ALIGNMENT ALGORITHMS REVISITED: ALIGNMENT ALGORITHMS FOR LOW SIMILARITY PROTEIN SEQUENCE COMPARISONS[1]

Michael J. Wise
Department of Genetics
University of Cambridge
Cambridge CB2 3EH, UK
mw263@cam.ac.uk

Abstract

The Smith-Waterman local alignment algorithm is the method of choice for protein database searches because it is often able to detect remote homologues for a query protein sequence. However, it is also well known that the reliability of this algorithm degrades sharply for proteins with low similarity to a given query - so-called "twilight zone" matches. In these situations, global alignments are often employed, based largely on anecdotal evidence. This study re-examines the efficacy of local versus global alignment algorithms. Among other results, the Smith-Waterman algorithm is found to be most effective when two proteins have a common domain (i.e. belong to the same subgroup) or have the same function. However, when only weak relationships exist, global methods are more effective than local ones. In addition, global methods provide a somewhat different point of view to local methods and can therefore be used in addition to local methods to improve search accuracy, even when higher level matches are possible.

1. Introduction

Viewed abstractly, the alignment of two protein sequences is a mathematical mapping from the two sequences (represented by strings of IUPAC codes) to a score which is a measure of the strings' similarity; the scores allow matches to be ranked from most similar to least. Viewed this way, the process is little different from other approximate matching database searches, e.g. when using one of the Internet search engines. However, in the case of sequence alignments one further assumption is also made: high similarity scores are taken to imply that the sequences are evolutionarily related. In other words, high similarity scores imply homology and the associations between pairs of amino acids from the two sequences form a biologically meaningful alignment. The assumption of evolutionary relatedness finds expression, for example, in the amino acid substitution matrices (e.g. the PAM matrices). However, one significant problem remains: what is to be inferred when similarity scores are low, in which case homology cannot be inferred and alignments are meaningless? Bork and Koonin suggest that global alignments can be employed to reduce "noise" and improve "signal" [2] - advice that is common anecdotally but for which evidence is scarce.

This study examines the efficacy of a number of different algorithms for use in protein database searches. However, rather than the conventional search for homology, a different, somewhat weaker question will be addressed: which algorithm is better able to predict whether two sequences have some property in common, such as structure-class or function.

The basis for these judgements is the protein structure domain database SCOP [8], in which proteins are understood as sequences of "domains", i.e. polypeptides which can fold independently.

There have been three distinct generations of sequence alignment algorithms, viz. those based on: Longest Common Subsequence Algorithm, Needleman-Wunsch Algorithm and the Smith-Waterman Algorithm

The first system to address the sequence alignment problem - the Needleman-Wunsch algorithm [9] - used dynamic programming. A similar approach evolved in the computer science literature, where the problem came to be known as the Longest Common Subsequence. (LCS) problem. Summarising the LCS problem, if S is a string, a subsequence of string S is formed by taking elements in order from S, where zero or more elements are ignored before another is taken. (In contrast, *substrings* allow no gaps.) The LCS of two strings S and T is the sequence of elements common to the two strings such that no longer sequence is available (though there may be multiple sequences with the same, maximal length). There are many discussions of LCS in the literature, e.g. [5]. Note that the use of "sequence" in LCS should not be confused with the biologist's use of the same term, which is a string nucleotides or amino acids. Except in the case of the LCS algorithm, "sequences" are of the biological sort.

The Needleman-Wunsch approach differs from the standard definition of LCS in that it adds the concept of scoring matrices. That is, rather than only scoring 1 for an exact match, and 0 for anything else, both exact matches and close mismatches (*conservative substitutions*) are scored. The algorithm is therefore inherently less efficient

1. A longer version of this paper may be found at www.bio.cam.ac.uk/˜mw263/ftp/doc/align_full.pdf.

than LCS because alternate paths must be examined. It is also worth noting that the original Needleman-Wunsch matrix only used positive values, based on the number of base changes in each codon, as opposed to more recent scoring matrices, which use both positive and negative values. Ref. [9] also details experiments with both zero and non-zero gap penalties. Both LCS and Needleman-Wunsch algorithms produce *global alignments*, i.e. the alignments span the entirety of both input sequences.

