Overview

Strategies for Searching Sequence Databases

BioTechniques 28:1174-1191 (June 2000)

Hugh B. Nicholas Jr., David W. Deerfield II and Alexander J. Ropelewski
Pittsburgh Supercomputing Center, Pittsburgh, PA, USA

ABSTRACT

We provide a detailed overview of the choices inherent in performing a sequence database search, including the choice of algorithm, substitution matrix and gap model. Each of these choices has implications that can be described as restrictions on the underlying model of sequence evolution, the expected degree of divergence between the query sequence and the database sequences (if one uses an evolutionary based matrix), as well as the sensitivity and selectivity of the search. We conclude with a series of recommendations for researchers performing these searches based on our experience and literature studies.

INTRODUCTION

Database searching is the application of knowledge gained from previous examinations of well-characterized sequences to the problem of discovering the biochemistry and physiology of a newly discovered gene or its protein product. The objective of a database search is to distinguish sequences related to the query sequence by some model (e.g., evolution) from unrelated sequences. This objective is different from the objective of aligning sequences, which is to discover the most likely history of changes between sequences already inferred to be related. This difference in objectives implies a very different set of choices for parameters and strategies. A database search is a computational investigation and, like a laboratory investigation, must be performed thoughtfully. A database search performed with injudicious parameters will lead to wrong answers and missed discoveries. While the default parameters on most database search servers are appropriate to a wide range of the most common circumstances, no single set of parameters will be the best for all searches. Thus there are at least three distinct situations where other parameters are likely to give better results.

The first is when sequences in the database that are homologous to the query sequence are evolutionarily highly diverged. The second is when either the query sequence or its homologues in the database are too short to achieve a statistically significant match using the default parameters. The third is when the researcher is interested in finding homologues in only a limited range of species whose evolutionary divergence from the species in which the query sequence was found is different from that implied by the default parameters. Examples of all of these situations are reported in the literature cited in this paper. This discussion is intended to help researchers recognize and adjust their database searches to these circumstances.

Database searching helps to evaluate whether a newly determined sequence is related to a previously determined and characterized gene or protein through a common evolutionary ancestor. Homology, relatedness through a common evolutionary ancestor, is not directly observable. In sequence database searching, we observe sequence likeness or similarity. If the likeness is great enough, we may infer that the two sequences are homologous. Thus, much of the previously determined knowledge we apply in database searching involves how to best measure sequence likeness and how to assess whether the observed degree of sequence likeness is sufficient to allow us to infer that the sequences are homologous; that is, related through divergent evolution.

The similarity scores for pairs of sequence residues, either amino acids or nucleotides, used to assess sequence likeness are the major sources of previous biological knowledge for database searching. Similarity matrices provide a quantifiable measure of the ability of one residue to replace another, instead of assuming that all residues are equally conserved and that all mismatches are equally bad, as does the unitary scoring matrix. Similarity matrices have been developed reflecting different degrees of evolutionary divergence. The point accepted mutation (PAM) (11) and block sum (BLOSUM) (14) similarity matrices widely used in protein database searches are two good examples. These matrices incorporate the information of which residues have successfully replaced each other during the course of evolution. Similar matrices are available, if not widely used, for nucleic acid sequences (30). The nucleic acid matrices can incorporate knowledge about differential rates of transitions and transversions in the same way that some amino acid substitutions are judged more favorable than others in protein similarity matrices.

The algorithm used in searching the database incorporates the knowledge of how to find the maximum degree of sequence likeness and is the second most important source of previous knowledge. The three most widely used programs today, Smith-Waterman (28), FASTA (25), and BLAST (3,4),

place different restrictions on the simple evolutionary model on which database searching is based. Smith-Waterman is the most rigorous algorithm and does not place any heuristic restrictions on the evolutionary model; it is both the most sensitive and the least selective algorithm. The actual pattern of evolutionary changes between a newly determined sequence and any homologue in the database can be incompatible with the heuristic restrictions imposed by either BLAST or FASTA. Alternatively, the additional selectivity that results from these restrictions can sometimes be an advantage. There is no single program that is always best at finding distantly related sequences for all gene and protein families, although the Smith-Waterman algorithm is most often the best (23,24).

Virtually all sequence database search programs now return an estimate of the statistical significance of the matches observed between the query sequence and sequences in the database. The statistical significance is an attempt to determine how often a score of a given value would be observed from the comparison of a random sequence to the length and composition of the query sequence and the sequences in the database.

Finally, the sequence database itself represents a large store of previously acquired knowledge. Making the best use of this knowledge can save many months of expensive laboratory experimentation and allow limited resources to be used to acquire truly new knowledge. The size of this potential gain is the determining factor in deciding how much effort to devote to any particular database search.

SEARCH ALGORITHM

In the next section, we will describe the operation of the three most widely used database searching algorithms: Dynamic Programming Algorithm [Smith-Waterman (28,33), Needleman-Wunsch (21) and Sellers (26,27)], Word Search [FASTA (25)] and Maximal Segments Pairs [BLAST (3)].

Dynamic Programming Algorithm

The dynamic programming algorithm is mathematically rigorous and, given a specific substitution matrix and gap model, is guaranteed to find the optimum score and alignment (21,26,28,33). Several variants of the dynamic programming algorithm are useful in different situations. The first use of the dynamic programming algorithm for sequence analysis was by Needleman and Wunsch (21) and independently by Sellers (26). These equivalent variants of dynamic programming provide a global alignment between two sequences, which forces the alignment to include the entire length of both sequences. The Smith-Waterman algorithm (28,33) yields a local alignment, which provides the regions within the pair of sequences that are the most similar given the choice of scoring matrix and gap penalties. A third variant of the dynamic programming algorithm first proposed by Sellers (27), often called the quasi-global alignment algorithm, in a well-behaved case, aligns an entire shorter sequence within the best matching region of a longer sequence. It has been shown that the Smith-Waterman (local alignment) algorithm is the most effective of the database searching dynamic programming algorithms for finding similar sequences, although the Needleman-Wunsch (global alignment) or Sellers (quasi-global) may be more appropriate for problem-specific final alignment.

The Smith-Waterman algorithm is a recursive mathematical equation (28,33). Recursive means that the results are computed in steps with any subsequent step depending on the answers to previous steps. The Smith-Waterman algorithm can be expressed as:

$$SW_{i,j} = \max\{SW_{i-1,j-1} + s(a_i,b_j);\ SW_{i-k,j} + gap_j;\ SW_{i,j-k} + gap_i;\ 0\}$$

where, for an affine gap penalty:

$$gap_j = open\_gap + (i-k+1)*extend\_gap$$

$$gap_i = open\_gap + (j-k+1)*extend\_gap \qquad \text{[Eq. 1]}$$

$SW_{i,j}$ is the Smith-Waterman score for the partial alignment ending at residue i of sequence a and residue j of sequence b. It can be visualized in a table form (Figure 1). The value of $SW_{i,j}$ is the maximum of four different terms. The first term, $SW_{i-1,j-1} + s(a_i,b_j)$, corresponds to extending the alignment by one residue from each sequence in which $s(a_i,b_j)$ is the similarity score for aligning the i residue in sequence a with the j residue in sequence b. The second term, $SW_{i-k,j} + gap_j$, describes extending the alignment by including residue j from sequence b and inserting a gap of k residues in length in sequence a. The third term, $SW_{i,j-k} + gap_i$, is the equivalent term for inserting a gap into sequence b. The fourth term, 0, does not allow the computed score ($SW_{i,j}$) to become negative. Thus, the best scoring regions do not have to overcome the effects of surrounding regions of low similarity to achieve a high score. For a global alignment (Needleman-Wunsch), the fourth term is eliminated (21,26). That is, the partial scores within the table are allowed to become negative and

