An Introduction to Sequence Similarity ("Homology") Searching

Gary D. Stormo[1]

[1]Washington University, School of Medicine, St. Louis, Missouri

ABSTRACT

Homologous sequences usually have the same, or very similar, functions, so new sequences can be reliably assigned functions if homologous sequences with known functions can be identified. Homology is inferred based on sequence similarity, and many methods have been developed to identify sequences that have statistically significant similarity. This unit provides an overview of some of the basic issues in identifying similarity among sequences and points out other units in this chapter that describe specific programs that are useful for this task. *Curr. Protoc. Bioinform.* 27:3.1.1-3.1.7. © 2009 by John Wiley & Sons, Inc.

Keywords: sequence similarity • homology • dynamic programming • similarity-scoring matrices • sequence alignment • multiple alignment • sequence evolution

AN INTRODUCTION TO IDENTIFYING HOMOLOGOUS SEQUENCES

Upon determining a new sequence, one common objective is to assign a function to that sequence, or perhaps multiple functions to various subsequences. Because homologous sequences, those related by common ancestry, generally have the same or very similar functions, their identification can be used to infer the function of the new sequence. The methods for identifying homologous sequences rely on finding similarities between sequences that are unlikely to happen by chance, thereby allowing the inference that the sequences have evolved from a common ancestor. The units in this Chapter all address aspects of this problem, finding significant similarity between new sequences and those that are currently in the databases of known sequences. This chapter introduction provides a primer on five fundamental issues related to similarity searches: (1) how to find the best alignment between two sequences; (2) methods for scoring an alignment; (3) speeding up database searches; (4) determining statistical significance; and (5) using information in multiple alignments to improve sensitivity.

OPTIMAL SEQUENCE ALIGNMENTS

Given any two sequences, there are an enormous number of ways they can be aligned if one allows for insertions and deletions (collectively referred to as indels or gaps) in the alignment. In order to assess whether the similarity score between two sequences is significant, one must obtain the highest possible score; otherwise one will underestimate the significance. Fortunately, there is a simple method, known as dynamic programming (DP), to determine the score of the best possible alignment between two sequences. Figure 3.1.1 outlines the DP algorithm. The two sequences, A and B, are written across the top and down the side of the DP matrix. Each element of the matrix, S(i,j), represents the score of the best alignment from the beginning of each sequence up to residues a_i, in sequence A, and b_j, in sequence B. That element is determined by comparing just three numbers, which represent the three possible end points of those alignments: a_i is aligned to b_j; a_i is aligned to a gap in sequence B; b_j is aligned to a gap in sequence A. The three numbers to be compared include the addition of the gap score, δ (which is always

Current Protocols in Bioinformatics 3.1.1-3.1.7, September 2009
Published online September 2009 in Wiley Interscience (www.interscience.wiley.com).
DOI: 10.1002/0471250953.bi0301s27