The form of alignment algorithm currently in use, Smith-Waterman algorithm [11], comes from the observation that the Needleman-Wunsch algorithm returns biologically implausible alignments; they simply contain too many gaps. While the Needleman-Wunsch algorithm added scoring matrices and pay-once gap penalties to the LCS framework, the Smith-Waterman algorithm further added the notion of alignment scoring threshold - typically zero - below which an alignment is terminated. With affine gap penalties and/or scoring matrices which include negative values for mismatches, alignment scores may be reduced to zero part way through an alignment. The Smith-Waterman algorithm therefore produces *local alignments*. The choice of gap creation and gap extension penalties can have a substantial bearing on the results returned by the Smith Waterman algorithms.

Previous projects have compared a number of alignment implementations, though usually based solely on the Smith-Waterman algorithm. For example, [3] compares a direct implementation of the Smith-Waterman algorithm, SSEARCH3, distributed as part of FASTA 3.0 suite of programs, and the approximate Smith-Waterman implementations, FASTA 3, BLAST and a revision of BLAST, WU-BLAST2. Although the methodologies used in [3] are entirely different to those employed here, some interesting correspondences occur.

2. Systems used in the Study

2.1 The Databases

The databases against which tests have been run were drawn from SCOP, which is a classification of (mainly) protein structures found in the PDB structures database. The SCOP database, as of May 1997, had 7,015 entries, each of which provides information about one domain in a particular structure described in PDB. When the non-protein references were removed and the different domains for multi-domain proteins are set against their corresponding proteins, the SCOP database covered 5,226 proteins. For each of these proteins, the corresponding sequence was obtained. A minor problem is that PDB (and hence SCOP) contains many sequences which are identical for all but a very small number of amino acids. (Such entries typically record mutations.) While it has been important in this study to have the full range of possible matches represented - from very closely related to totally unrelated - the results would be skewed if the many duplicates or near duplicates were retained. For this reason, the database was pruned in the following ways: after accepting only the 20 naturally occurring amino acids sequences less than 30aa were removed. In addition, duplicate sequences differing by fewer than three amino acids where removed. This scheme is more permissive than the most permissive of the PDB-select series of databases [7]. Nonetheless, when the sieving was completed only 2,390 sequences remained. It should also be noted that the final database included multi-domain proteins.

The SCOP database records the following information about each domain: its Class (CL), Common Fold (CF), Super Family (SF), Family (FA) and Protein Domain (PR). The arrangement of this information is hierarchic (from the most general CL to the most specific PR), so I shall refer to them as *descriptors* and refer to *descriptor levels*.

Corresponding to the reduced sequence database, a domains database was created listing the CL/CF/SF/FA/PR memberships for the (distinct) domains in each protein. To save space, the descriptions used by SCOP were (consistently) replaced with unique integers and the 5 integers concatenated into string. Numbering is unique with respect to a given descriptor level, CL, CF, etc. For example:

1all_A	1.1.1.2.13
3sdp_A	1.2.5.6.20 4.230.335.506.20

2.2 Algorithms and Similarity Measures

The alignment algorithms examined in this study are listed below. Notice that each calculates one or more *similarity measures* which are equal to or less than the number of amino acids in the shorter of the two input sequences. While much sophisticated work has been done on the statistical distributions underlying the Smith Waterman algorithm, little is known about the properties of the other similarity measures to be described below, so in order to compare like quantities, only raw scores have been used in these experiments.

2.2.1 Smith-Waterman The Smith-Waterman implementation used was SSEARCH3. The default gap creation and gap elongation penalties were used, namely: -12 and -2, respectively, together with the default substitution matrix, BLOSUM50. Parts of the program not relevant to the present study were removed, and the output was modified to return two results: The number of exact matches (identities) in the highest scoring (local) alignment (referred to later as SW_idents or SWI in tables), and the count of identities (as above) plus the number of conservative substitutions (SW_matches or SWM).

