin chain E	Chain F	3.2
23	6tim	Triosephosphate isomerase chain A	Chain B	2.2
24	8fab	Cab fragment from IGG1 chain A	Chain B	1.8
25	9ldt	Lactate dehydrogenase chain A	Chain B	2.0
26	9rsa	Ribonuclease chain A	Chain B	1.8

The Protein Database (PDB) code of each complex is noted. The chain is given in parentheses next to the description of the receptor and the ligand. The resolution of the complex is noted in the last column.

bound and unbound states; RMSDs that are on the molecular surface in the uncomplexed molecule, but are buried in the interface in the bound state, and those that are in the interior of the molecules in both states. These comparisons provide a reasonable gauge of the extent of movements of surface residues on binding ***(19)***. The drawback of such an approach is the paucity in such pairs of crystal structures. The statistically poor sample size (25 pairs of structures) does not allow the extraction of detailed ranges. However, it still enables the abstraction of a range of values that can be utilized.

This approach reduces considerably the number of docked configurations that are obtained, and hence aids in the ranking of the correct solutions, i.e., those resembling the crystal complexes. **Table 1** presents a list of 26 bound cases that we have utilized in the testing of this approach ***(19)***. All cases are protein–protein complexes. **Tables 2** and **3** present the results we have obtained for these cases. **Table 2** shows the reduction in the number of solutions in the clustered as compared to the unclustered solutions, following three tests: the overlap test, the hydrophobicity filter, and the connectivity. As we can see, the reduction is substantial. The table also lists the CPU times of the matching, in minutes. **Table 3** gives the RMSD of the best solution, and the ranking of the top scoring solution whose RMSD is under 5 Å. The ranking is given for both unclustered and clustered solutions. The rankings are further listed following each of these filters. As can be seen from **Tables 2** and **3**, the quality of these results is highly desirable: the RMSDs are low, the CPU times (on a 66-MHz Intel clone) are short, and the ranking high. Furthermore, the same set of parameters is used and, in particular, the entire molecular surfaces of the receptor and of the ligand are utilized, with no predefinition of the active sites. Further details of the results and of the procedures described here are given in ***(19)***.

Table 4 lists the unbound cases we have utilized in our tests ***(19)***. Nineteen receptor–ligand molecule pairs have been docked. The quality of the results can be assessed by inspection of **Tables 5–7**. **Table 5** gives the number of solutions obtained with the unclustered versus the clustered procedure, as well as the CPU, in minutes of the docking (matching) stage. As in **Table 2**, the number of solutions is listed following each of the filters. **Table 6** presents the ranking of the top scoring solution having an RMSD under 5 Å. **Table 7** gives the best RMSD that Norel et al. ***(19)*** have obtained. While certainly, as might be expected, the quality of the solutions is not as high as that obtained for the bound, complexed cases, it is still very high. Again, as for the bound case, entire molecular surfaces are utilized, with no additional biochemical data regarding the location of the active site. Also, the same set of parameters has been employed as previously. No further tuning is carried out.

Nevertheless, although these results are acceptable, this docking procedure works well for unbound cases as long as no appreciable conformational change

Table 2
The CPU and the Number of Obtained Potential Solutions for the Bound Cases

Complex	PDB	CPU docking	Unclustered solutions			Clustered solutions		
			Overlap	HF	CC	Overlap	HF	CC
1	1cho	3.3	8355	5375	2951	912	713	471
2	1fdl	17.1	63261	35697	16733	4034	3290	2181
3	1tec	4.3	30215	11227	9473	1659	1154	1042
4	1tgs	5.7	22646	6975	5827	1557	941	831
5	2hfl	20.8	65373	39896	20652	3870	3099	2166
6	2kai	4.4	27668	12110	10775	1791	1327	1227
7	2mnhb	14.3	57014	32995	9155	1809	1481	663
8	2ptc	5.3	26616	10163	8843	1798	1134	1027
9	2sec	3.5	26574	9146	7559	1783	1273	1114
10	2sni	4.4	31148	17801	14174	1926	1542	1367
11	2tgp	3.2	19012	7734	6720	1434	916	828
12	3hfm	21.3	85419	45013	21349	4284	3237	2274
13	4cpa	4.1	21240	12040	11975	1659	1320	1310
14	4hvp	2.8	8927	3723	1797	966	720	411
15	4sgb	1.8	11051	4707	4295	1002	642	591
16	4tpi	4.1	22858	9190	7764	1523	1007	889
17	1abi	12.4	8636	4473	4239	1183	814	773
18	1acb	7.6	34628	16074	13544	1698	1256	1121
19	1cse	3.3	24994	10258	8982	1590	1136	1024
20	1tpa	5.1	23203	8298	7374	1606	1026	950
21	2sic	6.3	48164	23156	12914	2125	1689	1229
22	5hmg	35.3	97260	47170	928	4193	3299	329
23	6tim	21.9	97166	42931	1112	3109	2544	351
24	8fab	4.5	28620	6992	159	1879	1158	93
25	9ldt	48.1	78700	42543	101	3670	3157	67
26	9rsa	5.8	33132	9880	3325	1441	1011	511

The second column indicates the PDB code for the complex. The third column notes the CPU time (in minutes) for the docking (matching) step. Docking has been performed on a 486 PC clone, running at 66 MHz. Columns 4–6 indicate the number of potential solutions that have passed the overlap test, the hydrophobicity test (in addition to the overlap filter), and the connectivity filter (in addition to the overlap and hydrophobicity filters), respectively. The overlap test, the connectivity and the hydrophobicity filters have been described in detail by Norel et al. ***(14,19)***. Columns 7–9 show the number of clusters that passed these same filters, respectively.

takes place between the unbound and the bound cases. If, however, a major conformational change does occur between, say, an open, unbound form to a closed, bound one, such a docking protocol would not be able to perform the docking successfully.

Table 3
The Rank of the Best Solution

			Without clustering			With clustering		
Complex	PDB	RMS (Å)	Overlap	HF	CC	Overlap	HF	CC
1	1cho	0.54	1	1	1	1	1	1
2	1fdl	1.50	2899	1994	1932	87	66	20
3	1tec	1.18	5	2	4	1	1	1
4	1tgs	1.14	1	1	1	1	1	1
5	2hfl	1.51	23	30	7	9	9	1
6	2kai	1.17	124	61	49	1	1	11
7	2mhb	0.70	1	1	1	1	1	1
8	2ptc	0.59	1	1	2	1	1	1
9	2sec	2.08	248	37	80	1	1	1
10	2sni	1.07	2	1	4	1	1	1
11	2tgp	0.59	1	1	1	1	1	1
12	3hfm	0.76	103	40	5	2	1	1
13	4cpa	1.02	2	2	2	1	1	3
14	4hvp	2.06	1	1	1	1	1	1
15	4sgb	1.88	71	5	13	1	1	5
16	4tpi	0.52	1	1	1	1	1	1
17	1abi	0.56	1	1	1	1	1	1
18	1acb	0.94	4	1	1	1	1	1
19	1cse	1.32	26	3	23	1	1	2
20	1tpa	0.23	1	1	1	1	1	1
21	2sic	1.11	1	1	5	1	1	1
22	5hmg	1.09	1	1	1	1	1	1
23	6tim	0.50	1	1	1	1	1	1
24	8fab	1.97	3	1	6	2	1	1
25	9ldt	2.52	1	1	1	1	1	1
26	9rsa	1.30	9	3	479	1	1	21

The second column indicates the PDB code for the complex. The third column lists the RMSDs of the interface atoms for the best solution. The rank of the best solution obtained following each of the filters (*see* legend to **Table 4**) is noted for the unclustered and clustered solutions. The lowest-ranking solution with an RMSD <5 Å is listed here. Overlap refers to the overlap test; HF: overlap test and hydrophobicity filter; CC: connectivity filter (in addition to the overlap and hydrophobicity filters).

In the next sections, we address such docking cases, where a significant hinge-bending movements may take place.

7. Flexible Docking Allowing Induced Fit in Proteins

Hinge-bending transitions may occur during molecular recognition and binding ***(26)***. In such cases, movements of whole domains or of small parts may

Table 4
The Unbound Examples

Complex	PDB	Receptor name	Res. (Å)	Ligand name	Res. (Å)
1	1hfm-1lym(A)	IG*G1 fv fragment	Model	Hysozyme (A)	2.5
2	1hfm-1lym(B)	IG*G1 fv fragment	Model	Lysozyme (B)	2.5
3	1tgn-4pti	Trypsinogen	1.6	Trypsin inhibitor	1.5
4	1tgn-5pti	Trypsinogen	1.6	Trypsin inhibitor	1.0
5	1tgn-6pti	Trypsinogen	1.6	Trypsin inhibitor	1.7
6	1tld-4pti	Beta-trypsin	1.5	Trypsin inhibitor	1.5
7	1tld-5pti	Beta-trypsin	1.5	Trypsin inhibitor	1.0
8	1tld-6pti	Beta-trypsin	1.5	Trypsin inhibitor	1.7
9	2hfl-1lyz	LG*G1 Cab fragment	2.5	Lysozyme	2.0
10	2hfl-6lyz	IG*G1 Cab fragment	2.5	Hysozyme	2.0
11	2pka-4pti	Kallikrein a	2.0	Trypsin inhibitor	1.5
12	2pka-5pti	Kallikrein a	2.0	Trypsin inhibitor	1.0
13	2pka-6pti	Kallikrein a	2.0	Trypsin inhibitor	1.7
14	2ptn-4pti	Trypsin	1.5	Trypsin inhibitor	1.5
15	2ptn-5pti	Trypsin	1.5	Trypsin inhibitor	1.0
16	2ptii-6pti	Trypsin	1.5	Trypsin inhibitor	1.7
17	2sbt-2ci2	Subtilisin noVo	2.8	Chymotrypsin inhibitor	2.0
18	Scha(A)-2ovo	Alpha-chymotrypsin (A)	1.7	Ovomucoid third domain	1.5
19	Scha(B)-2ovo	Alpha-chymotrypsin (B)	1.7	Ovomnucoid third domain	1.5

The PDB codes of the receptors and of the ligands are noted in the PDB column. The resolutions of the molecules are noted in columns 4 and 6.

take place at flexible joints. For example, movements of small units have been observed in the T4 lysozyme, in the catabolite gene-activator protein, in the triose phosphate isomerase, and in antibody–antigen binding. Binding of a receptor and a ligand frequently elicits movements of segments of the molecules that are involved. A switch from an open to a closed conformation may both push the water molecules out and trap the substrate or the reaction intermediates. It may also better position the ligand in the receptor pocket ***(27)***.