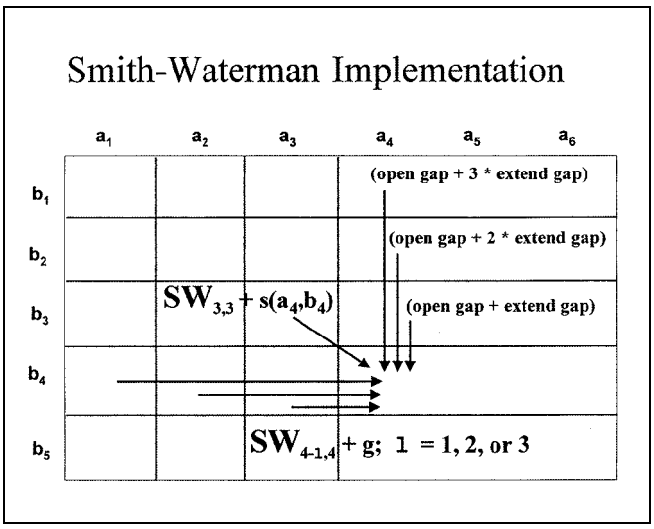

Figure 1. The two-way table layout of the calculations performed by the Smith-Waterman algorithm. Each column in the table corresponds to one residue of sequence A, a_i, and each row corresponds to one residue of sequence B, b_j. Each cell in the table, designated $SW_{i,j}$, corresponds to a partial alignment of sequence A with sequence B that ends at the sequence residues a_i and b_j. Each arrow in the table represents a term that is computed and evaluated during the calculation of table cell $SW_{4,4}$.

the boundaries of the alignment are defined by the ends of the sequences.

The alignment for a local search corresponds to the path that was used to compute the maximum score (19 in Table 1). This is accomplished in the program by starting at the maximum score and backtracking along the path until the first zero is reached. The bold and underlined numbers in Table 1 indicate this path. The MaxSegs algorithm developed by Waterman and Eggert (33) allows retrieval from the same table of additional, lower-scoring, nonintersecting alignments, which frequently provide valuable additional information about the relationship between the aligned sequences.

The dynamic programming algorithm is computationally demanding—all terms in Equation 1 for each cell in Table 1 must be computed to get the alignment score. Recovering the actual alignment requires either the entire table in memory, which for long sequences may require more memory than is available, or doubling the number of computations, which makes the required memory proportional to the length of the longest sequence (20). These computational requirements place database searches with the Smith-Waterman algorithm beyond the capabilities of personal computers.

Table 1. Smith-Waterman Alignment Table

		g	c	t	g	g	a	a	g	g	c	a	t
	0	0	0	0	0	0	0	0	0	0	0	0	0
g	0	5	0	0	5	**5**	0	0	5	5	0	0	0
c	0	0	10	3	0	1	**1**	0	0	1	10	3	0
a	0	0	3	6	0	0	6	**6**	0	0	3	15	8
g	0	5	0	0	11	5	0	2	**11**	5	0	8	11
a	0	0	1	0	4	7	10	5	**4**	7	1	5	4
g	0	5	0	0	5	9	3	6	10	**9**	3	0	1
c	0	0	10	3	0	2	5	0	3	6	**14**	7	0
a	0	0	3	6	0	0	7	10	3	0	7	**19**	12
c	0	0	5	0	2	0	0	3	6	0	5	12	15
t	0	0	0	10	3	0	0	0	0	2	0	5	17
						g	a	a	g	- g	c	a	
						g	c	a	g	a g	c	a	

Similarity Scores: DNA PAM 47, Match = 5, Mismatch = -4; Open Gap = 0, Extend Gap = -7

HEURISTIC ALGORITHMS

To reduce the computational requirements of routine database searches, a number of heuristic algorithms have been developed. The most popular heuristics are FASTA and BLAST. The heuristics can, in general, be described as a three-step process (Table 2). The first step is the initial heuristic-specific screening to identify which region (diagonal) of both the query and database sequence contains sufficient similarity for further consideration. The second step is the creation of an initial alignment in the identified region, often followed by a check to see if this alignment is statistically significant. The third step is the final alignment, usually a restricted version of the Smith-Waterman algorithm, in the region identified in Table 2, Steps 1 and 2. Substantial computational resources are saved because of the heuristic-specific reduction of the region (Table 2, Step 1) that must be investigated followed by the savings in the restricted Smith-Waterman alignment (Table 2, Step 3). The following sections will discuss the heuristic-specific limitations in FASTA and the various implementations of BLAST.

Word Search

The FASTA heuristic-specific screening is a word search. In a word (or k-tuple) search, both the query and library sequence are divided into overlapping "words" of a specific length (Figure 2). The lists of words from the query and library sequence are compared, and the diagonal with the most matching words is taken as the region most likely to contain the best alignment between the two sequences. The results from the word search are used to determine if the two sequences have a region of sufficient similarity to merit further examination, and combinations of the words are joined to create an initial alignment. If it contains sufficient sequence similarity, then a restricted Smith-Waterman alignment is

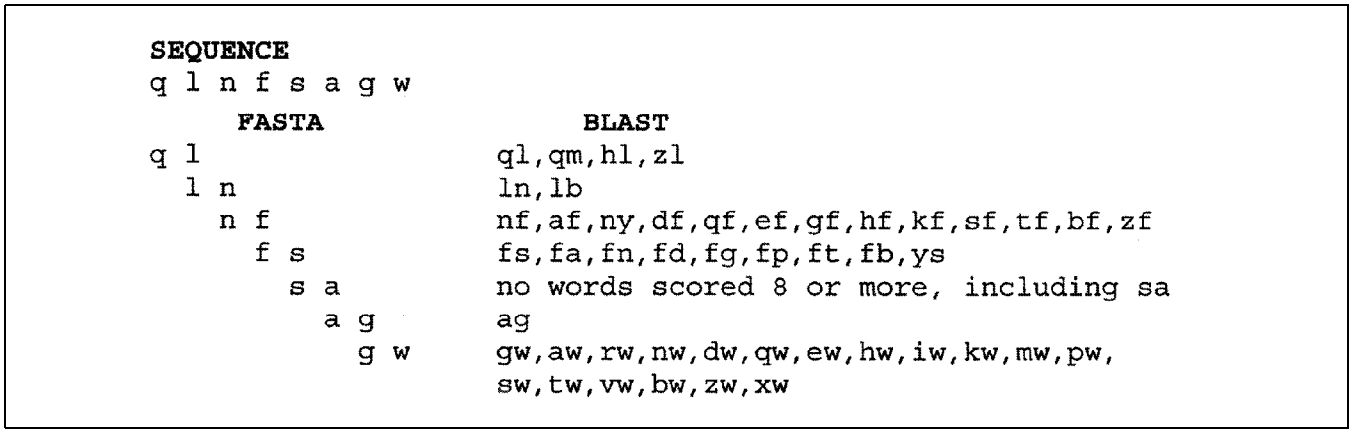

Figure 2. Word Lists as they are created for FASTA and BLAST. The word size is 2 in both examples shown here. For the BLAST example, we used the PAM120 matrix, and the threshold was arbitrarily assigned a score of 8. The sequence in this example is the adipokinetic hormone II from migratory locust.