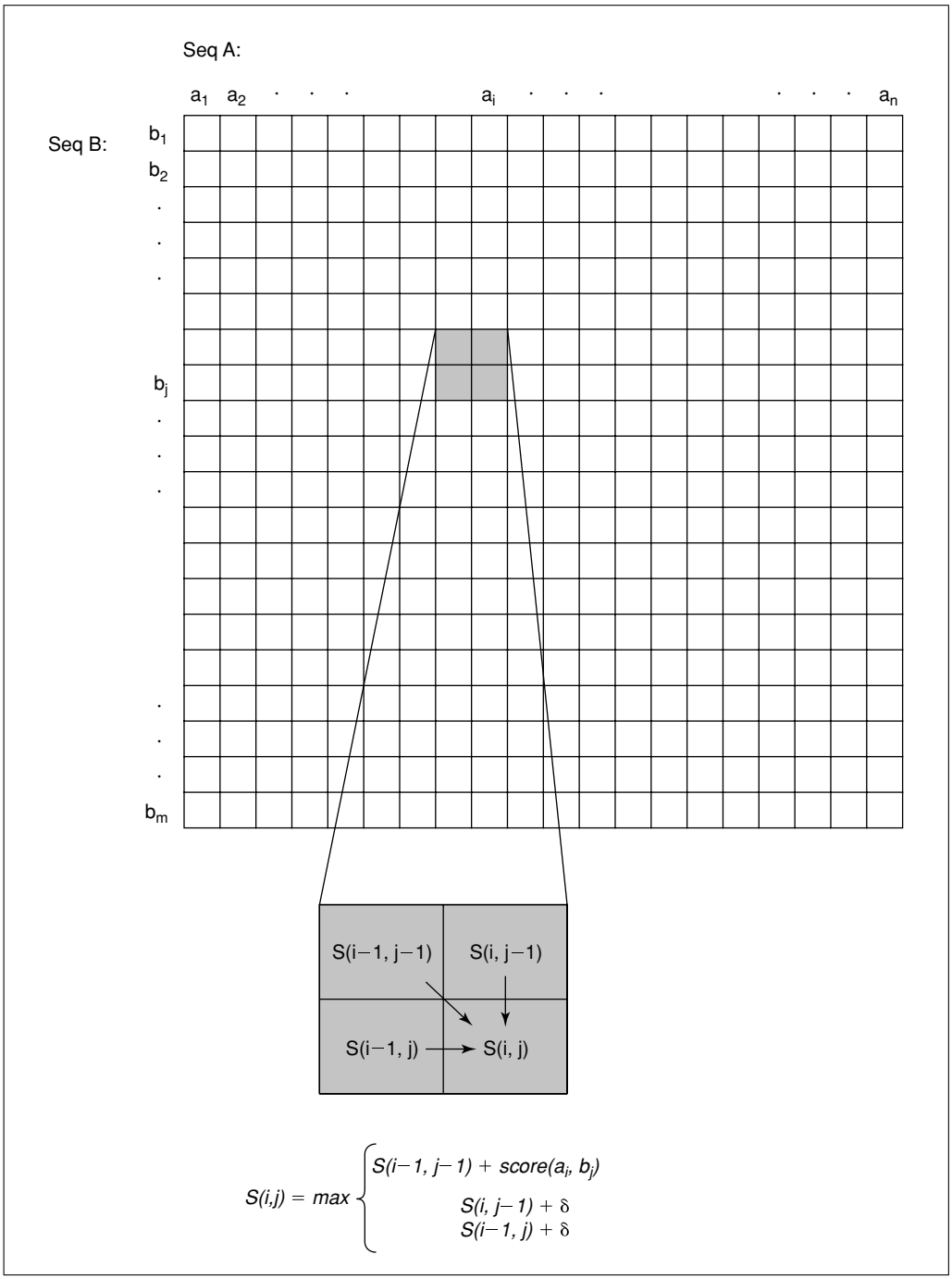

Figure 3.1.1 Dynamic programming algorithm for optimum sequence alignment. The two sequences are written across the top and along the right side of the matrix. The score of each element is determined by the simple rules shown for the enlarged section and described by the equations below it. (The top row and left column have special rules as described in the text.) The score of the best global alignment is the element S(n,m) and the alignment with that score can be obtained by backtracking through the matrix, determining the path that generated the score at each element.

negative because gaps are always penalized), to the elements directly above, S(i,j-1), and to the left, S(i-1,j), corresponding to the gaps in sequence B and A, respectively. The other number is the sum of the score of aligning a_i with b_j, which depends on what those residues are (see next section) and the score of the best alignment for the preceding subsequences, S(i-1,j-1). The top row and first column require special treatment, but the rule is still simple. Aligning a_1 with b_1 just gets *score*(a_1,b_1), but aligning a_1 with b_j

requires j-1 gaps in sequence B, so $S(1,j)=(j-1)\delta + score(a_1,b_j)$. Once the top row and first column are initialized, the entire matrix can be filled following the simple rule shown in Figure 3.1.1, and the score of the best possible alignment of the two sequences is the element in the lower right corner of the matrix, S(n,m). The actual alignment that gives that score can be found by tracing backwards through the matrix, starting from the final element, to identify the preceding element that gave rise to each score. Keep in mind that there may be more than one alignment with the highest possible score, but most programs only show one of them.

The result of the DP algorithm described above is a "global alignment" of the two sequences, where each residue of each sequence is included in the alignment and contributes to the score. However, often one is interested in the best "local alignment," an alignment that may contain only a portion of one, or both, sequences. For example, two proteins may have a common domain that one would like to identify but the remaining parts of the sequences are unrelated. Alternatively, perhaps one is comparing a newly sequenced gene to the entire genome of another species and you only expect it to match the segment corresponding to the homologous gene. A simple modification of the basic DP algorithm can be used to identify the best local alignment, as first deduced by Smith and Waterman (1981). The simple change is that only positive values are kept in the DP matrix, which just means that zero is one of the values to be compared and the maximum taken for S(i,j). Now the score of the best alignment is not necessarily the element S(n,m), but may occur anywhere within the matrix. Once the highest score is found, the alignment that generated that score can be found by tracing backwards through the matrix, as before, but stopping when the first zero is encountered.

