CHAPTER 3

Alignment of Pairs of Sequences

INTRODUCTION

PAIR-WISE SEQUENCE ALIGNMENT IS a very large topic to cover as one chapter. Thus, starting with this chapter, more detailed discussions of topics, and information on subjects of more peripheral interest, will be available from the Web site for this book. This site is organized according to the same subject headings as this chapter and can be found at http://www.bioinformaticsonline.org. In addition, starting with this chapter, procedural flowcharts will appear at the beginning of the Methods section of most chapters to provide an overview of the methods of analysis. This chapter discusses pair-wise sequence alignment. Multiple sequence alignment is discussed in Chapter 4.

DEFINITION OF SEQUENCE ALIGNMENT

Sequence alignment is the procedure of comparing two (pair-wise alignment) or more (multiple sequence alignment) sequences by searching for a series of individual characters or character patterns that are in the same order in the sequences. Two sequences are aligned by writing them across a page in two rows. Identical or similar characters are placed in the same column, and nonidentical characters can either be placed in the same column as a mismatch or opposite a gap in the other sequence. In an optimal alignment, nonidentical characters and gaps are placed to bring as many identical or similar characters as possible into vertical register. Sequences that can be readily aligned in this manner are said to be similar.

There are two types of sequence alignment, global and local, and they are illustrated below in Figure 3.1. In global alignment, an attempt is made to align the entire sequence, using as many characters as possible, up to both ends of each sequence. Sequences that are quite similar and approximately the same length are suitable candidates for global alignment. In local alignment, stretches of sequence with the highest density of matches are aligned, thus generating one or more islands of matches or subalignments in the aligned sequences. Local alignments are more suitable for aligning sequences that are similar along some of their lengths but dissimilar in others, sequences that differ in length, or sequences that share a conserved region or domain.

Global Alignment

For the two hypothetical protein sequence fragments in Figure 3.1, the global alignment is stretched over the entire sequence length to include as many matching amino acids as possible up to and including the sequence ends. Vertical bars between the sequences indicate

Figure 3.1. Distinction between global and local alignments of two sequences.

the presence of identical amino acids. Although there is an obvious region of identity in this example (the sequence GKG preceded by a commonly observed substitution of T for A), a global alignment may not align such regions so that more amino acids along the entire sequence lengths can be matched.

Local Alignment

In a local alignment, the alignment stops at the ends of regions of identity or strong similarity, and a much higher priority is given to finding these local regions (Fig. 3.1) than to extending the alignment to include more neighboring amino acid pairs. Dashes indicate sequence not included in the alignment. This type of alignment favors finding conserved nucleotide patterns, DNA sequences, or amino acid patterns in protein sequences.

SIGNIFICANCE OF SEQUENCE ALIGNMENT

Sequence alignment is useful for discovering functional, structural, and evolutionary information in biological sequences. It is important to obtain the best possible or so-called "optimal" alignment to discover this information. Sequences that are very much alike, or "similar" in the parlance of sequence analysis, probably have the same function, be it a regulatory role in the case of similar DNA molecules, or a similar biochemical function and three-dimensional structure in the case of proteins. Additionally, if two sequences from different organisms are similar, there may have been a common ancestor sequence, and the sequences are then defined as being homologous. The alignment indicates the changes that could have occurred between the two homologous sequences and a common ancestor sequence during evolution, as shown in Figure 3.2.

With the advent of genome analysis and large-scale sequence comparisons, it becomes important to recognize that sequence similarity may be an indicator of several possible

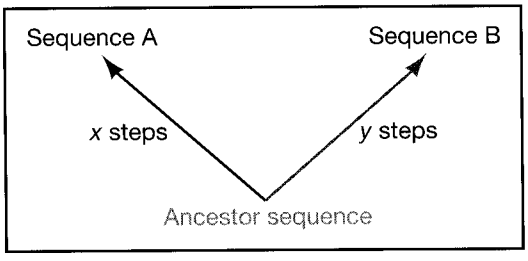


Figure 3.2. The evolutionary relationship between two similar sequences and a possible common ancestor sequence that would make the sequences homologous. The number of steps required to change one sequence to the other is the evolutionary distance between the sequences, and is also the sum of the number of steps to change the common ancestor sequence into one of the sequences (x) plus the number of steps required to change the common ancestor into the other (y). The common ancestor sequence is not available, such that x and y cannot be calculated; only $x + y$ is known. By the simplest definition, the distance $x + y$ is the number of mismatches in the alignment (gaps are not usually counted), as illustrated in Fig. 1.3. In a phylogenetic analysis of three or more similar sequences, the separate distances from the ancestor can be estimated, as discussed in Chapter 6.

types of ancestor relationships, or there may be no ancestor relationship at all, as illustrated in Figure 3.3. For example, new gene evolution is often thought to occur by gene duplication, creating two tandem copies of the gene, followed by mutations in these copies. In rare cases, new mutations in one of the copies provide an advantageous change in function. The two copies may then evolve along separate pathways. Although the resulting separation of function will generate two related sequence families, sequences among both families will still be similar due to the single gene ancestor. In addition, genetic rearrange-