Table 2. Schematic Representation of the Heuristic Algorithm

Step 1:Initial Word Search
- Identity for FASTA, expanded list for BLAST
- Initial filtering for both (BLAST requires two words on the same diagonal)

Step 2:Initial Alignment
- FASTA links strong diagonals
- BLAST expands good extended-word matches along the diagonal in both directions (final BLAST I alignment)
- Significance test

Step 3:Final Alignment
- FASTA and prior versions of BLAST perform a bounded (by window size) restricted Smith-Waterman
- Most recent BLAST performs a centered (on dipeptide in highest scoring 11-amino acid peptide) restricted Smith-Waterman
- Final significance test—BLAST uses the statistics underlying the MSP, while FASTA uses the database as a reference distribution.

performed on the region of the alignment table centered on the diagonal with the highest score and bounded by the window size. The results of this restricted Smith-Waterman alignment are reported as the optimal score. The computational savings can be seen for the comparison of two 300-residue proteins: a total of 90 000 cells would need to be computed for an unrestricted Smith-Waterman, while a restricted Smith-Waterman would require, at most, 9100 cells (assuming a window-size of 16 on the main diagonal).

The FASTA region-defining heuristic can be interpreted as a series of restrictions on the sequence evolution model used in comparing the sequences. The word-size parameter, usually two for proteins and six for nucleic acids, defines the first restriction. Using this restriction, FASTA constrains the evolution between a pair of sequences to preserve a number of unchanged dipeptides or hexanucleotides. The second restriction is the window size that limits the total sum of insertions or deletions one sequence can accumulate with respect to the other sequence in the course of evolution. The limitations in the FASTA approach can be shown in two pathological examples. The first example would have two proteins that share 50% identity, but the proper alignment consists of alternating match and mismatches. With a word size of two, there would be no word matches along the main diagonal of the dot plot for the sequences (although there will potentially be random or spurious word matches on the off-diagonals), and the proper alignment would probably not be found. The second case consists of two proteins that are almost identical, except the second protein has a 20-residue insertion into the middle of the sequence. If the window size is 16, then the restricted Smith-Waterman alignment will have insufficient alignment space to jump the 20-residue insertion. Thus, the resulting alignment score will either not be significant enough to be identified or the sequence before or after the insertion (whichever had the higher diagonal scores) will be reported, while the observation that the proteins were basically identical (with only one long insertion) will be missed. In practice, these pathological cases are very unlikely, but the word size restriction has been shown to affect database search results (23,24).

Maximal Segment Pairs

Maximal segment pair (MSP) alignments are defined by the first and fourth terms in the Smith-Waterman algorithm (Equation 1). Thus, the MSP alignment consists of long stretches of matches and mismatches without gaps. The advantage of this method is that the statistical significance of each found segment is easily determined. The disadvantage is that computational requirements can be comparable to Smith-Waterman, which effectively precludes this approach for routine database searching.

The BLAST algorithm contains a heuristic for reducing the computational requirements of the MSP algorithm. Like FASTA, BLAST divides the sequence into a list of overlapping words (Figure 2). BLAST extends the list to include all words that score above a specific matrix-defined threshold for the specified matrix. The value of the threshold limits the number of matches that will survive the first step (screening). The score for common, highly mutable peptides aligned with themselves may not be above the threshold required for inclusion in the list, and hence they would not be included. However, BLAST does provide the option of forcing the word into the list as long as the exact word is present in the sequence. The expansion of the word list to include all words that match with a suitably high score provides a large increase in sensitivity over using only words identically present in the sequence.

The alignment for the original BLAST algorithm was created by taking each identified matching word and then extending this match in both directions along the diagonal (without gaps) until the alignment score went below a cutoff point. The original algorithm reported the best segment if it was significant (see below). The two components of the BLAST heuristics can, as with those in FASTA, be interpreted as restrictions to the evolutionary model. The word size, usually three for proteins and 11 for nucleic acids, requires that there be at least one word-size segment within the conserved region that scores above the threshold. The second restriction is that a statistically significant ungapped segment must be conserved between the two sequences. For an average protein, the minimum length of this ungapped segment is around 35 residues.

The BLAST algorithm has undergone several refinements and improvements while attempting to maintain selectivity, increase sensitivity, decrease computational requirements and provide a better resulting alignment. The sensitivity was increased while the computational requirements were decreased by first lowering the threshold for the words to be identified and then requiring that there be at least two identified, nonoverlapping words on a diagonal before extending the alignment. Thus, more words are identified, yet the number of initial alignments examined (by extending matches along the diagonal in both directions) is decreased. The quality of the resulting alignment has been improved by first providing high-scoring

segments that were near the maximal scoring segment. Later, using an approach similar to FASTA, the maximal scoring segment is used to define a band that uses the Smith-Waterman algorithm to find a gapped alignment within the band.

The new gapped BLAST circumvents the problem of being restrained within an alignment region bounded by the window size while avoiding the high computational cost of an unrestricted Smith-Waterman alignment by extending the alignment out from a central high-scoring pair of aligned amino acids in a way analogous to how BLAST extends the initial maximal segment pair alignment. The initial pair of aligned amino acids is chosen as the middle pair of the highest scoring, 11-residue window in the high-scoring segment pair alignments. The Smith-Waterman algorithm is then used to extend the alignment in both directions until the score falls below a fixed percentage of the highest score computed in the Smith-Waterman phase. This method will find the highest-scoring Smith-Waterman alignment if two conditions are met. First, the calculation is extended until a score of zero is reached. Using a higher threshold for stopping the calculation the way BLAST does may risk not finding the complete alignment in return for a large savings in computer resources. The second criterion that must be met is that the initial pair of amino acids selected as the midpoint from which to extend the alignment must actually be part of the one that would be reported as the best by a complete Smith-Waterman alignment of the pair of sequences. The selection criteria appear to be effective in selecting an appropriate pair of amino acids from which to extend the alignment. Nonetheless, a full Smith-Waterman or Needleman-Wunsch alignment is recommended for publication.

Comparison of Search Algorithms

Both heuristics, FASTA and BLAST, approximate the Smith-Waterman algorithm and were developed to reduce the computational requirements to make routine database searches feasible. The heuristics will compute the same score for each alignment in a database search as the full Smith-Waterman if (*i*) the heuristic correctly identifies the proper region (Table 2, Step 1); (*ii*) the initial screening decision to continue analysis is correct (Table 2, Step 2); and (*iii*) the final alignment is correctly described by the restricted Smith-Waterman (Table 2, Step 3). The refinements in identifying the proper region and screening the result have established these heuristics as effective database search tools. Note that the computation of the significance for each alignment is program specific.