The frequent occurrence of domain (or part) movements suggests that in seeking to predict docked conformations, such movements ought to be considered. Approaches carrying out rigid docking computations will be successful only in those cases where the movements are relatively small, i.e., within the allowed thresholds.

Previous docking approaches have allowed either induced hinge flexibility in small ligands, such as drugs ***(28–34)***, or partial flexibility in protein receptors ***(35,36)***, e.g., partial flexibility of hydrogen bonding groups. None of the currently available approaches allow domain rotations. The technique we have

Table 5
The Number of the Obtained Potential Solutions

Complex	PDB	CPU docking	Unclustered solutions			Clustered solutions		
			Overlap	HF	CC	Overlap	HF	CC
1	1hfm-1lym(A)	23.7	107638	68859	32986	26275	20352	11475
2	1hfm-1lym(B)	7.9	65727	39489	19058	26703	19243	10685
3	1tgn-4pti	6.5	39816	21084	9321	7865	5234	2619
4	1tgn-5pti	10.6	57614	30224	13189	10272	6864	3453
5	1tgn-6pti	6.3	38276	15030	7008	4346	2779	1455
6	1tld-4pti	4.9	37256	19998	8562	7808	5413	2659
7	1tld-5pti	7.2	53965	29119	12590	10347	7039	3471
8	1tld-6pti	4.9	36711	14828	6798	4374	2922	1512
9	2hfl-1lyz	20.1	132126	93613	42785	24030	19584	10989
10	2hfl-6lyz	25.2	129921	87895	40457	23991	19157	10733
11	2pka-4pti	3.8	41699	24489	10872	8534	6399	3184
12	2pka-5pti	6.1	63264	36681	15531	11325	8495	4222
13	2pka-6pti	3.7	40437	18314	8349	4727	3385	1756
14	2ptn-4pti	5.6	36773	19376	6715	7688	5213	2156
15	2ptn-5pti	8.0	52990	28105	9546	10171	6923	2880
16	2ptn-6pti	5.5	36170	14045	5162	4336	2784	1200
17	2sbt-2ci2	4.3	50908	31285	14235	10170	7800	3582
18	5cha(A)-2ovo	3.3	38024	22322	13018	4502	3392	2194
19	5cha(B)-2ovo	6.1	39941	23019	13932	4611	3439	2289

The second column notes the PDB code for the example and the third column lists the CPU times in minutes needed to complete the docking (matching step; *see* legend to **Table 4**). Columns 4–6 indicate the number of potential, unclustered solutions which have passed the overlap test, the hydrophobicity test, and the connectivity filter, respectively. Columns 7–9 give the number of clusters for each of the groups of potential solutions. *See* footnote to **Table 2** for details.

developed ***(26,37–40)*** allows such motions to exist either in ligands or in receptors, large or small. Furthermore, we allow motions about several hinges simultaneously. We model full 3D rotations at the hinge, where the hinge can be positioned on the backbone, or at any points chosen in space. By choosing a hinge point in space, and allowing complete rotations, rather than rotations about a bond, we implicitly take into account rotations about several consecutive, or nearby, bonds. Addressing local conformational changes is essential if we are to achieve correct docked configurations of practical value.

On the computational side, the problem of addressing geometrical docking, allowing hinge bending, is highly complex. There have been two previous approaches allowing hinge-bending of domains or of smaller molecular subparts. The first approach docks each part separately. In the next step, the

Table 6
Rank of Scoring Solutions Having RMSO Under 5Å

		Solution number		Cluster number	
Complex	PDB	Overlap	CC	Overlap	CC
1	1hfm-1lym(A)	787	999	421	537
2	1hfm-1lym(B)	948	1316	88	281
3	1tgn-4pti	1050	677	71	53
4	1tgn-5pti	845	1045	2	1
5	1tgn-6pti	315	175	2	2
6	1tld-4pti	381	400	40	16
7	1tld-5pti	2615	1414	286	619
8	1tld-6pti	666	466	24	40
9	2hfl-1lyz	960	117	200	110
10	2hfl-6lyz	931	96	461	65
11	2pka-4pti	500	202	356	29
12	2pka-5pti	549	544	67	9
13	2pka-6pti	558	223	58	27
14	2ptn-4pti	363	160	13	9
15	2ptn-5pti	3594	4061	23	34
16	2ptn-6pti	642	480	1	56
17	2sbt-2ci2	1620	1019	154	92
18	5cha(A)-2ovo	30	20	3	11
19	5cha(B)-2ovo	188	86	2	2

The rank of the best solution obtained with and without the connectivity test is given in columns 3 and 4. The rank of the best clusters is noted in the fifth and sixth columns. *See* footnote to **Table 3** for additional details.

separately docked conformations are screened, seeking consistently docked solutions (e.g., **ref. *28***). In such solutions, the two docked parts would be positioned correctly with respect to each other, with the hinge joining them at the correct site, and no overlaps taking place at other locations. A major drawback of such an approach is that it does not use an essential piece of information a priori, i.e., the fact the we know the location of the hinge. Thus, a substantial portion of the conformations that are obtained cannot exist for the real joint-connected molecule. The second approach initially docks one part. The subsequent step involves a full conformational space search, with all rotations allowed about the hinge. This approach is reminiscent of a grid-based search, and is thus extremely time consuming and computationally untractable. Our approach a priori exploits the fact that the different parts belong to the same molecule, and the location of the hinge is known. It further contains global

Table 7
The RMSD of the Best Solution

Complex	PDB	HMS with respect to unbound reference state	RMS with respect to bound reference state	RMS of superimposed unbound on bound
1	1hfm-1lym(A)	2.97	2.43	1.88
2	1hfm-1lym(B)	2.80	3.09	2.18
3	1tgn-4pti	1.85	2.56	2.08
4	1tgn-5pti	1.22	1.92	1.23
S	1tgn-6pti	1.75	2.33	1.59
6	1tld-4pti	5.22	5.71	2.01
7	1tld-5pti	4.71	4.93	1.44
8	1tld-6pti	2.18	2.59	1.57
9	2hfl-1lyz	1.79	2.22	1.31
10	2hfl-6lyz	1.08	1.39	1.18
11	2pka-4pti	3.29	3.91	2.27
12	2pka-5pti	1.21	1.84	1.64
13	2pka-6pti	1.82	2.18	1.63
14	2ptn-4pti	3.53	4.41	2.11
15	2ptn-5pti	3.11	2.85	1.37
16	2ptn-6pti	1.28	1.90	1.66
17	2sbt-2ci2	2.62	2.80	1.62
18	5cha(A)-2ovo	1.49	1.76	1.79
19	5cha(B)-2ovo	1.64	2.22	1.77

Column 2 lists the PDB code. Columns 3 and 4 note the RMS of the best solution. Two RMS values are given, with respect to two reference states. The first is with respect to the matching unbound–unbound state (with the unbound molecules superimposed on their complexed, bound PDB counterparts) and the second RMS is computed with respect to the bound, complexed, PDB solution. The RMS is computed using the interface atoms. *See* **ref. 19** for further details. The last column shows, for comparison, the RMS between the bound and unbound chains.

consistency checks as an integral part of the matching. The matching votes are collectively assembled from all parts of the molecule simultaneously. Hence, this approach finds optimally docked conformations even if one of the parts (say, the smaller one) collects only a relatively small number of votes, whereas a second part achieves a favorable molecular surface complementarity of the two, receptor–ligand molecule-pair. If the two (or more) parts together still score high, their docked configuration would be retained. The position of the hinge is picked manually, at the more flexible joints. A full 3D rotation is allowed at the hinge. This model ***(26)*** is more general than the one with a single rotatable bond, as a rotation about a bond has only one degree of freedom.

8. The Hinge-Bending, Robotics-Based Algorithm

This method is computer vision and robotics based. As with our rigid-body approach, it represents and matches the molecules in a transformation-invariant manner ***(37)***. There are two stages: preprocessing and recognition. In the preprocessing step, the smaller (ligand) molecule is considered. The hinge location is defined to be the origin of a 3D Cartesian coordinate frame, called the "ligand frame." Its orientation is set arbitrarily. For each noncollinear triplet of interest points (e.g., the critical points describing the molecular surface), we define a unique triplet-based Cartesian frame. Denote the triplet points by a, b, and c. Define the origin at a, the direction of the x-axis as the direction of the line from a to b, the direction of the z-axis as the direction of the cross product of the vectors ab with ac, and the direction of the y-axis as the direction of the cross product of the unit vectors in the x and z directions. This is the "triplet frame." The ordered triplet of the triangle side lengths serves as an address to a hash table, where the ligand and the part identification are stored. In addition, the hash table bin contains the transformation between the triplet frame and the ligand frame.