2.2.2 Needleman-Wunsch SSEARCH3 was also used for these computations, except that the gap penalties were both set to zero. The use of zero gap penalties differentiates this implementation from other Needleman-Wunsch implementations, including the commonly used program GAP (part of the GCG suite of sequence analysis tools). Zero gap penalties were chosen in the belief that this is closer in spirit to the Needleman-Wunsch algorithm as described in [9], with its use of zero penalties or pay once penalties. Contrast this with GAP which, like the Smith-Waterman implementation, has both gap creation and gap elongation penalties. The only difference between GAP and the Smith-Waterman algorithm is that the latter has a threshold below which matching ceases. The Needleman-Wunsch algorithm used in this study therefore implements one option studied by Needleman and Wunsch and provides a greater contrast between the Needleman-Wunsch and Smith-Waterman algorithms.

Corresponding to the two results being returned for the Smith-Waterman implementation, the Needleman-Wunsch implementation returned NW_idents (NWI) and NW_matches (NWM).

2.2.3 Longest Common Subsequence The implementation of the LCS algorithm used in this study is due to [10] and is extremely efficient. A single result is returned: the number of identities in an LCS alignment (LCS_idents, or LCS1). A second, slightly altered implementation of the Rick algorithm was used to produce the LCS of sequences which have been converted to overlapping dipeptides. That is, the lexicon now contains 400 elements and matching substrings therefore involve a minimum of 2 amino acids. The resulting similarity measure is called LCS_2aa_idents, (or LCS2).

2.2.4 Alignment of Structure Tokens Because at least one level of judgement involves considerations of protein structure, another possible source of similarity measures when one only has sequence information are mappings of amino acids into tokens representing secondary structure elements. In particular, the GORIV suite [6], among many others, takes input protein sequences and returns sequences containing the structure-tokens H (corresponding to a part of an alpha-helix), E (beta-strand) and C (coil). Two similarity measures were computed based on alignments of sequences of structure-tokens.

GOR_match_LCS (abbreviated GLCS)
An LCS alignment is done between the pairs of structure-token sequences.

GOR_match_SW (abbreviated GSW).
Recognising that genuine secondary structure elements generally span several amino acids, a SW alignment was undertaken using a gap creation penalty of 0 and gap elongation penalty of -4. The zero gap creation penalty was chosen because GOR often introduces spurious single secondary structure elements, while the -4 elongation penalty was chosen to force a preference for contiguous alignments.

In both cases the unit matrix was used (i.e. exact matches only).

2.2.5 Lengths of the Sequences It has long been known that the length of the input sequences can be a factor in the resulting alignment scores. The two sequence lengths were therefore also reported: Length_S1 (abbreviated L1), and Length S2 (L2), with Length_S1 always the longer of the two.

2.3 Experimental Methodology

The experimental methodology has been to assume that we do not know anything about the sequences in the database described above, and to align each sequence against every other using a number of different algorithms, in order to determine which algorithm, or combination of algorithms, is better able to predict whether two sequences have domains with the same Class, Common Fold, Super Family, Family or Protein Domain. That is, are matches (identical descriptors) and non-matches (differing descriptors) for the various descriptor levels in the domains database correctly predicted by sequence alignment scores. Note that, while SCOP is structured on the understanding that the five description levels form a hierarchy, for these experiments the five levels were treated independently.

To test a particular sequence similarity measure's ability to predict matches at the various descriptor levels, a *classifier* was constructed for each similarity measure. That is, each sequence was compared with every other using the algorithms described earlier and values recorded for the various similarity measures. The machine-learning/classification system Ripper [4], a rule induction system, was then applied to the similarity measures to construct rules. However, being unnormalised, the rules themselves are of no interest. More important are the error-rates produced when the rules were tested; the error-rate for the rules derived for a given similarity measure provides evidence for the predictive power of the similarity measure once the error-rate, due to the classifier itself, has been discounted.

In order to construct classifiers, the 2,390 sequences in the sequence database (and the corresponding domains database) were first divided into a training and a test subset. To balance the subsets, the sequences were sorted by length and allocated alternately to the subsets. Then, within the respective subsets, each sequence was aligned with all the others and the numbers of matches and non-matches were recorded. A significant problem was the great imbalance in numbers of non-matches versus matches. To counter this bias, *stratified* samples have been used in building and testing the classifiers. That is, the number of matches is balanced by an equal number of non-matches, randomly chosen from descriptor levels lower in the hierarchy. For example, 4,497 training set matches at the SF level are balanced by an equal number of non-matches drawn from the CF, CL and total-non-match levels. Without stratification, the classifiers would only ever return the highly over-represented decision-class, *non-match*.