The two parameters used to describe the performance of a search algorithm are sensitivity and selectivity. Sensitivity is the number of related sequences found in a database search, while selectivity is the number of unrelated sequences identified in a database search. Ideally, one would want a program that is highly sensitive (recovers all related sequences) while being selective (no false positives). The Smith-Waterman algorithm often shows better sensitivity than either heuristic, although the heuristics have continued to be refined and optimized so that the Smith-Waterman's advantage is now relatively small (4,5). All algorithms demonstrate similar selectivity, with BLAST usually the most selective. The decreased sensitivity and increased selectivity of the heuristics have been attributed to the initial word search and screening (Table 2, Steps 1 and 2), which tend to eliminate more diverged sequences and false positives (24). With the advances in algorithms, effective search parameters and statistical significance models, the most common problem encountered by the algorithms is sufficient sensitivity to find all related database sequences (4,5,24). It is not uncommon for only half of the known related sequences to be identified in a trial database search (4,5,24).

For proteins, the BLAST word-based heuristic is usually more sensitive than FASTA; however, the combination of a long word size and modifications of the heuristic to improve protein sensitivity decreases the BLAST performance for nucleic acids relative to FASTA.

The failure to find all of the sequences alluded to above is the limitation that gives rise to the first of the circumstances mentioned previously where the default search parameters can be inappropriate for a specific sequence database search. A sequence database search need not find all of the homologues but only find one or more characterized homologues that provide leads for further experimentation. However, for a particular newly determined query sequence there may be only a few homologues in the sequence database, all of which are highly diverged from the query sequence. None of these few homologues may be discovered by a search using the default settings. This is because the default settings have been appropriately chosen to find the largest number of sequences for query sequences that are representative of the known members of the family for a large number of families that are present in large numbers in the sequence databases. Thus the default parameters are generally set for a moderate level of sequence divergence rather than a very high level. Much of the rest of this review addresses what to do in this circumstance.

SIMILARITY MATRICES

To use any of these search algorithms, one must quantify whether the substitution of one residue for another is likely to conserve the physical and chemical properties necessary to the structure and function of the protein or is more likely to disrupt essential structural and functional features of the protein. This choice of similarity matrix is the most critical determinant of which sequences will be reported as similar to the query sequence in the database search report. Numerous approaches have been used to create such quantifications, referred to as similarity matrices. Similarity matrices have been based on explicit or implicit (empirical) evolutionary models, structural properties such as Chou-Fasman propensities, chemical properties such as charge, polarity, and shape, as well as combinations such as those used in the structure-genetics matrix. Regardless of the underlying approach, all similarity matrices are attempts to quantify whether a mutation preserves or disrupts the function of a protein. Note that the underlying approach of the substitution matrix defines the basis for the similar sequences that will be found and reported in a database search. Thus, if you wish to infer an evolutionary relationship between similar sequences, you should use an evolution-based matrix.

Table 3. BLOSUM and PAM Similarity Matrices Equating the Matrices Based on the Entropy

BLOSUM		PAM		
Matrix	**Entropy**	**Matrix**	**Entropy**	**% Identity**
BLOSUM90	1.18	PAM100	1.18	43
BLOSUM80	0.99	PAM120	0.98	38
BLOSUM60	0.66	PAM160	0.70	30
BLOSUM52	0.52	PAM200	0.51	25
BLOSUM45	0.38	PAM250	0.36	20

The following discussion describes the approaches for several early similarity matrices, followed by a presentation of the theory and methodology for the creation of the two most commonly used evolution-based matrices.

Early Matrices

Early sequence alignment programs used the unitary scoring matrix, in which all matches were equally good and all mismatches were equally penalized. This scoring matrix was sometimes appropriate for DNA and RNA comparisons (in which transitions equal transversions and the sequences are moderately diverged), but not for protein comparisons.

Many alternatives to the unitary scoring matrix have been suggested. One of the earliest suggestions was a scoring matrix based on the minimum number of bases that must be mutated to convert a codon for one amino acid into a codon for a second amino acid. This matrix, known as the minimum mutation distance matrix, has succeeded in identifying more distant relationships among protein sequences than the unitary matrix approach. The minimal mutation distance matrix is an improvement over the unitary matrix because it incorporates knowledge about the process of mutating one amino acid into another. However, it is limited because the minimum mutation distance matrix does not include codon usage or selection processes.

Another improvement over the unitary matrix is a scoring matrix based on selected physical, chemical or structural properties shared and not shared by the 400 pairs of amino acids. Specific instances of this approach work well for some sequences but not for others. The approach works best if the matrix is based on properties that have been strongly conserved during the evolution of the sequence family. This is because the properties matrix attempts to specify the criteria that determine whether or not a mutation can survive and be fixed in a population. However, this approach suffers from problems of balancing the contributions of the different properties to the positive selection of mutations and from ignoring the different rates at which different mutations are generated.

Empirical Evolutionary Matrices and Log-Odds Scores

The biggest improvement achieved over the unitary matrix (and other theoretically based matrices) was based on the empirical study of the evolutionary replacements of amino acids. Margaret Dayhoff pioneered this approach in the 1970s when she made an extensive study of the frequencies in which amino acids substituted for each other during evolution (11). The studies involved carefully aligning all of the proteins in several families and then constructing phylogenetic trees for each family. This approach incorporates both the generation and selection of mutations and has been successful in sequence alignment applications. We present below more details of two widely used families of these empirically based matrices.

Computing Similarity Matrices

A similarity score (S_{ij}) is computed using information theory to provide a measure for the probability of residue i replacing residue j in an alignment (Equation 2).

$$S_{ij} = \log_2(R_{ij}) = \log_2(q_{ij}/p_i p_j) \qquad \text{[Eq. 2]}$$

where q_{ij} is the relative frequency with which residues i and j are observed to replace each other in related sequences.

In Dayhoff's original research, a separate q_{ij} and q_{ji} were tabulated. Dayhoff and others have since made the simplifying assumption that $q_{ij} = q_{ji}$ and averaged these values. The terms, p_i and p_j, are the expected probabilities for these residues and are usually the frequencies at which residues i and j occur in the database. The product, $p_i p_j$, is the frequency at which these residues would be expected to replace each other if the pattern of mutations were random. The similarity score (S_{ij}) is the base 2 logarithm of the ratio (R_{ij}) of the observed and expected frequencies of mutation. The base 2 logarithm allows the direct computation of entropy (see below) and a straightforward calculation of statistical significance from the score. The similarity matrices in common use have generally been scaled either by using logarithms to the base ten (the Dayhoff PAM matrices) or by multiplying by a factor of two or three (the BLOSUM matrices). This scaling makes no differences in which sequences are reported as similar by a database search (1,3) but is done to allow the similarity matrices to be represented as integer numbers with a lesser loss of accuracy. Scores above zero ($S_{ij} > 0.0$) indicate that two residues replace each other more often than would be expected if the replacement rate were random. Likewise, scores below zero indicate that residues replace each other less often than would be expected if the replacement rate were random. Thus, a positive alignment score means that the pattern of identities and substitutions described by an alignment is more likely to result from previously observed evolutionary processes than to result from random replacements.

The average amount of information available per position in an alignment computed using a specific matrix is referred to as either the information content or relative entropy, which is reported in bits. A bit is the amount of information required to distinguish between two possibilities, such as distinguishing between an alignment of homologous sequences from an alignment of random sequences. The equation for calculating entropy for a similarity matrix is shown in Equation 3 and given for selected matrices in Table 3.

$$\text{Entropy} = \sum_{i = 1, 20} \sum_{j = 1, 20} q_{ij} \cdot S_{ij} \qquad \text{[Eq. 3]}$$

Counting the Replacements

The primary difference between the two major evolution-based similarity matrices is in the computations of the replacement frequency (q_{ij}). The PAM matrices use counts derived from an explicitly tree-like, branching evolutionary model. The BLOSUM matrices use counts derived from highly conserved blocks within an alignment.