In the recognition stage, all noncollinear triplets of critical points describing the receptor molecular surface are considered. For each triplet, their Cartesian frame is calculated (*see* above), and the triangle sides computed. The lookup (hash) table is accessed according to the triangle side lengths. The prerecorded transformation stored at the corresponding bin is applied to the receptor-based triplet frame. This results in a computed "candidate ligand frame." The origin of the candidate ligand frame is the "candidate hinge location." A vote is next cast for the location and the orientation of the candidate, hinge-centered, ligand frame.

At the end of this recognition stage, after all receptor triangles have been examined, the accumulator of votes is searched, seeking high scoring hinge locations. The hinge location defines the 3D translation that the ligand has to undergo in this candidate docking. The rotations are computed in the next, verification, stage. The high-scoring hinge locations are determined according to the minimal percentage value of the number of votes received by the highest scoring hinge ***(26)***.

In the verification step both the interpart (intramolecular) and intermolecular penetration is checked. The criteria and considerations here are, in general, as described previously for the rigid-body docking, although the details that have actually been implemented differ. Similar in spirit, although not in details, here too, clustering of the transformations is applied. The ranking is computed by calculating a contact percentage.

This algorithm directly exploits the fact that both parts of the molecule share the same hinge. This has been done by locating the origin of the ligand refer-

ence frame at the hinge. Hence, both parts contribute votes to a reference frame at the same (hinge) location, even though the orientation of the two parts with respect to each other may be different. In particular, this enables picking up a correct docked conformation even if one of the parts has only a small number of matched receptor–ligand critical point pairs. That would occur if overall the conformation still scores high. On the other hand, had we docked each part separately, such a solution might have been overlooked.

Here we have described the algorithm for a single hinge, with the hinge defined to be in the ligand. Nevertheless, it can be easily seen that the situation is symmetrical. We have already implemented it for the case where the hinge is in the receptor. In addition, it has already been implemented and applied to the double-hinge case ***(26)***. There, instead of having one ligand frame, we have two (or more) frames, each centered at a different hinge. During the preprocessing stage, for each ligand triplet in a single part, the transformations to the two (or more) ligand frames are computed and stored. The recognition stage is unchanged with one exception: for each receptor triplet we tally votes for as many frames as the number of transformations stored in the table bin.

9. Some Results Obtained Allowing Hinge Bending

This method has already been applied successfully to a number of bound, and of unbound, cases achieving quality docked configurations rapidly ***(26,37–40)***.

The location of the hinge has been defined by a comparison of the open, unbound conformation and the closed, bound one. However, different locations in the general vicinity have also been tested, with similar results. This suggests that the technique is quite robust. Specifically, we have applied our method to five bound complexes and one unbound case. The bound cases include the HIV-1 protease complexed with the U-75875 inhibitor, the dihydrofolate reductase complexed with methotrexate and separately with NADPH, lactate dehydrogenase complexed with NAD-lactate, and a FAB fragment of an IgG antibody complexed with a peptide antigen (residues 69–87 of myohemerythrin ***[40]***). In each of these cases, the flexible docking has been carried out with hinge bending in the ligands. In all cases we have reproduced the crystal binding modes. The average RMSD we have obtained for the correct solution is 1.4 Å and the average CPU time on a Silicon Graphics SGI-Challenge R8000 machine, is 1 min. The crystallographically correct solutions rank high. In all cases, additional predictive binding modes are obtained as well ***(26,37–40)***.

In addition to the foregoing complexes, we have also examined thoroughly the calmodulin receptor (CaM) and its peptide ligand ***(26)***. There, we have allowed flexibility either in the ligand, or in the receptor. Moreover, in the ligand, either one or two hinges have been allowed. Different locations of the hinges have also been tested, obtaining consistently similar results. Again, low RMSDs

have been obtained (i.e., for the single-hinge case in the peptide ligand, 2.53 Å for the first part; 1.17 Å for the second; for the double-hinge case, 2.03 Å for the first part; 0.98 Å for the second, and 1.03 Å for the third part.) Similarly, the CPU times are short. A full description is given elsewhere ***(26)***.

10. Conclusions

Protein molecules are dynamic entities. Indeed, this is a critical aspect of their biological function. It is therefore extremely important to consider molecular flexibility when we develop a realistic docking scheme. Yet implementing such a realization is a very difficult task in practice.

Here we have described several currently available approaches to handle molecular flexibility. These include techniques for handling flexibility in the ligands, and partial flexibility in the receptor, as well as our hinge-bending computer-vision, robotics based techniques. Nevertheless, in addition to bending and flexing the molecules, molecular surface movements, as described here, implemented within the framework of either the rigid-body algorithms, or within the hinge-bending ones, are also a route to consider. For those cases where the movements are not large, such approaches may prove very useful. On the other hand, if larger-scale movements need to be enabled, approaches such as the hinge-bending ones described here could prove a method of choice. We are currently extending these to handle more than two hinges in either of the molecules.

Here we have described geometrical algorithms and geometrical-based filtering of the obtained solutions. The next, essential, step is the chemistry of the interacting molecules ***(41)***. This critical aspect is not addressed in this chapter.

Acknowledgments

The authors thank our many colleagues and students who have contributed and enabled this work: Drs. J. V. Maizel, S. L. Lin, D. Xu, D. Fischer, R. Norel, B. Sandak, C.-J. Tsai, and A. Li. In particular, this chapter draws on recent work by our students R. Norel (**ref. *19***) and B. Sandak (**refs. *26*** and ***40***). The research of R. Nussinov has been sponsored by the National Cancer Institute, Department of Health and Human Services (DHHS), under contract no. 1-CO-74102 with SAIC. The content of this chapter does not necessarily reflect the views or policies of the DHHS, nor does mention of trade names, commercial products, or organizations imply endorsement by the U. S. Government. The research of H. J. Wolfson and R. Nussinov in Israel has been supported in part by a grant from the Israel Science Foundation administered by the Israel Academy of Sciences, by grant no. 95-00208 from the Binational Science Foundation, Israel, by a grant from the Rekanati foundation, and by the Magnet and the Minerva funds.

References

1. Cherfils, J., Duquerroy, S., and Janin, J. (1991) Protein-protein recognition analyzed by docking simulations. *Proteins: Struct. Funct. Genet.* **11,** 271–280.
2. Cherfils, J. and Janin, J. (1993) Protein docking algorithms: simulating molecular recognition. *Curr. Opin. Struct. Biol.* **3,** 265–269.
3. Lawrence, M. C. and Colman, P. M. (1993) Shape complementarity at protein/protein interfaces. *J. Mol. Biol.* **234,** 946–950.
4. Kuntz, I. D., Blaney, J. M., Oatley, S. J., Langridge, R., and Ferrin, T. (1982) A geometric approach to macromolecule-ligand interactions. *J. Mol. Biol.* **161,** 269–288.
5. Goodsell, D. S. and Olson, A. J. (1990) Automated docking of substrates to proteins by simulated annealing. *Proteins: Struct. Funct. Genet.* **8,** 195–202.
6. Jiang, F. and Kim, S. H. (1991) "Soft docking": matching of molecular surface cubes. *J. Mol. Biol.* **219,** 79–102.
7. Wang, H. (1991) Grid-search molecular accessible surface algorithm for solving the protein docking problem. J. Comp. Chem. **12,** 746–750.
8. Shoichet, B. K. and Kuntz, I. D. (1991) Protein docking and complementarity. *J. Mol. Biol.* **221,** 79–102.
9. Walls, P. H. and Sternberg, M. J. E. (1992) New algorithm to model protein–protein recognition based on surface complementarity. *J. Mol. Biol.* **228,** 277–297.
10. Kasinos, N., Lilley, G. A., Subbarao, N., and Haneef, I. (1992) A robust and efficient automated docking algorithm for molecular recognition. *Protein Eng.* **5,** 69–75.
11. Katchalski-Katzir, E., Shariv, I., Eisenstein, M., Friesem, A. A., Aflalo, C., and Vakser, I. A. (1992) Molecular surface recognition: determination of geometric fit between protein and their ligands by correlation techniques. *Proc. Natl. Acad. Sci. USA* **89,** 2195–2199.
12. Norel, R., Fischer, D., Wolfson, H. J., and Nussinov, R. (1994) Molecular surface recognition by a computer vision based technique. *Protein Eng.* **7,** 39–46.
13. Norel, R., Lin, S. L., Wolfson, H., and Nussinov, R. (1994) Shape complementarity at protein–protein interfaces. *Biopolymers* **34,** 933–940.
14. Norel, R., Lin, S. L., Wolfson, H., and Nussinov, R. (1995) Molecular surface complementarity at protein–protein interfaces: the critical role played by surface normals at well placed, sparse, points in docking. *J. Mol. Biol.* **252,** 263–273.
15. Fischer, D., Lin, S. L., Wolfson H. J., and Nussinov, R. (1995) A geometry-based suite of molecular docking processes. *J. Mol. Biol.* **248,** 459–477.
16. Connolly, M. (1986) Shape complementarity at the hemoglobin $\alpha_1\beta_1$ subunit interfaces. *Biopolymers* **25,** 1229–1247.
17. Helmer-Citterich, M. and Tramontano, A. (1994) PUZZLE: a new method for automated protein docking based on surface shape complementarity. *J. Mol. Biol.* **235,** 1021–1031.
18. Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, E. F. Jr., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1997) The protein