In other words, the rules generated by application of the classifier to the training set similarity measure values (for a given algorithm) were then tested using the similarity measures due to the same algorithms applied to the corresponding disjoint testing database, and the error-rates reported.

3. Experiment 1: Basic Tests

This experiment examines the basic efficacy of each metric at predicting a shared domain for the various descriptor levels. At the same time, the issue of variability in the results is examined.

There are two sources of variability in the methodology used in this study: variability due to the use of stratification (i.e. descriptor-level training and testing databases can vary with the samples taken), and variability inherent in the construction of classifiers based on noisy data.

3.1 Methods

To get an estimate of the impact of the two sorts of variability, each similarity measure was used singly to predict each of the descriptor levels, 50 experiments in all.

To examine the impact of classification variability, the experiments were run on a stratified database in which the training and testing sets were pooled, and 10-fold cross validation was used for the CF, SF, FA and PR descriptor levels. Because of its much larger size, only 5-fold cross validation was used for the CL descriptor level. In k-fold cross validation, the set is split into k partitions; k-1 are used to construct the classifier while the remaining partition is used to test the classifier. The classification error is averaged over the k trials, together with estimates of the variability in the classification error. These values are reported in the second and third columns of the table below.

The other source of variability is due to stratification. In the second pair of results columns below, the experiments were repeated with independent training and test sets each time drawn from restratified databases. The restratification was repeated 10 times (and was independent of the database used for the cross-validation experiments).

3.2 Results

For compactness, in the results table below, and those on following pages, only the top three results are generally shown.

Experiment		Xval		Strat.	
		Mean	SD	Mean	SD
		%Error		%Error	
CL	L2	40.47	0.10	41.46	0.4
	GLCS	41.22	0.04	41.2	0.03
	GSW	41.50	0.03	41.55	0.06
CF	L2	28.24	0.56	37.9	1.39
	L1	30.23	0.35	39.13	0.99
	GLCS	37.19	0.66	36.96	0.76
	GSW	37.42	0.41	37.66	0.58
SF	L1	29.02	0.42	32.0	0.67
	L2	29.13	0.25	35.07	1.03
	LCS1	30.88	0.38	32.36	0.6
	NWM	31.01	0.33	32.28	0.83
FA	SWI	19.31	0.56	18.01	0.42
	SWM	22.32	0.38	21.31	0.56
	L2	32.54	0.31	35.53	2.1
PR	SWI	4.59	0.22	5.28	0.14
	SWM	6.46	0.30	7.45	0.38
	LCS2	10.41	0.42	10.87	0.28
	NWI	18.49	0.43	18.92	0.38
	LCS1	18.80	0.47	19.59	0.42

Looking first at the results themselves, the following picture emerges. When the matches are weak (scores are low), particularly at the CL and CF levels, none of the algorithms work very well as predictors, although the global methods, specifically GOR_match_LCS and GOR_match_SW, seem to work slightly better than the local methods. This trend is accentuated for the SF level, but is totally reversed in the FA and PR levels, where the local methods are significantly better and only LCS_2aa_idents is competitive among the global methods. The reason for the latter reversal is that at the higher match levels the global methods tend to saturate, particularly when comparing sequences of very different lengths. That is, the scores rapidly approach the length of the shorter sequences. LCS_2aa_idents is most competitive because saturation is limited. Notice also that, particularly at low match levels, the lengths of the sequences are a significant predictor. This is marked for Length_S2, the lengths of the shorter sequences.

Turning to the question of variability, it is clear from the above table that, while the variability is largely a result of differing samples due to stratification, the total variability is small compared to the results. The worst case is 5.91% (FA_L2), and the average is 1.98%. The large difference between the cross-validation and stratification error values for CL_L1, CF_L1 and CF_L2 are anomalous and arise because of the complicated rule-sets generated for these classifiers based on length. (Averaged over 10 restratifications, the average number of rules for CL_L1, CL_L2, CF_L1 and CF_L2 are respectively 26.3, 27.0, 25.4 and 20.4, versus an average for the remaining experiments of 4.06 rules.) In general, these results indicate that basing further experiments on a single stratification sample and separate training and test sets is a feasible methodology.