One additional modification to the basic DP algorithm is often used. It is known from extensive studies of protein sequences that gaps of multiple residues are much more frequent than the same number of separate gaps. For example, a single deletion event might remove three residues at once, and that event is much more likely than for three separate events that each delete one residue. To capture this effect, one can use two gap penalties, one that counts the occurrence of a gap, usually referred to as the gap-opening penalty, and a different, lower one, that is proportional to the length of the gap. This can be easily incorporated into the DP algorithm with only a minor increase in the time required to find the best alignment (Fitch and Smith, 1983). The DP algorithm is guaranteed to give the highest-scoring alignment of the two sequences if the only evolutionary processes allowed are substitutions and indels. While real evolutionary processes also include duplications, transpositions, and inversions, those are not considered, as they cannot be handled by the DP approach and would drastically increase the time required to find the optimal alignment. However, it is generally seen that substitutions and indels are much more common than the other processes, so finding the best aligned considering only those processes will most frequently indicate the true optimum alignment.

SCORING SEQUENCE SIMILARITY

UNIT 3.5 discusses the construction of protein similarity matrices and considerations to take into account when choosing which to use for a specific search. This section briefly describes some of fundamental issues regarding similarity scores. The score of an alignment is the sum of the scores for each position in the alignment, as we saw in the DP algorithm for finding the highest-scoring alignment. For every pair of residues, which might be RNA, DNA, or protein, there is a score defined such that the more similar the sequences are, the higher the score. For RNA and DNA, the scores often just distinguish between matches, where the two residues are the same, and mismatches. However, sometimes it is useful to distinguish between mismatches that are transitions, changes from one purine or pyrimidine to the other, from those that are transversions, changes between a purine and a pyrimidine. For proteins, the situation is more complex and one needs a

score for all possible amino acid pairs and, ideally, one that reflects the probability of one amino acid being replaced by another during the course of evolution. Replacements are governed by two processes. The first is the mutation that changes one amino acid to another, and this will be much more likely for amino acids that have similar codons. For example, a single transition mutation can change a His to a Tyr (CAT to TAT), but three transversions would be necessary to change a His to a Met (CAT to ATG). The second process is that the mutation must survive selection for it to be observed, and this requires that the new amino acid can substitute for the previous one without disrupting the protein's function. This is more likely if the two amino acids have similar properties, such as size, charge, and hydrophobicity. Which properties are most important may depend on where the amino acid is within the protein sequence and what role it plays in determining the structure and function of the protein. Some positions in a protein may be highly constrained and only a few, perhaps only one, amino acids will work, whereas other positions may tolerate many different amino acids and therefore be highly variable in related protein sequences. This implies that one would really like to have different scoring matrices depending on the constraints at particular positions in the protein, but this would require many more parameters, and often one is lacking the information needed to choose the appropriate matrix. This problem can be partially addressed using multiple alignments of protein families, as described below. However, for general database searches, where one hopes to identify homologous sequences in the collection of known sequences, a general similarity-scoring matrix, which is the average over various types of constrained positions, is often utilized.

Most commonly used similarity matrices are empirically based on reliable alignments of protein families. For instance, the BLOSUM matrices (*UNIT 3.5*) are based on the alignments in the BLOCKS protein family alignments (*UNIT 2.2*). These are reliable alignments of protein domains across many different protein families. The score assigned to a specific pair of residues is the log-odds of their co-occurrence in aligned positions compared to what would be expected by chance given their individual frequencies. So for two amino acids, *a* and *b*, the similarity score is:

$$\text{score}\,(a,b) = \ln \frac{f(a,b)}{f(a)\,f(b)}$$

where *f(a,b)* is the observed frequency at which those amino acids are aligned and *f(a)* and *f(b)* are their frequencies in the entire database. If the two amino acids occur together at a frequency expected by chance, which is the product of the independent frequencies, then the score is 0. Amino acids that occur together more often than expected by chance get positive scores and ones that occur less frequently get negative scores. The score for aligning an amino acid with itself is always the highest score for that amino acid, as it is more likely to occur than any substitution. Positive-scoring pairs represent amino acids with similar properties, and they are often observed to substitute for one another, whereas negative scoring pairs are dissimilar amino acids that are less likely to substitute for each other. It is important to note that the expected score of a random alignment is negative. In addition to the amino acid similarity scores, one also needs to have penalties for indels in the alignment, and those values are also based empirically on many reliable alignments over many different protein families.