databank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.* **112,** 535–542.
19. Norel, R., Lin, S. L., Xu, D., Wolfson, H., and Nussinov, R. (1998) Molecular surface variability and induced conformational changes upon protein–protein association, in *Structure, Motion, Interaction and Expression of Biological Macromolecules. Proceedings of the Tenth Conversation* (Sarma, R. H. and Sarma, M. H., eds.), Adenine Press, Albany, NY, pp. 31–51.
20. Janin, J., Miller, S., and Chothia, C. (1988) Surface, subunit interfaces and interior of oligomeric proteins. *J. Mol. Biol.* **204,** 155–164.
21. Peters, K. P., Fauck, J., and Frommel, C. (1997) The automatic search for ligand binding sites in proteins of known three-dimensional structure using only geometric criteria. J. Mol. Biol **256,** 201–213.
22. Nussinov, R. and Wolfson, H. (1991) Efficient detection of motifs in biological macromolecules by computer vision techniques. *Proc. Natl. Acad. Sci. USA* **88,** 10,495–10,499.
23. Connolly, M. L. (1983) Analytical molecular surface calculation. *J. Appl. Cryst.* **16,** 548–558.
24. Lin, S. L., Nussinov, R. Fischer, D., and Wolfson, H. (1994) Molecular surface representation by sparse critical points. *Proteins* **18,** 94–101.
25. Lin, S. L. and Nussinov, R. (1996) Molecular recognition via the face center representation of molecular surface. J. Mol. Graphics **14,** 78–97.
26. Sandak, B., Wolfson, H., and Nussinov, R. (1998) Flexible docking allowing induced fit in proteins: insights from open to closed conformational isomers. *Proteins* **32,** 159–174.
27. Gerstein, M., Lesk, A. M., and Chothia, C. (1994) Structural mechanisms for domain movements in proteins. *Biochemistry* **33,** 6739–6749.
28. DesJarlais, R. L., Sheridan, R. P., Dixon, J. S., Kuntz, I. D., and Venkataraghavan, R. (1986) Docking flexible ligands to macromolecular receptors by molecular shape. J. Med. Chem. **29,** 2149–2153.
29. Leach, A. R. and Kuntz, I. D. (1992) Conformational analysis of flexible ligands in macromolecular receptor sites. J. Comp. Chem. **13,** 730–748.
30. Knegtel, R. M. A., Antoon, C. R., Boelens, R., and Kaptein, R. (1994) Monty: a Monte Carlo approach to protein–DNA recognition. *J. Mol. Biol.* **235,** 318–324.
31. Clark, K. P. and Ajay. (1995) Flexible ligand docking without parameter adjustment across four ligand receptor complexes. *J. Comp. Chem.* **16,** 1210–1226.
32. Welch, W., Ruppert, J., and Jain, A. N. (1996) Hammerhead: fast, fully automated docking of flexible ligands to protein binding sites. *Chem. and Biol.* **3,** 449–462.
33. Rarey, M., Kramer, B., Lengauer, T., and Klebe, G. (1996) A fast flexible docking method using incremental construction algorithm. *J. Mol. Biol.* **261,** 470–489.
34. Jones, G., Willet, P., Glen, R. C., Leach, A., and Taylor, R. (1997) Development and validation of a genetic algorithm for flexible docking. *J. Mol. Biol.* **267,** 727–748.
35. Leach, A. R. (1994) Ligand docking to proteins with discrete side-chain flexibility. *J. Mol. Biol.* **235,** 345–356.

36. Jones, G., Willet, P., and Glen, R. C. (1995) Molecular recognition of receptor sites using a genetic algorithm with a description of desolvation. *J. Mol. Biol.* **245,** 43–53.
37. Sandak, B., Nussinov, R., and Wolfson, H. J. (1995) An automated computer-vision and robotics based technique for 3-D flexible biomolecular docking and matching. *Comp. Appl. BioSci.* **11,** 87–99.
38. Sandak, B., Wolfson, H. J., and Nussinov, R. (1996) Hinge-bending at molecular interfaces: automated docking of a dihydroxyethylene-containing inhibitor of the HIV-1 protease, in *Proceedings of the Ninth Conversation in Stereodynamics* (Sarma, R. H. and Sarma, M. H., eds.), Adenine Press, Albany, NY, 233–252.
39. Sandak, B., Nussinov, R., and Wolfson, H. J. (1996) Docking of conformationally flexible proteins, in *Seventh Symposium on Combinatorial Pattern Matching*, Laguna Beach, CA, and *Lecture Notes in Computer Science*, Springer Verlag, New York. **1075,** 271–287.
40. Sandak, B., Nussinov, R., and Wolfson, H. J. (1998) A flexible method for biomolecular structural recognition and docking allowing conformational flexibility. J. Co.
41. Lengauer, T. and Rarey, M. (1996) Computational methods for biomolecular docking. *Curr. Opin. Struct. Biol.* **6,** 402–406.

18

Protein–Protein Docking

Generation and Filtering of Complexes

Michael J. E. Sternberg, Henry A. Gabb, Richard M. Jackson, and Gidon Moont

1. Introduction

Knowledge of the three-dimensional (3D) structure of a protein–protein complex provides insights into the function of the system that can guide, for example, the systematic design of novel regulators of activity. However, at the end of 1997, there were more than 5000 protein structures in the Brookhaven databank (PDB) but less than 200 sets of coordinates for protein–protein complexes. This disparity is reminiscent of the protein–sequence/protein–structure gap and similarity motivates the development of computational methods for structure prediction. This chapter describes the strategy to start with the coordinates of the two molecules in their unbound states and then computationally model the structure of the bound complex including the conformational changes on association. For reviews of the field of protein docking *see* **refs. *1–3***.

We first describe the strategy recently developed in our laboratory that implements a docking study in the following stages (*see* **Fig. 1**):

1. Generation of series of docked complexes using the rigid-body approximation.
2. Application of known distance constraints, particularly details of the binding sites in one or both proteins.
3. Screening the docked structures generated by (1) and (2) to identify the correct solution by removing false positives.
4. Refinement of the rigid body structure to consider conformational change combined with further screening of possible solutions.

This approach is discussed in the context of related algorithms developed by other groups.

From: *Methods in Molecular Biology*, vol. 143: *Protein Structure Prediction: Methods and Protocols*
Edited by: D. Webster © Humana Press Inc., Totowa, NJ

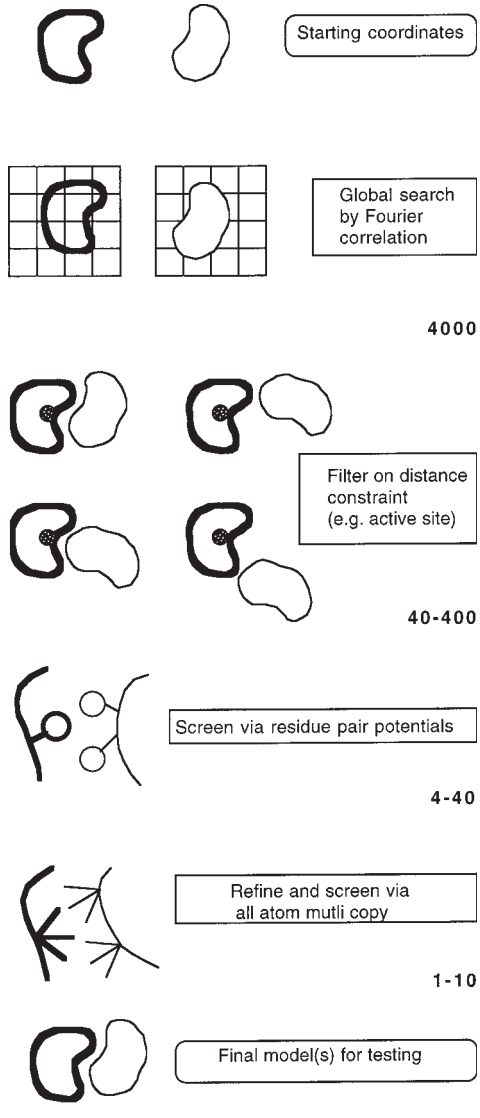


Fig. 1. Schematic of the strategy for generating and screening docked protein complexes. The number of complexes generated refer to the test cases of enzyme inhibitors (*see* **Table 1**). FTDOCK performs a global search for the docking of the two starting molecules. Only complexes that do not have unfavorable electrostatic complementarity are considered and 4000 complexes ranked by surface complementarity and generated. The active site of the molecule shown as a speckled sphere is taken as known and used as a distance constraint (FILTR). The next step (RPDOCK) is screening based on residue–pair potential using C_β atoms (C_α for glycine). Then, side-chain rotamers are sampled by a multicopy method that also performs a limited rigid-body search (MULTIDOCK).