4. Experiment 2: Pairwise Combinations

In this set of experiments, classifiers are constructed for all pairs of the non-length similarity measures.

4.1 Results

Expt Pair			Testing %Error	Testing SD
CL	NWI	GLCS	39.73	0.09
	NWM	GLCS	39.78	0.09
	LCS1	GSW	39.91	0.09
	NWM	GSW	39.92	0.09
CF	LCS1	GSW	34.07	0.56
	NWM	GLCS	34.18	0.56
	LCS1	GLCS	34.59	0.56
	NWI	GLCS	35.17	0.56
SF	LCS1	GLCS	26.83	0.48
	LCS1	GSW	28.41	0.49
	NWM	GSW	28.46	0.49
FA	SWI	NWI	16.67	0.54
	SWI	GLCS	16.82	0.54
	SWI	LCS1	16.91	0.55
	SWI	SWM	16.99	0.55
PR	SWI	NWM	3.74	0.22
	SWI	LCS1	3.74	0.22
	SWI	NWI	3.81	0.22
	SWI	LCS2	4.08	0.23
	SWI	GSW	4.09	0.23
	SWI	SWM	4.13	0.23

From the above table it can be seen that combining similarity measures results in somewhat improved predictive accuracy. Notice that it is not necessarily the combination of the best single similarity measures that produces a better combined similarity measure. (This is most evident in the higher descriptor levels).

5. Experiment 3: Comparing Local versus Global Alignments

The results reported above suggest that global methods are preferable when only low-level alignments (e.g. same structure class) are possible, but local methods are preferable when the sequences have common functions or domains. The explanation for the relative strength of global methods at low similarity values is that, due to the action of gap penalties matching scores degrade rapidly with falling match levels, so for the alignments where only class matches are possible, alignments may be reduced to a number of islands, none of which are significant. By contrast, global methods have the entire sequence in their purview, so are more likely to find a match if one exists. At the other extreme, as suggested earlier the global methods tend to saturate when faced with a strong alignment.

Further evidence for the differing nature of local and global alignment strategies can be obtained by taking the pairwise correlations between the the eight non-length similarity measures.

Metric1	Metric2	r^2
NW_idents	LCS_idents	0.9967
NW_matches	LCS_idents	0.9827
GOR_match_SW	GOR_match_LCS	0.9783
NW_idents	NW_matches	0.9748
NW_matches	GOR_match_LCS	0.9284
LCS_idents	GOR_match_LCS	0.8955
NW_matches	GOR_match_SW	0.8878
NW_idents	GOR_match_LCS	0.8842
LCS_idents	GOR_match_SW	0.8507
NW_idents	GOR_match_SW	0.8393
NW_idents	LCS_2aa_idents	0.7405
LCS_idents	LCS_2aa_idents	0.7282
NW_matches	LCS_2aa_idents	0.6583
SW_idents	LCS_2aa_idents	0.6188
SW_idents	SW_matches	0.6165
LCS_2aa_idents	GOR_match_LCS	0.5434
GOR_match_SW	LCS_2aa_idents	0.523
SW_matches	LCS_2aa_idents	0.3767
SW_matches	NW_idents	0.34
SW_matches	LCS_idents	0.3364
SW_matches	NW_matches	0.3206
SW_matches	GOR_match_LCS	0.2911
SW_matches	GOR_match_SW	0.2854
SW_idents	NW_idents	0.2436
SW_idents	LCS_idents	0.2323
SW_idents	NW_matches	0.1882
SW_idents	GOR_match_SW	0.1436
SW_idents	GOR_match_LCS	0.1408

While most of the correlations conform to one's expectations, there are some surprises.

While NW_idents, NW_matches and LCS_idents are extremely well correlated the same cannot be said for SW_idents versus SW_matches, which are only moderately well correlated. In addition, LCS_2aa_idents is only somewhat better correlated with LCS_idents than it is with either NW_matches or SW_idents.