The PAM or Dayhoff Family of Matrices

Margaret Dayhoff and co-workers performed a careful global alignment (including both highly conserved and variable regions) on a number of closely related sequences, which allowed them to create the appropriate evolutionary trees (11). The ancestral sequences are computed for each of the internal nodes in the tree using the principle of minimum replacements or maximum parsimony. They then tabulated which residues were conserved and which ones were mutated for each branch in the evolutionary tree (Figure 3). Not all mutations were observed (e.g.,W→C) because of the small amount of data available. The tabulated counts were then converted into the appropriate replacement frequencies for a single point mutation. Using these underlying replacement frequencies for a single point mutation, Dayhoff and co-workers asked the question: what would the replacement frequencies be after "*n*" PAM were observed within a stretch of 100 residues? Matrix multiplication of the single point mutational frequencies results in replacement frequencies that can then be converted (Equation 2) into similarity scores representing the expected substitution pattern after a selected number of mutations. An estimated percent identity for each matrix can be computed, assuming a database-consistent distribution of residues. This approach has been applied successfully to both nucleic acids (30) and proteins (11).

Since each matrix is developed for a specific PAM distance, how well do the matrices work if the sequences that are to be identified are at a different PAM distance? That is, if two sequences are diverged by 240 PAM units, would a PAM40 matrix identify them in a database search? The efficiency of a matrix at the various PAM distances is computed as the ratio of the sequence pairwise score for the matrix in question divided by the sequence pairwise score with the properly diverged matrix (1) (Figure 4). A more complete graph can be seen at http://www.psc.edu/biomed/data/graphs.html. A database search with a different matrix is indicated (1) if the new matrix is able to increase the alignment score by 2 bits (or a significance factor of four). For proteins, this level is approximately (34-2)/(34) (where 34 bits is required for a significant database result) or 0.94 efficiency. Figure 4 shows the efficiency of four PAM matrices. Note that no matrix effectively covers the entire range of percent identity above an efficiency of 0.94. The PAM160 matrix provides the best coverage of the range of interesting sequences typically found in a database search (which is similar to the BLOSUM62). One would need two or three different matrices for an effective search of all divergence distances in a database search. We provide recommendations for matrix selection from the literature in Table 5.

The BLOSUM Family of Matrices

Henikoff and Henikoff created multiple sequence alignments of related proteins and identified conserved regions without gaps, the BLOCKS database, which serves as the source of the data for the BLOSUM matrices. These BLOCKS constitute the regions that would most likely be found in a database search. To create a specific BLOSUM matrix, in each BLOCK, all sequences that share at least *n* percent identity (where *n* = 50% for the BLOSUM50) are grouped together to create weighted set representations (Figure 5). The counts are then tabulated between the weighted set representations within each BLOCK (Figure 5). The counting procedure is the sum-of-pairs evolutionary model that allows direct evolution between any pair of sequences in the family rather than through a

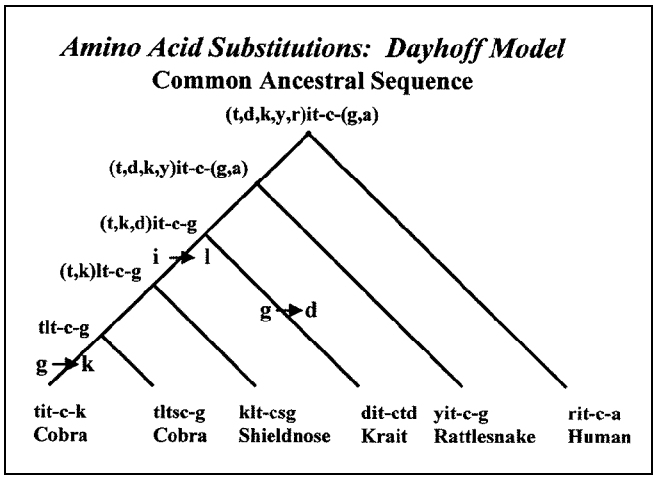

Figure 3. An evolutionary tree with the recreated ancestral sequences at the internal nodes and three amino acid substitutions indicated on branches of the tree. The observed amino acid substitutions tabulated in computing the Dayhoff PAM similarity matrices were counted by using such recreated ancestral sequences and substitutions.

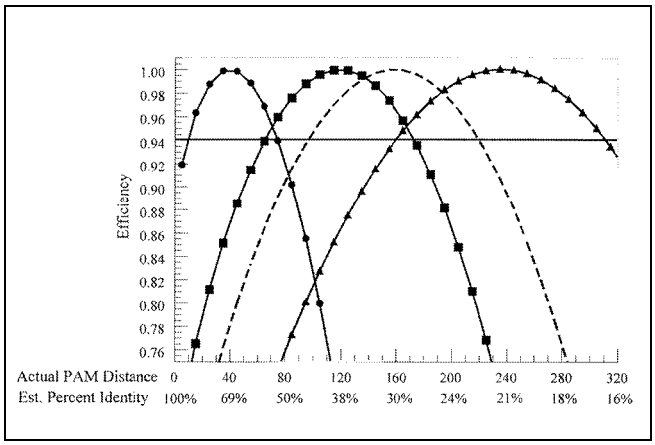

Figure 4. Efficiency plots for the protein scoring by the PAM40 (closed circles), PAM120 (closed boxes), PAM160 (dashed line) and PAM240 (closed triangles) of sequences that are at an "actual PAM distance" apart. The estimated (assuming a database-consistent distribution of residues) percent identity is also provided for reference. The solid line (efficiency = 0.94) represents the level at which using a second matrix would be beneficial.

branching process involving intermediate sequences (Figure 3). The counts are then added across all BLOCKS and used to compute the q_{ij} values (Equation 2).

The efficiency of the BLOSUM series of matrices can also be estimated but not directly computed because these matrices are not determined based on a single set of single point mutation frequencies that are then matrix-multiplied to a given set of distances. The BLOSUM62 is often the default matrix in a number of search programs. The BLOSUM62 matrix, which is similar in efficiency to the PAM160 matrix (Figure 4), has the widest span across the range of interesting sequences found in a database search. Thus, the BLOSUM62 is often the default matrix in database searching programs. But, once again, for complete coverage of all divergence distances, multiple matrices should be used.

Differences between the PAM and BLOSUM Models

The BLOSUM and PAM matrices are the most widely used amino acid similarity matrices for database searching and sequence alignment. There are three primary differences between the PAM and BLOSUM matrices: (*i*) all PAM matrices are derived using matrix multiplication from a single set of single point mutation frequencies derived from an explicit evolutionary model, while BLOSUM matrices are derived with a sum-of-pairs evolutionary model using the BLOCKS database with different weight set representations (sequence groups); (*ii*) the PAM matrices are based on mutations observed in both highly conserved and variable regions in a global alignment; while the BLOSUM matrices are based exclusively on local, highly conserved regions without gaps; and (*iii*) the PAM matrices are based on a limited number of observations (e.g., C→W was not observed), while the BLOSUM matrices are based on a large number of observations.