2. Materials

The software for the strategy is available from our Web site under the home page `http://www.bmm.icnet.uk` and is free to academic users. The initial stage of performing a rigid-body docking is implemented in the program FTDOCK as reported in Gabb et al. ***(4)***. As described in **Subheading 3.1.**, FTDOCK performs the rigid-body docking using the Katchalski-Katzir et al. algorithm ***(5)*** that uses Fast Fourier Transforms (FFT) to search the translational binding space of two rigid molecules. At present, there are two implementations of FTDOCK: one uses the FFT routines from Numerical Recipes Software ***(6)*** and can be implemented on a variety of hardware platforms and the other exploits the more efficient Silicon Graphics library functions but is specific to this platform. FTDOCK can run the FFT as parallel processes on a Silicon Graphics Challenge computer and a complete search of binding space can be completed using eight R10000 processors in a few hours. The subsequent application of distance constraints is presently implemented in a postprocessor program FILTR, but this is liable to amendment in subsequent implementations of the package.

Rigid-body docking is implemented in several other docking protocols and these procedures can provide suitable starting models for the subsequent filtering and refinement stages. Here we refer to Chapter 17 by Nussinov and Wolfson in this volume and also the global-range molecular-matching (GRAMM) implementation of the Katchalski-Katzir et al. algorithm by Vakser ***(7)*** (`http://reco3.musc.edu/gramm`).

The initial screening of docked complexes can be performed using residue pair potentials and is implemented in a package called RPDOCK (*see* Moont et al. ***[8]***). The subsequent refinement using multicopy conformations for side chains is implemented in MULTIDOCK (*see* Jackson et al. ***[9]***). For a single 300-residue complex on a R10000 processor, RPDOCK requires less than 10 s, and MULTIDOCK without solvent <10 min and with solvent around 30 min.

3. Methods

3.1 Rigid-Body Docking by Fourier Correlation Theory

The initial step is the rigid-body docking of the two molecules to generate a set of complexes and is performed by FTDOCK (*see* **ref. *4*** for details). The approach is based on the Fourier correlation methodology proposed by Katchalski-Katzir et al. ***(5)***. Two molecules A and B are placed onto 3D grids each of size $N \times N \times N$ and each node l,m,n is assigned a value

$$a_{l,m,n} = \begin{cases} 1 \text{ for grid points on the surface} \\ \rho \text{ for the core} \\ 0 \text{ for the outside of the molecule} \end{cases} \quad (1)$$

where ρ has a negative value (we use –15) for grid nodes within the surface layer of thickness t (we use between 1.5 and 1.2 Å)

$$b_{l,m,n} = \begin{array}{l} 1 \text{ for the molecule} \\ 0 \text{ for outside of molecule} \end{array} \quad (2)$$

The complementarity of shape $c_{\alpha,\beta,\gamma}$ between the molecules is then evaluated from

$$c_{\alpha,\beta,\gamma} = \sum_{l=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} a_{l,m,n} \cdot b_{l+\alpha,m+\beta,n+\gamma} \quad (3)$$

where α, β, and γ are the translational shift of molecule B with respect to A for a given relative orientation of the two grids. A high value for c denotes a complex with good surface complementarity, whereas a negative value for c denotes a complex with penetration into the core of molecule A. Low or zero values for c denote little or no overlap of the surfaces. The use of a surface thickness t provides a softness to the docking that accommodates local conformational changes on complex formation. Calculation of c requires N^3 multiplications and additions for every N^3 α,β,γ shifts. However, the use of discrete Fourier transforms reduces the calculation of c to the order of $N^3 \ln N^3$. For a global search of the docking of two proteins, c must be calculated for a series of relative orientations of one molecular grid with respect to the other. A total of 6912 orientations are required for angular deviations of 15° and 22,105 for 10°.

In general, molecular recognition in protein complex formation includes both shape complementarity and electrostatic effects. Accordingly, we introduced into the Fourier correlation approach a treatment of electrostatics. The charge–charge interaction is evaluated from point charges of one molecule interacting with the potential from the other molecule sampled at grid points. A key aspect of the treatment of electrostatics is to provide a smoothness to the energy landscape eliminating artificially highly favorable or very unfavorable interactions that result from the rigid-body docking, without treatment of conformational changes. Charges are assigned to the atoms of molecule A and the electrostatic potential evaluated outside and as the surface of the molecule from

$$\phi_{l,m,n} = \sum_{j} (q_j / \varepsilon(r_{ij}) r_{ij}) \quad (4)$$

where $\phi_{l,m,n}$ is the potential at node l,m,n (position i), q_j is the charge on atom j, r_{ij} is the distance between i and j (with a minimum value of 2 Å to avoid artificially large values of the potential) and $\varepsilon(r_{ij})$ is a distance-dependent dielectric function. Inside at grid nodes corresponding to core molecule A, $\phi_{l,m,n}$ is zero. For molecule B, charges are assigned to neighboring grid points giving a function $q_{l,m,n}$. The electrostatic interaction $e_{\alpha,\beta,\gamma}$ for a shift of α,β,γ is calculated from

$$e_{\alpha,\beta,\gamma} = \sum_{l=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} \phi_{l,m,n} \cdot q_{l+\alpha,m+\beta,n+\gamma} \quad (5)$$

This function is analogous to that for shape complementarity and accordingly is also evaluated by Fourier correlation for computational efficiency. Benchmarking showed that the electrostatic function is best used as a binary filter, removing any complex with an unfavorable (>0) interaction. For favorable electrostatic interactions, the complexes are scored solely by the shape complementarity.

The global search is performed by storing the three most favorable complexes for all translations from a scan with a given orientation of the molecules. After all orientations are sampled, the top set (typically 4,000) of complexes are examined and filtered. In a test system of six enzyme–inhibitor complexes, typically a list of hundreds of complexes would need to be examined after a global search starting from unbound components to identify a model that is close (<2.5 Å root-mean-square [RMS] for the C_α atoms at the interface) to the correct structure (*see* **Subheading 3.6.**). This lack of selectivity arises in part from the problems associated with rigid-body docking.

3.2. Distance Constraints

Knowledge of the location of the binding site on one, or both, protein drastically reduces the number of possible complexes. This information is often available, e.g., knowledge of the active site of an enzyme when docking its inhibitor. There can also be experimental information from studies such as mutagenesis or crosslinking. Alternatively, there are computational methods to predict the location of the binding site from coordinates. Examination of clefts, charged sites, and potential hydrogen bonding groups in a protein can suggest a binding site ***(10)***. Recently, Jones and Thornton ***(11)*** have developed a procedure based on patch analysis that reports a 66% success rate for predicting binding sites. In addition, phylogenetic analysis mapping conserved sequences onto a structure can prove to be a valuable tool to predict functional sites ***(12,13)***. Such distance constraints can therefore be applied to list of docked complexes and in our package a procedure FILTR is supplied.

3.3. Use of Residue Pair Potentials in Screening Docked Complexes

Generally less than 100 solutions need to be examined after a global search, that has been followed by filtering on distance restraints. Methods have therefore been developed for subsequent screening. These approaches need to model the interactions stabilizing the complex employing functions that are sufficiently robust to cope with the limitations in modeling the structure of the complex. Over the last few years, empirically derived residue–residue pair potential have been widely used in protein modeling, especially fold recognition (*see* Chapter 7 in this volume by Jones and Chapter 8 by Reva and Finkelstein and review by

ref. *14*). The theory is that because these potentials are derived from observations, they will incorporate the dominant thermodynamic effects. Moreover, when evaluated at the residue, rather than the atom level, the functions can provide a smooth energy landscape that is not dominated by small changes in atomic positions. These considerations have led us to develop an approach for screening docked complexes using pair potentials.

The frequencies F_{ab} of pairings between residues of type a and b having a C_β–C_β distance (for Gly C_α) less than a cutoff d_{cut} is evaluated from a nonredundant database of protein chains. These observed frequencies are compared to those for a random state and the model used for this state is the based on the molar fractions, i.e., it is purely compositional. Let n_a and n_b be the total occurrences of residues of types a and b and T the total number of all pairings,

$$T = \sum_{a=1}^{20} \sum_{b=1}^{20} F_{ab} \tag{6}$$

The molar expected frequency for the a–b pairing is given by

$$E_{ab} = \mathrm{T} \cdot \left(n_a / \sum_{a=1}^{20} n_a \right) \cdot \left(n_b / \sum_{b=1}^{20} n_b \right) \tag{7}$$

A log-odds score for a pairing of residue types a and b is derived:

$$S_{ab} = \log_{10} (F_{ab} / E_{ab}) \tag{8}$$

The total score for a complex is obtained by summing the S_{ab} values for all residue pairings between the two molecules with the distance less than d_{cut}. In addition, the potential can only be evaluated between residues that have relative accessibility above a cutoff of (A_{cut}) to exclude buried side chains. This total score can then be divided by the total number of contacting pairs of residues. High-scoring complexes are evaluated by this function to be more favorable. This screening procedure is implemented in the program RPDOCK. For enzyme inhibitors, the recommended approach uses d_{cut} of 12 Å, A_{cut} of 5%, and divides the total score by the number of contacts. These values would, a priori, be recommended for other protein–protein complexes. However, for antibody–antigen complexes, which tend to have a flatter interacting surface, a different set of parameters is suggested (use d_{cut} of 17 Å, A_{cut} of 20%, and as before divide by the total score by the number of contacts).

In this approach, the stabilities of the different complexes are directly evaluated from the log–odds ratio. In many applications, these types of log–odds ratios are converted to a potential of mean force by the application of Boltzmann's principle. However, the validity of this approach has been questioned ***(15,16)*** and, accordingly, we simply treat the scores as a statistical measure of relative stabilities. In our evaluation of this strategy we also considered an alternate model for the random state based on the contact expected frequency

(16), but this was found to be less effective in screening docked complexes. In addition, pair potentials were derived for individual atoms or atomic- function groups ***(17)***. These proved less suitable for screening complexes than the residue-based molar-fraction approach. The poor performance of atomic pair potentials is probably due to their sensitivity to the atomic positions, which need to be more precise than can be obtained from a complex generated by rigid body docking.