FAST SEARCHING METHODS

The DP method described above is guaranteed to find the highest-scoring alignment between two sequences. The time it takes to complete the alignment is proportional to the number of DP matrix elements to compute, which is the product of the sequence lengths. While this is very fast for comparing any two sequences of reasonable length, it is

not practical for searching the current sequence databases, which contain many millions of sequences and many billions of residues. Therefore methods have been developed that can search entire databases much faster. While these methods do not guarantee finding the absolute best alignments, they have been finely optimized so that they have very high sensitivities and generally do obtain the optimal, or near-optimal, alignments. The most commonly used method is BLAST (Altschul et al., 1990; *UNITS 3.3, 3.4 & 3.11*). BLAST is convenient to use through the online service (*UNITS 3.3 & 3.4*), has access to the major DNA and protein databases, and provides convenient tools for displaying and analyzing the output. One can also download BLAST to a local computer for in-house use (*UNIT 3.11*). But most importantly BLAST has been shown to be very sensitive, giving results nearly equivalent to running the Smith/Waterman DP algorithm on the whole database but in a small fraction of the time.

As a brief and approximate description of how BLAST works, consider the DP matrix of Figure 3.1.1. Most of the time is spent determining the scores of elements that do not contribute to the final, optimum alignment. BLAST achieves its speed by eliminating most of the computations and focusing its effort on the highest-scoring paths through the matrix. It does this by creating an index of the "words," short subsequences, from the query sequence, and high-scoring matches to those words can be identified very quickly in the database. Those initial matches, or "hits," are extended using DP but with cutoffs employed so that when any extension becomes unlikely to lead to a significant match it is terminated. The matches that do lead to significant alignments are extended as far as possible to give a result essentially equivalent to that obtained by the full DP algorithm. Since most sequences in the database, and most of the possible alignments even to the most similar sequences, are eliminated quickly from further consideration, BLAST achieves an enormous increase in speed with only a small loss of sensitivity.

THE SIGNIFICANCE OF AN ALIGNMENT SCORE

Obtaining a score for an alignment does not, by itself, tell you whether it is significant. One needs to determine what is the probability of observing such a score by chance, given the scoring system used and the lengths of the sequences being compared. It is important to realize that the distribution of optimum alignment scores is not normal. If one were to consider the scores of all possible alignments, one would expect a normal distribution. However, we are only considering the highest-scoring alignments to each sequence in the database, and the distribution of those maximum, or extreme, values will be quite different from the distribution of scores for all alignments. Consider the following simple example. If one throws two dice, the score distribution follows the binomial distribution, and the probability of getting a 12 is 1/36. But if one throws two pair of dice, and records only the highest score, the distribution is quite different, with the probability of getting a 12 nearly twice as high, $71/36^2$ to be precise. When searching a large database with a query sequence, one is interested only in the highest-scoring alignments, those that are mostly likely to represent homologous sequences. However, to answer the question of how high a score is required to be unlikely to have occurred by chance, one must know the distribution of highest scores expected by chance. Karlin and Altschul (1990) showed that for alignments without gaps, the distribution of highest scores follows a Gumbel distribution, a type of extreme value distribution. Furthermore, they showed how to compute the dependency of that score distribution on the scoring system that is used and the length of the query and database sequences. In addition, while their theory only holds analytically for ungapped alignments, empirical evidence indicates that it holds quite well for gapped alignments too. This allows one to compute a statistical significance for the best scoring alignments in a database search and from that infer whether the best matches are likely to be homologous sequences or could have arisen by chance. The BLAST program computes these values as part of the output, with

most people focusing on the E-value, which is the number matches with equal or greater scores that are expected by chance. When E-values are much less than one, they are essentially the same as p-values, the probability of an equal or greater score by chance, but E-values can exceed one, whereas p-values cannot.

MAKING AND USING MULTIPLE SEQUENCE ALIGNMENTS

The previous sections of this unit have focused on aligning pairs of sequences, or more generally searching an entire database for high-scoring alignments to a query sequence, which requires many pairwise comparisons. However, if one has several members of a protein family, an alignment of those sequences can provide additional information about the sequence constraints at different positions of the proteins. For example, some positions may be critical for the function of those proteins and perhaps only one, or a few, amino acids are allowed at those positions. If that is the case, then a general similarity-scoring matrix is not appropriate and one should utilize the information available from the multiple alignments to constrain the search for homologous proteins. The two main issues are how the multiple alignments are made and how they are used to identify new members of the protein family. Several units in this chapter and the previous one address various aspects of the construction and use of multiple sequence alignments. *UNIT 3.7* provides an overview of multiple sequence alignment.