In empirical tests of the effectiveness of the two series of matrices, both generally perform well (11,14). However, when comparing the results of a single database search, the BLOSUM matrices will usually perform slightly better (15,17,24). This likely reflects the fact that the BLOSUM matrices are based on the replacement patterns found in more highly conserved regions of the sequences. These patterns discovered in database searches serve as anchor points in alignments involving complete sequences. It is reasonable to expect that the replacements that occur in highly conserved regions will be more restricted than those that occur in highly variable regions of the sequence.

The PAM matrices still perform relatively well despite the small amount of data underlying them. The most likely reasons for this are the care used in constructing the alignments and phylogenetic trees used in counting replacements and the fact that they are explicitly based on a simple model of evolution. They still perform better than many of the more modern matrices that have been less carefully constructed. Both the PAM and BLOSUM matrices generally perform better than matrices based explicitly on criteria other than observed replacement frequencies (17) (e.g., properties, structure-genetic and minimum mutation distance).

Multiple sequence alignment:

```
Bpi_Bovine          npGivaRItqkgLdyacqqgvltlQkele
 Bpi_Human          npGvvvRIsqkgLdyasqqgtaalQkelk
Cept_Human          eaGivcRItkpaLlvlnhetakviQtafq
 Lbp_Human          npGlvaRItdkgLqyaaqeqllalQsell
Lbp_Rabbit          npGlitRItdkgLeyaareqllalQnkll
```

Weighted set representation:

Group 1: [n][p][G][½i,½v][v][½a,½v][R][t][½s,½t][q][k][g][L][d][y][a][½s,½c][q][q][g][½t,½a][½a,½l][½a,½t][t][Q][k][e][t][½k,½e]

Group 2: [e][a][G][i][v][c][R][t][t][k][p][a][L][t][v][t][n][h][e][t][a][k][v][i][Q][t][a][t][q]

Group 3:
[n][p][G][t][½i,½v][½a,½t][R][t][t][d][k][g][L][½q,½e][y][a][a][½q,½r][e][g][t][t][a][t]
[Q][½s,½r][½e,½k][t][t]

Replacement Counts:

```
v -> t = 0.0 (within the same group)
v -> a = 0.5 * 1.0          = 0.5
v -> l = 0.5 * (0.5 + 0.5) = 0.5
t -> a = 0.5 * 1.0          = 0.5
t -> l = 0.5 * (0.5 + 0.5) = 0.5
a -> l = 1.0 * (0.5 + 0.5) = 1.0
l -> l = 0.0 (within the same group)
```

Figure 5. Counting replacements for a BLOSUM70 similarity matrix. Three different groups were defined: group 1 = Bpi (bovine and human), group 2 = Cept (human) and group 3 = Lbp (human and rabbit). Replacements within a group were not counted.

Scores for Nucleic Acids

For coding regions of genes, it is generally better to use the translated amino acid sequence for a database search (2). Many factors can interfere with nucleic acid-based searches. One of these is the compositional bias found in many organisms and organelles. Another is that some DNA sequences are derived from mRNA while others are genomic DNA with introns, and the exons may be too short to give a significant alignment with an mRNA-derived sequence. A third reason is that for distant relationships, the protein sequence alignment will generally contain more information (Table 4).

For noncoding regions, there are additional considerations in the choice of search algorithm. BLAST uses a very long word size for nucleic acid sequences. Also, the modifications to the BLAST heuristic that improve its sensitivity for protein sequences do not work well for nucleic acid sequences because nucleic acids have only a four-letter alphabet and the similarity scores are usually calculated with equal frequencies of replacement for all of the nucleotides (transitions equal transversions). Thus, FASTA is more sensitive than BLAST for even moderately diverged nucleic acid sequences and should be used instead of BLAST if a heuristic is desired.

Transition versus Transversion

It is possible to use different sets of evolutionary matrices for nucleic acids just as have been recommended for proteins. Two theoretical PAM models have been proposed, one using a uniform evolution model assuming equal rates of transitions and transversions and another using a biased model with a 3 to 1 transition to transversion ratio (29). Note that the uniform model at PAM 47 provides substitution values similar to the

Table 4. Significance Levels Using a Protein or a Nucleic Acid Query

Protein Percent Identity	Protein PAM Distance	Protein Sig. Len. (34 bits)	Protein Info. Per. Residue	N.A. to Protein Efficiency	N.A. Info. Per. Codon	N.A. Sig. Len. (41 bits)	N.A. PAM Distance	N.A. Percent Identity
100	0	8	4.17	**1.44**	**6.0**	**21**	**0**	**100**
83	20	12	2.95	**1.23**	**3.63**	**34**	**16**	**86**
63	**50**	**17**	**2.00**	**1.01**	**2.02**	**61**	**40**	**69**
43	**100**	**29**	**1.18**	**0.73**	0.86	143	80	51
32	**150**	**45**	**0.76**	**0.51**	0.39	315	120	40

Data derived from Dayhoff et al. (5) and States et al. (6).
N.A. = Nucleic Acid; Info. Per. = Information Per; Sig. Len. = Significant Length

ones historically recommended by Smith et al. (29).

There have also been a few empirical studies on gene-pseudogene evolution (7,16). These studies provide replacement frequencies that can be easily converted into empirical log-odds scoring matrices such as the BLOSUM protein matrices already described. These derived log-odds matrices may prove useful for researchers interested in searching for pseudogenes.

SCORING INSERTIONS AND DELETIONS

The gap penalty represents the probability that an insertion will start or continue relative to either a match or mismatch. Thus, the value of the gap penalty must be consistent with and coupled to the values contained within the substitution matrix. Furthermore, the value of the gap penalty must be consistent with the goal of the alignment—finding similar sequences in a database search or aligning sequences to provide the best estimate of the evolutionary history. There have been many approaches to quantify the gap penalty (12,32).

The most commonly used gap penalty consists of two terms, an open gap penalty and extended gap penalty. These terms allow the gap penalty to separate the cost of opening an insertion versus extending the preexisting gap. This two-term gap penalty performs better both in database searches (24) and in providing the best alignment (12). In practice, the open gap penalty is usually relatively large, while the extended gap penalty is more modest.

Database searching scores for related sequences are usually dominated by relative long ungapped segments. Thus, the gap penalty value has only a modest effect on computing the score for related sequences, but the expected score for a random sequence alignment increases with a decrease in the gap penalty. Decreasing the gap penalty increases the chances for identifying spurious sequences (decreasing selectivity). For database searches, a general rule of thumb is that the open gap penalty should be larger than the largest match score and twice the largest mismatch penalty, while the extend gap penalty is usually set at a minimal value of -1.

The final alignment represents the best estimate of the evolutionary history of changes that occurred during the divergence of the two sequences from their most recent ancestor. Studies of protein structural alignments (22) and pseudogenes (13) indicate that insertions and deletions are usually short (one or two residues in length) and become more frequent as the sequences become more highly diverged. Producing the best alignment usually requires reducing the open gap penalty and increasing the extended gap penalty, which results in the formation of small gaps in variable regions. Thus, the contrast between database searching and final alignment is easy to see—database searching identifies long, highly conserved regions, while the final alignment contains shorter conserved regions separated by small gaps.

STATISTICAL SIGNIFICANCE

A database search will yield a number of high-scoring alignments, which leads to the question: is the score or the length of the alignment of the query and database sequence greater than what one would expect from a random sequence? BLAST uses an explicit approach to assess significance, while FASTA and most implementations of the dynamic programming algorithm use a semi-empirical approach.