3.4. Molecular Mechanics Refinement of Protein Interfaces Incorporating Solvation

An additional method for screening solutions is to use a molecular mechanics energy function. We have developed a method to refine protein–protein interfaces that models the effect of side-chain conformational change, solvation, and limited rigid-body movement of the interacting molecules ***(9)***. As well as being a possible screening method, it is also a conformational refinement procedure, producing information on the energy contributions of specific residue contributions to protein binding.

The proteins are described at the atomic level by multiple copies of side chains on a fixed peptide backbone modeled according to commonly occurring side-chain conformations from the library of Tuffery et al. ***(18)***. The surrounding solvent environment is described by "soft" sphere Langevin dipoles for water (*see* next paragraph) that interact with the protein. Energy refinement is based on a two-step process in which (1) a probability-based conformational matrix of the protein side chains is refined iteratively by a mean-field method. A side chain interacts with the protein backbone and the probability-weighted average of the surrounding protein side chains and solvent molecules (2) The resultant protein conformations then undergo rigid-body energy minimization to relax the interface. Steps (1) and (2) are repeated until convergence of the interaction energy.

The "soft" sphere Langevin dipole (LD) model reproduces the solvation-free energy of a solute with its surrounding water environment. The interacting water molecule is represented by a van der Waals particle, an LD (modeling the electrostatic interaction), and a field-dependent hydrophobic energy. The model is based on that developed by Luzhkov and Warshel ***[19]***). In addition to a realistic solvation model we use a self-consistent mean-field approach to optimize protein side-chain conformations, given the main-chain atom coordinates ***(20,21)***. This describes a protein of N residues whose main-chain coordinates (N, C_α, C, O, and C_β) are fixed. A residue side chain, i, (with the exception of Gly, Ala, and Pro) has a discrete number, K_i, of conformations (rotamers). Therefore, side-chain degrees of freedom can be defined by a conformational matrix CM of dimension, N by max(K_i), where each rotamer, k, has a probabil-

ity of CM(i,k), bounded by the condition that the sum of the probabilities for a given residue, i, must be equal to 1.

The object of the mean-field approach is to determine the most probable set of side-chain rotamers from a limited total number of rotamers. The potential of mean force, $E(i,k)$, on the kth rotamer of residue, i, is given by;

$$E(i,k) = V(\chi_{ik}) + V(\chi_{ik},\chi_{mc}) + \sum_{j=1}^{N}\sum_{j\neq i}^{Ki}\sum_{l=1} \mathrm{CM}(j,l)V(\chi_{ik},\chi_{jl}) + E_{\mathrm{sol}}(i,k) \quad (9)$$

where V is the potential energy, χ_{ik} are the coordinates of atoms in rotamer k of residue i and χ_{mc} are the coordinates of atoms in the protein main chain. The first term represents the internal energy of the rotamer. The second term represents the interaction energy between the rotamer and all the main-chain atoms. These two values are constant for a given rotamer on a given main chain. The third term represents the interaction energy between the rotamer and all the rotamers of other residues weighted by their respective probabilities. The fourth term $E_{\mathrm{sol}}(i,k)$ represents the potential of mean force acting at rotamer, k, of residue, i, due to the surrounding solvent environment. Each residue rotamer has a number, M_i,k_j, of precalculated interactions with all surrounding solvent sites. For an LD in conflict with a rotamer the probability of the site is dependent on the probability, $\mathrm{CM}_{\mathrm{conflict}}(j,l)$, of the rotamer in conflict. If no rotamer is in conflict the probability of the site is 1. Thus the additional solvation term is given by:

$$E_{\mathrm{sol}}(i,k) = \sum_{\mathrm{LD}=1}^{Mi,k} [1 - \sum_{j=1}^{N}\sum_{j\neq i}^{Kj}\sum_{l=1} \mathrm{CM}_{\mathrm{conflict}}(j,l)]\, G(\chi_{ik} + \chi_{mc}, \chi_{\mathrm{LD}}) \quad (10)$$

with

$$0 \leq [1 - \sum_{j=1}^{N}\sum_{j\neq i}^{Ki}\sum_{l=1} \mathrm{CM}_{\mathrm{conflict}}(j,l)] \leq 1 \quad (11)$$

where $G(\chi_{ik} + \chi_{mc}, \chi_{\mathrm{LD}})$ is the free energy of interaction between the main chain plus side-chain atoms of a given rotamer and the "soft" sphere LD (van der Waals, electrostatic, and hydrophobic components).

Given the effective potentials acting on all K_i possible rotamers of residue, i, the probability of the rotamer can be calculated according to the Boltzmann principle as

$$\mathrm{CM}(i,k) = e^{-E(\mathrm{i,k})/RT} / \sum_{k=1}^{Ki} e^{-E(i,k)/RT} \quad (12)$$

where R is the Boltzmann constant and T is the temperature. The values of CM(i,k) are substituted back into the equation describing E(i,k) and its new value recalculated. This process is repeated until values of CM(i,k) converge. The predicted structure corresponds to the highest probability rotamer for each residue. Following each complete cycle of side-chain mean-field optimization, rigid-body minimization was performed on the resultant coordinates of the

interacting protein molecules (note that solvation cannot be included in this step). The larger molecule is kept stationary, whereas the six degrees of freedom (three rotational and three translational) of the smaller molecule are moved according to the path determined by the derivatives to minimize the intermolecular interaction energy.

The objective potential energy function used thoughout is a molecular mechanics force field that includes the "soft" sphere LD model for solvation in the mean-field optimization step. The protein–protein interaction energies are constrained to be within boundaries and therefore produce a smoother energy surface. Unfavorable van der Waals interactions are truncated to a maximum value of 2.5 kcal/mol. This was chosen to correspond with an electrostatic interaction scheme in which a minimum allowed distance separation between two interacting charges q_i and q_j is set so that atom pairs that come closer than allowed are rescaled to realistic values.

3.5. Application to Modeling Protein–DNA Complexes

Predictive docking of protein–DNA complexes is considered to be a more difficult problem than protein–protein complexes, as the DNA tends to undergo substantial conformational change on association and the highly charged DNA backbone can dominate in approaches to model the electrostatic component in molecular recognition. However, the foregoing docking protocol can be modified to tackle systems, such as repressor–DNA complexes, that do not have a gross conformational change on association.

FTDOCK has been applied to modeling repressor–DNA ***(22)***. A specific charge set was developed for DNA that damps the phosphate charges and exaggerates the partial charges on the chemical groups in the DNA helix groove. For the protein, the main chain, the fully charged side chains, and Asn, Gln, and His are assigned charges. In general, for distance filtering, there often is knowledge of the recognition base sequence, and sometimes residues on the protein that interact with the DNA have been identified. Empirical potentials have also been derived analogously to quantify amino acid–nucleotide interactions. The best parameters for screening were found to use a molar fraction model for the random state with d_{cut} of 13 Å but using a sparse matrix that only scored interactions with charged or polar amino acids (C, D, E, H, K, N, Q, R, S, and T). For refinement, there is a requirement to model the conformational changes in both the DNA and the protein. For this step, the reader is referred to the MONTY procedure developed by Kaptein's group ***(23,24)***.

3.6. Sample Results

The protein–protein procedure has been evaluated on six enzyme inhibitors and two antibody–antigen complexes starting with both molecules as unbound

coordinates and on a further two systems starting with bound antibody (HyHEL5 and HyHEL10) docking to unbound antigen ***(4,8,9)***. A correct prediction is taken as a complex with an RMS deviation of the C_α atoms at the interface of no more than 2.5 Å. FTDOCK was run for a global search followed by applying the distance constraint that one of the three active site residues must interact with the inhibitor or one of the antibody-combining loops interacts with the antigen. **Table 1** gives the number of solutions in the list produced by FTDOCK and the number of correct predictions in the list. The subsequent columns give the rank of the first correct solution after FTDOCK alone, then screening the FTDOCK results with pair potentials (RPDOCK) and then screening the FTDOCK results by multicopy refinement without solvent (MULTIDOCK). Finally, the result of first ranking the FTDOCK results by PRDOCK, taking the top 10% of this list for the enzymes and the top 40% for the antibodies, and then ranking these by MULTIDOCK is given. For the enzyme inhibitors, this approach leads to the need to examine no more than four alternatives with the exception of subtilisin with its inhibitor for which no correct prediction was generated in the initial FTDOCK scan (*see* **Fig. 2** for predicted docked complex). For antibody–antigen complex modeling is poorer, possibly as a result of the lower binding affinities in these systems.

The procedure has also been benchmarked on eight repressor–DNA complexes starting with unbound protein coordinates (except for one repressor) and model-built B-DNA. In general, the results ***(22)*** show that predictive docking can yield a limited number of repressor–DNA complexes that can be used, e.g., for the design of subsequent experiments.

3.7. Other Procedures to Screen Docked Protein Complexes

There are several computational methods to predict binding free energies in biological systems, but we focus our overview to methods that have been applied specifically to protein docking applications, as not all such methods are sufficiently robust and/or computationally tractable for the protein–protein docking problem. It is generally accepted that, although contact score or surface area burial can successfully discriminate between 95–99% of the structures generated by a typical docking algorithm, it remains unable to discriminate between the remaining small percentage of solutions (typically 100–1000 structures) (e.g., Shoichet and Kuntz's study ***[25]***). We have presented two different but complementary approaches to screening docked solutions. These methods are essentially independent of each other, however, as shown discrimination is improved by combining the results of the pair-potential and molecular mechanics refinement methods to give a consensus answer.