6. Conclusions

The principal conclusion is that, in the context of database searches, global methods and search criteria based on common domains may be useful for detecting matches at the point where local search methods are failing to find anything. On the other hand, when a match is strong - homology is strongly indicated - local methods not only have better predictive accuracy but also provide more biologically meaningful alignments than global alignments. However, even then, combining a local method with a global method may improve sieving for possible matches.

This suggests the following may be a useful procedure: if a Smith-Waterman match cannot be found, global methods may be used to add weight to the candidate matches found by the Smith-Waterman scan. Alternatively, the list of matches found by a global-alignment scan against the SCOP sequences can be searched for commonalities of class, common fold, etc. Secondary structure tokenisation of sequences (whether used in conjunction with LCS or SW) seems to be the most effective of the global methods under investigation, with LCS based on single amino acids also being competitive. Interestingly, while LCS based on overlapping dipeptides is a global method, it nonetheless seems to be competitive with the local methods at the FA and PR descriptor levels, while also being competitive with the other global methods at the lower descriptor levels. In short, LCS based on overlapping dipeptides warrants further investigation.

While it is interesting that a number of the findings from [3] have been confirmed in this study across both local and global alignment algorithms, a major finding of [3] is that statistically-based similarity measures significantly outperform either percent-match or raw scores. The statistically-based scores are founded on the extreme-value distribution [1]. Unfortunately, this distribution is not appropriate for global alignments so further work needs to be done to discover an appropriate background statistical distribution which can then be used to provide a basis for decisions about the statistical significance of global alignment match scores. Overall, the conclusion must be that global methods deserve further investigation.

7. Acknowledgements

I would like to acknowledge the generous support for my Fellowship at Pembroke College, Cambridge, provided by Bristol-Myers Squibb.

8. Bibliography

[1] SF Altschul and W Gish, ‘‘Local Alignment Statistics’’, *Computer Methods for Macromolecular Sequence Analysis*, ed. Russell F. Doolittle, pp. 460-480, Academic Press (1996) (Methods in Enzymology 266).

[2] P Bork and EV Koonin, ‘‘Predicting Functions from Protein Sequences - Where are the Bottlenecks?’’, *Nature Genetics* **18**, pp. 313-318 (1998).

[3] SE Brenner, C Chothia and TJP Hubbard, ‘‘Assessing Sequence Comparison Methods with Reliable Structurally Identified Distant Evolutionary Relationships’’, *Proceedings of the National Academy of Sciences (USA)* **95**, pp. 6073-6078 (1998).

[4] WW Cohen, ‘‘Fast Effective Rule Induction’’, *Twelfth International Conference on Machine Learning*, Lake Tahoe, U.S.A, pp. 115-123, Morgan Kaufmann (July 9-12, 1995).

[5] TH Cormen, CE Leiserson, RL Rivest and C Stein, *Introduction to Algorithms (2e)*, MIT Press (2001).

[6] J Garnier, J.-F Gibrat and B Robson, ‘‘GOR Method for Predicting Protein Secondary Structure from Amino Acid Sequence’’, *Computer Methods for Macromolecular Sequence Analysis*, ed. Russell F. Doolittle, pp. 540-553, Academic Press (1996) (Methods in Enzymology 266).

[7] U Hobohm, M Scharf, R Schneider and C Sander, ‘‘Selection of Representative Protein Data Sets’’, *Protein Science* **1**, pp. 409-417 (1992).

[8] TJP Hubbard, B Ailey, SE Brenner, AG Murzin and C Chothia, ‘‘SCOP: a Structural Classification of Proteins Database’’, *Nucleic Acids Research* **27**(1), pp. 254-256 (January, 1999).

[9] SB Needleman and CD Wunsch, ‘‘A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins’’, *Journal of Molecular Biology* **48**, p. 443−453 (1970).

[10] C Rick, ‘‘A New Flexible Algorithm for the Longest Common Subsequence Problem’’, *Nordic Journal of Computing* **2**(4), pp. 444-461 (1995).

[11] TF Smith and MS Waterman, ‘‘Identification of Common Molecular Subsequences’’, *Journal of Molecular Biology* **147**, p. 195−197 (1981).