Explicit Approach: Using Information Theory and the Length of the Alignment

Each alignment contains a certain amount of information per position, which is approximately the product of the entropy of the matrix used (Table 3) and the length of the alignment (1,18,19) or, more exactly, the alignment score in bits. Current statistical models have not developed the ability to compute entropy for a gap. Thus, the information content of only ungapped alignments can be explicitly computed (18,19).

The amount of information in a database search required for an alignment to be significant is the base 2 logarithm of the product of the length of query sequence and the total number of residues in the database. From these two relationships, we can determine the minimum required length of an ungapped alignment (Equation 4).

$$\text{Minimum Significant Alignment Length} = \frac{\log_2 (\text{query length} \bullet \text{residues in db})}{\textit{entropy of matrix}} \quad \text{[Eq. 4]}$$

Table 5. Recommended Matrices to Be Used for a Database Search

PAM Matrices		BLOSUM Matrices	
3 Matrices	2 Matrices	3 Matrices	2 Matrices
PAM40	PAM120	BLOSUM60	BLOSUM60
PAM120	PAM250	BLOSUM40	BLOSUM35
PAM240		BLOSUM30	

Thus, using a 500-residue query sequence to search a 40 million-residue database using a BLOSUM62 matrix (Table 3) requires a minimum alignment of 36 residues to be significant. BLAST (maximal segment pairs) uses this approach, along with a similar sum statistic, to determine significance in the second step (Table 2). If the query sequence is shorter than the required minimum length, then no match from a database search can be judged to be significant, and one needs to select a higher entropy matrix.

Semi-Empirical Approach: Using the Database Search to Create a Reference Distribution

FASTA and various implementations of the dynamic programming algorithm (e.g., Smith-Waterman) allow the alignment to contain gaps. Thus, the explicit approach described above cannot be used to determine the significance of the alignment resulting from the query and a database sequence. A number of models has been developed to allow the estimation of the maximum score (cutoff score) for a random sequence to be aligned with the query sequence. The majority of these models are based on the simple observation that, with respect to any newly determined query sequence, the database contains primarily random sequences. That is, the pertinent random process is evolution from independent, unrelated ancestral sequences rather than from a common ancestral sequence.

The results of the database search are usually presented as a histogram of score versus frequency of occurrence. There is a large peak in the histogram representing the bulk of the database, the random sequence portion and then a number of isolated peaks representing the high-scoring hits. It is the large peak representing the random sequence portion of the database that needs to be fit. Two approaches will be described in detail. First, the histogram of scores for the random sequence portion of the database is fitted using an extreme value distribution. This allows the determination of the mean for the histogram, the size of a standard deviation and the computation of the maximum random score. Second, the decay after the peak of the histogram of scores is empirically observed to be exponential (9,10,32). When the log of the frequency is plotted, this decay is a straight line. Extrapolating this line to a frequency of one sequence (essentially the axis intercept) computes a score larger than might be reasonably expected from a random sequence in the database (9,10). Thus, any appreciably higher score is statistically rare. This is the approach used in many implementations of the dynamic programming algorithm.

Both approaches to determine the cutoff score are based on a well-defined random sequence portion of the database search. The shape of this region depends on the gap penalty being large enough (32). If the gap penalty is too low, the scores are observed to decay much more slowly, broadening the distribution of scores, which may confuse either approach to fit the histogram and obscure the significant hits.

All significance computations are based on a given probability of finding and aligning specific residues, usually the database composition. Thus, a region formed almost exclusively of one or two residues (e.g., a stretch as small as 20–30 residues of poly-C or poly-G) can skew the significance computation and lead to unrealistic results. Methods exist to remove these low-complexity regions before database searches (2).

FUTURE CONSIDERATIONS

It is not uncommon for only half of the related sequences to be identified in a trial database search with a well-characterized, large family of sequences (4,5,24). What happens when a database search reveals no related sequences? Are there no related sequences or are the already sequenced members of the family too diverse from the query to be identified in a database search? Assuming the latter, one needs to improve the sensitivity of the search. Radical improvement in sensitivity will require a major shift in the underlying model of evolution implied by the database search. There is a maximum sensitivity for the combination of search algorithm and substitution matrix that cannot be improved. The heuristics, which are approximations of the Smith-Waterman algorithm, are approaching the quality of the Smith-Waterman and probably have limited room for improvement. The evolution model expressed in the general database searching programs depends on residue-specific mutation rates regardless of the position of the residue in either the three-dimensional structure or the function of the residue in the protein. With this simplistic model, there is little overall difference between a highly diverged sequence and a random sequence.

In our opinion, the best hope for improving the sensitivity of the database searching algorithm is to improve the underlying evolutionary model, probably through the use of more sophisticated statistical models, such as the maximum likelihood alignments (6,31) or related methods using hidden Markov representations (8). These algorithms evaluate all possible alignments rather than exclusively consider the best scoring alignment. In general, highly diverged homologous sequences have many alignments that are almost as good as the best ones, while alignment scores for nonhomologous sequences decrease rapidly from the best score. Thus, addition over the alternative alignments allows more sensitive discrimination between highly diverged homologous sequences and unrelated, nonhomologous sequences. These algorithms are at least as computationally demanding as dynamic programming algorithms, and there are still no effective heuristic approximations available because the algorithms themselves are still at the research stage.

One would assume that finding related sequences should become easier in the future because the sequence databases are

doubling in size almost yearly, and several complete genomes have been sequenced. Thus, the potential number of hits should increase. While this number will increase, the nonhomologous portion of the database will increase at least as fast. Actually, this doubling of the sequence databases may increase the difficulty in identifying related sequences. This can be clearly seen when using the explicit approach to compute significance. Each year, an additional bit of information will be required for an alignment to be significant because the database doubles in size. Over the next five years, this will result in lengthening the minimum required ungapped alignment from 36 to almost 45 residues. One approach to ease the problems of growth in the database is to reduce its size by creating a nonredundant database. That is, for each cluster of highly identical sequences, one sequence is kept, while the others are removed. Alternatively, species- or genus-specific databases are also being created. While these approaches do limit the effects of the growth of the database, neither approach is ideal and can result in lost data. For example, sequentially searching a number of species-specific databases will provide a greater measure of significance than is actually valid.

Alternatively, one can use an abstraction to represent groups of sequences, creating what is in essence a nonredundant database. The group of sequences is usually a sequence family, while the abstraction can be a consensus sequence, weighted set representations, hidden Markov model (HMM) or position-specific scoring matrix (PSSM). This approach is now being used by the classification libraries ProSite, BLOCKS (PSSM), PRINTS (PSSM) and Pfam (HMM). Many of the abstractions allow a more accurate model for evolution, that is, defining some positions as rigorously conserved, while others allow random mutations.

CONCLUSION

In this paper, we have reviewed the four major aspects of sequence database searching: algorithm, substitution matrix, gap penalty and significance calculation. We have described how these components relate and how the choices affect the results of a database search. Based on this discussion, we recommend the following procedure for carrying out an effective database search: (*i*) use a local favorite program (Smith-Waterman, BLAST or FASTA) or the Web server of your choice; (*ii*) use at least two and preferably three similarity tables (Table 5); and (*iii*) if Smith-Waterman or FASTA is used, be sure the open gap penalty is large enough.

If the initial runs do not uncover any similar sequences and

a heuristic has been used, repeat the search using a highly diverged matrix and the Smith-Waterman algorithm. Alternatively, assuming an evolution-based matrix was used, use a highly diverged matrix from the other series (e.g., if three BLOSUM matrices were used, then try the PAM320).