There are, however, many other scoring methodologies that are not dependent purely on contact scores–surface area burial. This includes the principle

Table 1
Discrimination Provided by Generation and Filtering Docked Protein–Protein Complexes

System	Total no. after FTDOCK	N ≤ 2.5 Å in FTDOCK list	Rank FTDOCK	Rank RPDOCK	Rank MULTI-DOCK	Rank RPDOCK and MULTI-DOCK
αCHYN-HPTI	94	1	3	1	2	1
αCHY-ovomucoid	86	5	11	3	1	1
Kallikrein-BPTI	363	18	130	5	2	1
Subtilisin-CHY I	26	2	8	1	12	2
Subtilisin-subtilisin I	—	—	—	—	—	—
Trypsin-BPTI	228	8	16	7	26	4
D1.3-lysozyme	694	2	168	34	235	84
D44.1-lysozyme	586	5	39	18	108	42
HyHEL5-lysozyme	516	2	226	97	31	23
HyHEL10-lysozyme	756	5	62	169	13	4

For details, *see* **Subheading 3.6**. αCHYN, α-chymotrypsinogen; αCHY, α-chymotrypsin; HPTI, human pancreatic trypsin inhibitor; BPTI, bovine pancreatic trypsin inhibitor; CHYI-chymotrypsin inhibitor; subtilisin I, subtilisin inhibitor D1.3. D44.1, HyHEL5, and HyHEL10 are monoclonal antibodies (for details of coordinate files see **ref. *4***). RPDOCK was run using the recommended values for enzyme-inhibitor and antibody–antigen complexes (*see* **Subheading 3.3.**). Some degenerate identical complexes included our earlier studies have been excluded from the table.

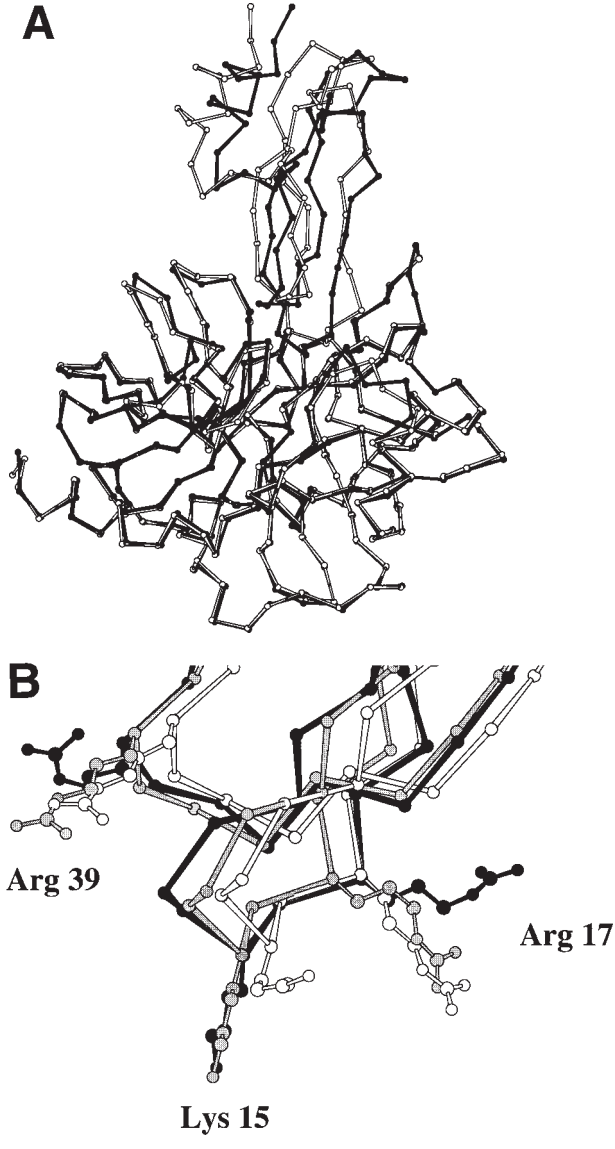


Fig. 2. Predicted docking of BPTI to kallikrein. **(A)** The predicted docking by FTDOCK of bovine pancreatic trypsin inhibitor (BPTI) to kallikrein (below) shown as a superposition of the C_α chain trace. Predicted from FTDOCK in white and X-ray in black. **(B)** The inhibitory loop of BPTI with three critical side chains is shown as it is docked to kallekrein (not shown). In white are the results from FTDOCK, in gray the results after refinement by MULTIDOCK, and in black the X-ray structure of the complex. The most important side chain that buries deep into the active site of kallikrein has its conformation improved by MULTICOPY.

that surface area burial is proportional to the free energy of binding if account is taken of the nature of the constituent interfaces. This is the basis of empirically derived atomic solvation parameters ***(26,27)***, which have been used extensively in protein–protein recognition. Recently, Wallqvist and Covell ***(28)*** have extended this concept by using a statistically derived atom–atom surface burial scheme from observations of surface burial by atoms across the interface of known enzyme-inhibitor complexes. Indeed the subject of statistically derived atom–atom- ***(17)*** and residue–residue ***(29)***-based pair potentials as applied to protein–protein docking ***(8)*** is a relatively unexplored field. However, atom–atom pair potentials have been employed to estimate the binding free energy of small molecules to proteins (e.g., **ref.** ***30***). Furthermore, multiple linear regression methods have also been used to derive atom–atom potentials (e.g., **ref.** ***31***) and used to estimate protein-small molecule interactions where the parameters are estimated using the crystal structures and experimental binding constants.

Alternatively, molecular mechanics-based energy functions have been developed with varying levels of sophistication. Energy minimization has been applied with some success ***(25,32)***, as have Monte Carlo docking methods using grid-based potentials ***(33–35)***. Furthermore, attempts to treat the desolvation effects on binding in addition to interaction potential have been developed. Cummings et al. ***(36)*** and Weng et al. ***(37)*** used atomic solvation parameters (discussed earlier), whereas we ***(38)*** developed a continuum approach, where the free energy change is the sum of the electrostatic free energy (calculated using the Poisson–Boltzmann equation) and a surface area based hydrophobic free energy. Abagyan and Totrov ***(39)*** have developed a modified image approximation to calculate the electrostatic solvation contribution to binding, which has also been used in docking applications. Indeed, several of the methods we have discussed also take into account the loss in conformational entropy on binding due to freezing out side-chain motions, and where this has been combined with a conformational search methodology, encouraging results have been reported in protein systems where conformational changes are fairly limited.

4. Conclusions

Over the last few years substantial progress has been made toward solving the protein-docking problem. For systems such as enzyme inhibitors with a high binding constant, predictive docking aided by knowledge of an active site can suggest a list of a few complexes that can provide models for testing. Our studies suggest that antibody–antigen complexes may be harder to model. Possibly this is due simply to the lower free energy of association or it may also be a consequence of clonal selection dictating the antibody structure that binds to

the antigen, whereas evolution has selected an inhibitor that is optimal (or near optimal) for binding to the enzyme. Some types of protein–DNA complexes are also proving amenable to predictive docking. Clearly starting with rigid bodies will limit the success of the approach to systems with substantial conformational change on association.

Comparison of different docking approaches remains difficult. Of particular value in evaluating strategies are the two blind trials of docking that have been held ***(40,41)***. The continued testing of algorithms against a wide range of targets will assist in the development of computational methods that are robust and based around a sound understanding of the protein-docking problem.

5. Notes

5.1. Rigid-Body Docking (FTDOCK)

1. Docking is performed on grid-based representations of the structures, which are easier to handle computationally and mathematically. Each molecule is reduced to a discrete function (i.e., discretized) that approximately describes its 3D structure. The accuracy of the approximation is determined by the resolution of the grid, which is subject to two constraints. First, the Fast Fourier Transform (FFT) used by the program requires that grid dimensions be powers of two (i.e., 2, 4, 8, 16, 32, 64, 128, etc.). Second, FTDOCK is written in Fortran 77, which lacks dynamic memory allocation. Grid sizes must be fixed at compile time rather than run time. Future versions of the program will remove these restrictions (Fortran 90 and a better, more portable FFT will be used). In the meantime, the user can improve grid resolution by removing parts of the molecule not involved in binding. In antibodies, e.g., only the variable portion is involved in antigen binding and the constant regions can safely be removed.
2. In order to score shape complementarity, it is necessary for FTDOCK to delineate the surface and core of the molecules during discretization. There are several constraints on the surface thickness. First, it cannot be smaller than the grid spacing. Furthermore, if grid resolution is too coarse then the distinction between surface and core is poor. However, as the surface layer of the model increases, the ability to score real shape complementarity diminishes. This leads to predicted complexes with high correlation scores but considerable surface overlap. There are two possible solutions to this problem. First, decrease the value of core grid nodes in the discrete function representing the molecule. This has the effect of increasing the penalty for surface penetration. Second, use a computer with sufficient power and memory to handle larger grids. The latter option is recommended.
3. When grid resolution is not a problem, surface thickness can be used to modulate scoring stringency. A thin surface layer is less tolerant of overlap in the structures, whereas a thicker surface layer softens the scoring function. For example, if the user feels that there will be little conformational change on association, a thinner surface layer is recommended. Similarly, if the available structures are only solved to low resolution, a thicker surface layer should be used.