One can generate optimal multiple alignments by extending the DP method to multiple sequences, following the same rules as described above but expanded to multiple dimensions, one for each sequence. However, the time required to fill in the DP matrix is proportional to the product of the length of all the sequences, which becomes impractical for more than a few sequences (Lipman et al., 1989). Practical methods for multiple sequence alignment use one of two approaches. One is a progressive alignment strategy where pairs of sequences are aligned and then new sequences are added to that alignment, or alignments are aligned to each other, but all steps only involve pairwise comparisons that can be accomplished efficiently using the DP strategy described. The Pileup program from GCG (*UNIT 3.6*) uses that strategy, as does the ClustalW program (*UNIT 2.3*), which is probably the most popular method for constructing multiple alignments. Psi-BLAST also uses a similar iterative strategy within the BLAST suite of programs (*UNIT 3.4*) to build multiple alignments from the matches identified by BLAST. The other strategy uses an interative refinement approach, where an initial approximate alignment is built and then sequences are realigned to the resulting model until the method converges to a final multiple alignment. The method is based on an expectation maximization (EM) algorithm, similar to that used in MEME (*UNIT 2.4*) but allowing gaps in the alignment. The EM algorithm is at the heart of the Hidden Markov Model (HMM) approach to protein family models first developed by the Haussler group (Krogh et al., 1994; Eddy, 1996; overview in *UNIT 3.7 & APPENDIX 3A*). In an HMM each position of the alignment is represented by a probability distribution over all of the amino acids and there are probabilities assigned for insertions and deletions. This model captures, in a probabilistic fashion, the natural variability and constraints that are observed in a protein family. The T-Coffee approach (*UNIT 3.8*) is somewhat different in that it combines, through a consistency-based scoring system, pairwise alignments that can be obtained from a variety of different and independent approaches. For example, if structural information is available and useful for determining the correct alignment that can be incorporated along with other types of information.

Once a multiple alignment is built, it can be used to identify new sequences that are members of the family. Gribskov et al. (1987) first described such an approach where the multiple alignment is converted to a "profile," and new sequences are scored against it using a DP algorithm and a sum-of-pairs method. This is equivalent to finding the best

alignment of the new sequence to the multiple sequence alignment, where the scoring is the average similarity score to each sequence in the alignment. This is also how ClustalW works internally, as it is building up the multiple alignment (*UNIT 2.3*). A new sequence can also be aligned to an HMM for a protein family using a version of the DP algorithm, the same method that is used in the iterative realignment for building the HMM. Pfam (*UNIT 2.5*) is a database of protein families represented as HMMs, and a new sequence can be compared to the database to determine how similar it is to each family and if it can be inferred to belong to any of the families. The InterPro resource of databases and analysis tools (*UNIT 2.7*) also facilitates the identification of protein families that may be homologous to a new sequence of interest.

SUMMARY

When one obtains a new sequence and wants to assign a function to it, identifying homologous sequences with known function is a reliable strategy. Two complementary approaches can be easily employed and are generally very useful. The first is to search the databases of known sequences, using programs like BLAST, to determine if there are any that are sufficiently similar to be homologous, and if those sequences have known functions such that information can be transferred with reasonable reliability. One can also search the databases of known protein families that include the extra information from multiple sequence alignments, such as Pfam and the InterPro databases. While those protein family databases are less comprehensive than the single sequence databases, simply because many sequences do not have known family membership, they can be more sensitive at identifying distantly related sequences that may be missed without knowledge of the specific constraints associated with specific families. Therefore, a combined approach that utilizes both types of resources can be most effective in identifying homologous sequences for a new sequence of interest.

LITERATURE CITED

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.

Eddy, S.R. 1996. Hidden Markov models. *Curr. Opin. Struct. Biol.* 6:361-365.

Fitch, W. and Smith, T. 1983. Optimal sequence alignments. *Proc. Natl. Acad. Sci. U.S.A.* 80:1382-1386.

Gribskov, M., McLachlan, A.D., and Eisenberg, D. 1987. Profile analysis: Detection of distantly related proteins. *Proc. Natl. Acad. Sci. U.S.A.* 84:4355-4358.

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. U.S.A.* 87:2264-2268.

Krogh, A., Brown, M., Mian, I.S., Sjölander, K., and Haussler, D. 1994. Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.* 235:1501-1531.

Lipman, D.J., Altschul, S.F., and Kececioglu, J.D. 1989. A tool for multiple sequence alignment. *Proc. Natl. Acad. Sci. U.S.A.* 86:4412-4415.

Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *J. Mol. Biol.* 147:195-197.