To produce the final alignment between two sequences, choose a matrix that reflects the appropriate divergence for the two sequences (Table 2), reduce the gap penalty relative to the values used in the database search and use the MaxSegs (Waterman-Eggert) version of the dynamic programming algorithm to provide the best local alignment and also search for repeats in the sequences.

If there are questions about which matrix or open gap penalty should be used, try several choices to see which parts of the alignment are invariant and which parts are more suspect.

ACKNOWLEDGMENTS

This overview is based on a tutorial developed under a grant from the National Institutes of Health National Center for Research Resources grant no. P41 RR06009. This material has been presented in workshops supported by the Research Resource grant and as a part of a workshop on nucleic acid and protein sequence analysis presented annually at the PSC. The latter workshop is funded by grant no. T-15 HG00015 from the NIH's National Human Genome Research Institute. The authors acknowledge the helpful comments and discussions with the other Pittsburgh instructors: Dr. Michael Gribskov from the San Diego Supercomputer Center, Dr. Gary Churchill from Jackson Labs, Dr. Stephen Altschul from the NCBI, Dr. Michael Waterman from University of Southern California and Dr. Gary Stormo from Washington University in St. Louis. We would like to express our appreciation to the referees whose comments helped us more clearly express several important points in this overview.

REFERENCES

1. **Altschul, S.F.** 1991. Amino acid substitution matrices from an information theoretic perspective. J. Mol. Biol. *219*:555-665.
2. **Altschul, S.F., M.S. Boguski, W.Gish and J.C. Wooten.** 1994. Issues in searching molecular sequence databases. Nat. Genet. *6*:119-129.
3. **Altschul, S.F., W. Gish, W. Miller, E.W. Myers and D.J. Lipman.** 1990. Basic local alignment search tool. J. Mol. Biol. *215*:403-410.
4. **Altschul, S.F., T.L. Madden, A.A. Schaffer, J. Zhang, Z. Zhang, W. Miller and D.J. Lipman.** 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. *25*:3389-3402.
5. **Agarwal, P. and D.J. States.** 1998. Comparative accuracy of methods for protein sequence similarity search. Bioinformatics *14*:40-47.
6. **Bishop, M. and E. Thompson.** 1986. Maximum liklihood alignment of DNA sequences. J. Mol. Biol. *190*:159-165.
7. **Blake, R.D., S.T. Hess and J. Nicholson-Tuell.** 1992. The influence of nearest neighbor on the rate and pattern of spontaneous point mutations. J. Mol. Evol. *34*:189-200.
8. **Bucher, P. and K. Hoffman.** 1996. A sequence similarity algorithm based on a probabilistic interpretation of an alignment scoring system, p. 44-51. *In* D. States, P. Agarwal, T. Gaasterland, L. Hunter and R. Smith (Eds.), ISMB-4. AAAI Press, Menlo Park, CA.
9. **Collins, J.F. and A.F.W. Coulson.** 1988. The significance of protein sequence similarities, p. 474-487. *In* R.F. Doolittle (Ed.), Methods in Enzymology, Vol. 183, Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences. Academic Press, San Diego, CA.
10. **Collins, J.F., A.F.W. Coulson and A. Lyall.** 1988. The significance of protein sequence similarities. Comput. Appl. Biosci. *4*:67-71.
11. **Dayhoff, M.O., R.M. Schwartz and B.C. Orcutt.** 1978. A model of evolutionary change in proteins, p. 345-352. *In* M.O. Dayhoff (Ed.), Atlas of Protein Sequence and Structure. National Biomedical Research Foundation, Washington DC.
12. **Fitch, W.M. and T.F. Smith.** 1983. Optimal sequence alignments. Proc. Natl. Acad. Sci. USA *80*:1382-1386.
13. **Gu, X. and W.H. Li.** 1995. The size distribution of insertions and deletions in human and rodent pseudogenes suggests logarithmic gap penalties for sequence alignment. J. Mol. Evol. *4*:464-473.
14. **Henikoff, S. and J.G. Henikoff.** 1992 Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. USA. *89*:10915-10919.
15. **Henikoff, S. and J.G. Henikoff.** 1993. Performance evaluation of amino acid substitution matrices. Proteins: Structure, Function Genetics. *17*:49-61.
16. **Hess, S.T., J.D. Blake and R.D. Blake.** 1994. Wide variations in neighbor-dependent substitution rates. J. Mol. Biol. *236*:1022-1033.
17. **Johnson, M.S. and J.P. Overington.** 1993. A structural basis of sequence comparisons: an evaluation of scoring methodologies. J. Mol. Biol. *233*:716-738.
18. **Karlin, S. and S.F. Altschul.** 1990. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proc. Natl. Acad. Sci. USA *87*:2264-2268.
19. **Karlin, S. and S.F. Altschul.** 1993. Applications and statistics for multiple high-scoring segments in molecular sequences. Proc. Natl. Acad. Sci. USA *90*:5873-5877.
20. **Myers, E.W. and W. Miller.** 1988. Optimal alignments in linear space. Comput. Appl. Biosci. *4*:11-17.
21. **Needleman, S.B. and C.D. Wunsch.** 1970. A general method applicable to the search for similarities in the amino acid sequences of two proteins. J. Mol. Biol. *48*:443-453.
22. **Pascarella, S. and P. Argos.** 1992. Analysis of insertions/deletions in protein structures. J. Mol. Biol. *20*:461-471.
23. **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.
24. **Pearson, W.R.** 1995. Comparison of methods for searching protein sequences databases. Protein Science *4*:1145-1160.
25. **Pearson, W.R. and D.J. Lipman.** 1988. Improved tools for biological sequence comparison. Proc. Natl. Acad. Sci. USA. *85*:2444-2448.
26. **Sellers, P.H.** 1974. On the theory and computation of evolutionary distances. SIAM J. Appl. Math. *26*:787-793.
27. **Sellers, P.H.** 1979. Pattern recognition in genetic sequences. Proc. Natl. Acad. Sci. USA *76*:3041-3041.
28. **Smith, T.F. and M.S. Waterman.** 1981. Identification of common molecular subsequences. J. Mol. Biol. *147*:195-197.
29. **Smith, T.F., M.C. Waterman and C. Burkes.** 1985. The statistical distribution of nucleic acid similarities. Nucleic Acids Res. *13*:645-656.
30. **States, D.J., W. Gish and S.F. Altschul.** 1991. Improved sensitivity of nucleic acid database search using application-specific scoring matrices methods. Methods Enzymol. *3*:66-77.
31. **Thorne, J.L., H. Kishino and J. Felsenstein.** 1991. An evolutionary model for maximum likelihood alignment of DNA sequences. J. Mol Evol. *33*:114-124.
32. **Vingron, M. and M.S. Waterman.** 1994. Sequence alignment and penalty choice. J. Mol. Biol. *235*:1-12.
33. **Waterman, M.S. and M. Eggert.** 1987. A new algorithm for subsequence alignments with application to tRNA-rRNA comparisons. J. Mol. Biol. *197*:723-728.

Address correspondence to:

Dr. Hugh B. Nicholas Jr.
Pittsburgh Supercomputing Center
4400 Fifth Avenue
Pittsburgh, PA 15213, USA
Internet: Nicholas@psc.edu