5.2. Refinement of Protein Interfaces (MULTIDOCK)

1. When screening a large number of putative docked complexes (>50), it is sensible to proceed with refinement without the inclusion of solvent. First, the time saving is substantial (*see* **Subheading 2.**). Second, although inclusion of solvent does appear to enhance the ranking of nativelike solutions, it does not turn a nativelike solution with a poor *in vacuo* energy into a high-ranking solution. Hence solvation can be employed as a final screen on say the top 50 solutions.
2. The total number of residues included in the simulation is presently limited to 450. However, the inclusion of only a limited number of interface residues can speed up the calculation, i.e., the smaller the number of residues treated by the multicopy representation in the interface the faster the calculations. In testing on several systems, a cutoff for inclusion of residues in the mobile interface region (i.e., for residues whose C_β atoms are within a given cutoff of any C_β of the other molecule) of ≥10 Å gave similar results to longer distances. Also, atom–atom and residue–residue cutoff distances can be manipulated for the purpose of either speed or accuracy.
3. The protein ATOM records should be checked for errors before running the program, as nonstandard amino acids are not supported. Residues with the incorrect number of atoms for a given side chain will cause the program to halt. The user must either truncate the residue, to say, alanine (and change the residue name to reflect the atoms present), or rebuild the side chain (generally recommended).

References

1. Janin, J. (1995) Protein-protein recognition**.** *Prog. Biophys. Molec. Biol.* **64,** 145–166.
2. Shoichet, B. K. and Kuntz, I. D. (1996) Predicting the structure of protein complexes: a step in the right direction**.** *Chem. Biol.* **3,** 151–156.
3. Sternberg, M. J. E., Gabb, H. A., and Jackson, R. M. (1998) Predictive docking of protein–protein and protein-DNA complexes. *Curr. Opin. Struct. Biol.*, in the press.
4. Gabb, H. A., Jackson, R. M., and Sternberg, M. J. E. (1997) Modelling protein docking using shape complementarity, electrostatics and biochemical information**.** *J. Mol. Biol.* **272,** 106–120.
5. Katchalski-Katzir, E., Shariv, I., Eisenstein, M., Friesem, A. A., Aflalo, C., and Vakser, I. A. (1992) Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques**.** *Proc. Natl. Acad. Sci. USA* **89,** 2195–2199.
6. Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992) *Numerical Recipes in FORTRAN — The Art of Scientific Computing*. 2nd ed. Cambridge University Press, Cambridge, UK, pp. 490–602.
7. Vakser, I. A. (1997) Evaluation of GRAMM low-resolution docking methodology on the hemagglutinin-antibody complex**.** *Proteins* **Supplement 1,** 226–230.
8. Moont, G., Gabb, H. A., and Sternberg, M. J. E. (1999) Use of pair potentials across protein interfaces in screening predicted docked complexes. *Proteins* **35,** 364–373.
9. Jackson, R. M., Gabb, H. A., and Sternberg, M. J. E. (1998) Rapid refinement of protein interfaces incorporating solvation: application to the docking problem. *J. Mol. Biol.* **276,** 265–285.

10. Laskowski, R. A., Luscombe, N. M., Swindells, M. B., and Thornton, J. M. (1996) Protein clefts in molecular recognition and function. *Protein Sci.* **5,** 2438–2452.
11. Jones, S. and Thornton, J. M. (1997) Prediction of protein–protein interaction sites using patch analysis. *J. Mol. Biol.* **272,** 133–143.
12. Lichtarge, O., Bourne, H. R., and Cohen, F. E. (1996) An evolutionary trace method defines binding surfaces common to protein families. *J. Mol. Biol.* **257,** 342–358.
13. Pazos, F., Helmercitterich, M., Ausiello, G., and Valencia, A. (1997) Correlated mutations contain information about protein–protein interactions. *J. Mol. Biol.* **271,** 511–523.
14. Vajda, S., Sippl, M., and Novotny, J. (1997) Empirical potentials and functions for protein folding and binding. *Curr. Opin. Struct. Biol.* **7,** 222–228.
15. Thomas, P. D. and Dill, K. A. (1996) Statistical potentials extracted from protein structures: how accurate are they? *J. Mol. Biol.* **257,** 457–469.
16. Skolnick, J., Jaroszewski, L., Kolinski, A., and Godzik, A. (1997) Derivation and testing of pair potentials for protein folding. When is the quasichemical approximation correct? *Protein Sci.* **6,** 676–688.
17. Melo, F. and Feytmans, E. (1997) Novel knowledge-based mean force potential at atomic level. *J. Mol. Biol.* **267,** 207–222.
18. Tuffery, P., Etchebest, C., Hazout, S., and Lavery, R. (1991) A new approach to the rapid determination of protein side-chain conformations. *J. Biomol. Struct. Dyn.* **8,** 1267–1289.
19. Luzhkov, V. and Warshel, A. (1992) Microscopic models for quantum mechanical calculations of chemical processes in solution: lD/AMPAC and SCAAS/AMPAC calculations of solvation energies. *J. Comput. Chem.* **13,** 199–213.
20. Koehl, P. and Delarue, M. (1994) Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. *J. Mol. Biol.* **239,** 249–275.
21. Lee, C. (1994) Predicting protein mutant energetics by self-consistent ensemble optimization. *J. Mol. Biol.* **236,** 918–939.
22. Aloy, P., Moont, G., Gabb, H. A., Querol, E., Aviles, F. X., and Sternberg, M. J. E. (1998) Modeling repressor proteins binding to DNA. *J. Mol. Biol.* **33,** 535–549.
23. Knegtel, R. M. A., Antoon, J., Rullmann, C., Boelens, R., and Kaptein, R. (1994) MONTY: a Monte Carlo approach to protein-DNA recognition. *J. Mol. Biol.* **235,** 318–324.
24. Knegtel, R. M. A., Boelens, R., and Kaptein, R. (1994) Monte Carlo docking of protein-DNA complexes: incorporation of DNA flexibility and experimental data. *Protein Eng.* **7,** 761–767.
25. Shoichet, B. K. and Kuntz, I. D. (1991) Protein docking and complementarity. *J. Mol. Biol.* **221,** 327–346.
26. Eisenberg, D. and McLachlan, A. D. (1986) Solvation energy in protein folding and binding. *Nature* **319,** 199–203.
27. Horton, N. and Lewis, M. (1992) Calculation of the free energy of association for protein complexes. *Protein Sci.* **1,** 169–181.
28. Wallqvist, A. and Covell, D. G. (1996) Docking enzyme-inhibitor complexes using a preference-based free-energy surface. *Proteins* **25,** 403–419.

29. Sippl, M. J. (1990) Calculation of conformational ensembles from potentials of mean force. An approach to the knowledge-based prediction of local structures in globular proteins**.** *J. Mol. Biol.* **213,** 859–883.
30. Verkhivker, G. M. and Rejto, P. A. (1996) A mean field model of ligand-protein interactions: implications for the structural assessment of human immunodeficiency virus type 1 protease complexes and receptor-specific binding**.** *Proc. Natl. Acad. Sci. USA*, **93,** 60–64.
31. Bohm, H. J. (1994) The development of a simple empirical scoring function to estimate the binding constant of a protein-ligand complex of known three-dimensional structure. *J. Comput. Aided Mol. Des.* **8,** 243–256.
32. Cherfils, J., Duquerroy, S., and Janin, J. (1991) Protein-protein recognition analyzed by docking simulation**.** *Proteins* **11,** 271–280.
33. Goodsell, D. S. and Olson, A. J. (1990) Automated docking of substrates to proteins by simulated annealing**.** *Proteins* **8,** 195–202.
34. Hart, T. N. and Read, R. J. (1992) A multiple-start Monte Carlo docking method**.** *Proteins* **13,** 206–222.
35. Stoddard, B. L. and Koshland Jr, D. E. (1993) Molecular recognition analyzed by docking simulations: the aspartate receptor and isocitrate dehydrogenase from *Escherichia coli*. *Proc. Natl. Acad. Sci. USA* **90,** 1146–1153.
36. Cummings, M., Hart, T., and Read, R. (1995) Atomic solvation parameters in the analysis of protein–protein docking results. *Protein Sci.* **4,** 2087–2099.
37. Weng, Z., Vajda, S., and Delisi, C. (1996) Prediction of protein complexes using empirical free energy functions**.** *Protein Sci.* **5,** 614–626.
38. Jackson, R., M. and Sternberg, M. J. E. (1995) A continuum model for protein–protein interactions: application to the docking problem. *J. Mol. Biol.* **250,** 258–275.
39. Totrov, M. and Abagyan, R. (1994) Detailed *ab initio* prediction of lysozyme-antibody complex with 1. 6 Å accuracy. *Nat. Struct. Biol.* **1,** 259–263.
40. Strynadka, N. C. J., Eisenstein, M., Katchalski-Katzir, E., Shoichet, B., Kuntz, I., Abagyan, R., Totrov, M., Janin, J., Cherfils, J., Zimmerman, F., Olson, A., Duncan, B., Rao, M., Jackson, R., Sternberg, M., and James, M. N. G. (1996) Molecular docking programs sucessfully determine the binding of a beta-lactamase inhibitory protein to TEM-1 beta-lactamase**.** *Nat. Struct. Biol.* **3,** 233–238.
41. Dixon, J. S. (1997) Evaluation of the CASP2 docking section**.** *Proteins* **Supplement 1,** 198–204.