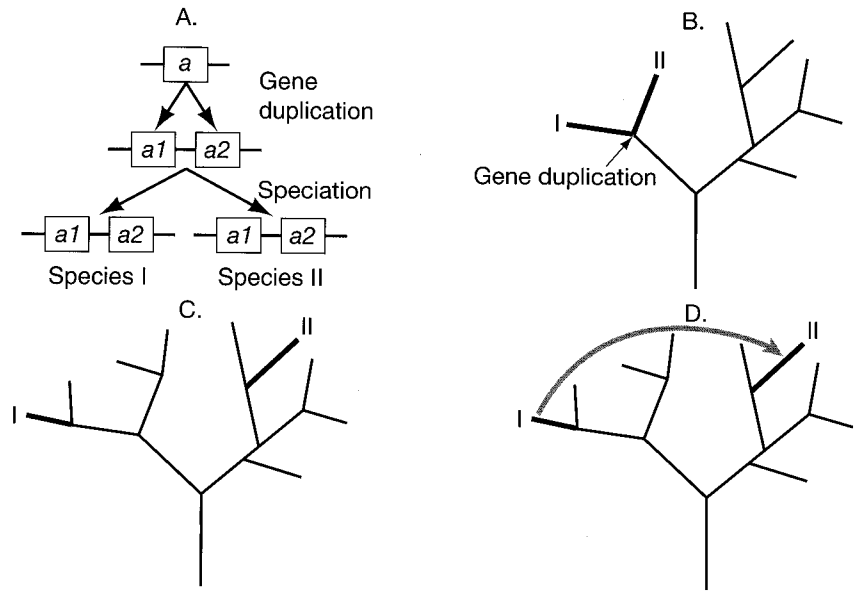


Figure 3.3. Origins of genes having a similar sequence. Shown are illustrative examples of gene evolution. In *A*, a duplication of gene *a* to produce tandem genes *a1* and *a2* in an ancestor of species I and II has occurred. Separation of the duplicated region by speciation gives rise to two separate branches, shown in *B* as blue and red. *a1* in species I and *a1* in species II are orthologous because they share a common ancestor. Similarly, *a2* in species I and *a2* in species II are orthologous. However, the *a1* genes are paralogous to the *a2* genes because they arose from a gene duplication event, indicated in *A*. If two or more copies of a gene family have been separated by speciation in this fashion, they tend to all undergo change as a group, due to gene conversion-type mechanisms (Li and Graur 1991). In *C*, a gene in species I and a different gene in species II have converged on the same function by separate evolutionary paths. Such analogous genes, or genes that result from convergent evolution, include proteins that have a similar active site but within a different backbone sequence. In *D*, genes in species I and II are related through the transfer of genetic material between species, even though the two species are separated by a long evolutionary distance. Although the transfer is shown between outer branches of the evolutionary tree, it could also have occurred in lower-down branches, thus giving rise to a group of organisms with the transferred gene. Such genes are known as xenologous or horizontally transferred genes. Transfer of the P transposable elements between *Drosophila* species is a prime example of such horizontal transfer (Kidwell 1983). Horizontal transfer also is found in bacterial genomes and can be traced as a regional variation in base composition within chromosomes. A similar type of transfer is that of the small ribosomal RNA subunits of mitochondria and chloroplasts, which originated from early prokaryotic organisms. Symbiotic relationships between organisms may be a precursor event leading to such exchanges. Other rearrangements within the genome (not shown) may produce chimeric genes comprising domains of genes that were evolving separately.

ments can reassort domains in proteins, leading to more complex proteins with an evolutionary history that is difficult to reconstruct (Henikoff et al. 1997).

Genes that are descended from a common ancestor are called homologs.

Evolutionary theory provides terms that may be used to describe sequence relationships. Homologous genes that share a common ancestry and function in the absence of any evidence of gene duplication are called orthologs. When there is evidence for gene duplication, the genes in an evolutionary lineage derived from one of the copies and with the same function are also referred to as orthologs. The two copies of the duplicated gene and their progeny in the evolutionary lineage are referred to as paralogs. In other cases, similar regions in sequences may not have a common ancestor but may have arisen independently by two evolutionary pathways converging on the same function, called convergent evolution. There are some remarkable examples in protein structures. For instance, although the enzymes chymotrypsin and subtilisin have totally different three-dimensional structures and folds, the active sites show similar structural features, including histidine (H), serine (S), and aspartic acid (D) in the catalytic sites of the enzymes (for discussion, see Branden and Tooze 1991). Additional examples are given in Chapter 10 (p. 509). In such cases, the similarity will be highly localized. Such sequences are referred to as analogous (Fitch 1970). A closer examination of alignments can help to sort out possible evolutionary origins among similar sequences (Tatusov et al. 1997).

It is important to describe these relationships accurately in publications. A common error in the molecular biology literature is to refer to sequence "homology" when one means sequence similarity. Sequence "similarity" is a measure of the matching characters in an alignment, whereas homology is a statement of common evolutionary origin.

As pointed out by Fitch and Smith (1983), sequences can be either homologous or nonhomologous, but not in between. The genetic rearrangements referred to above can give rise to chimeric genes, in which some regions are homologous and others are not. Referring to the entire sequences as homologous in such situations leads to an inaccurate and incomplete description of the sequence lineage.

Another complication in tracing the origins of similar sequences is that individual genes may not share the same evolutionary origin as the rest of the genome in which they presently reside. Genetic events such as symbioses and viral-induced transduction can cause horizontal transfer of genetic material between unrelated organisms. In such cases, the evolutionary history of the transferred sequences and that of the organisms will be different. Again, with the capability of detecting such events in the genomes of organisms comes the responsibility to describe these changes with the correct evolutionary terminology. In this case, the sequences are xenologous (Gray and Fitch 1983). Recently, Lawrence and Ochman (1997) have shown that horizontal transfer of genes between species is as common in enteric bacteria, if not more common, than mutation. Describing such changes requires a careful description of sequence origins. As discussed in Chapters 6 and 10, phylogenetic and other types of sequence analyses help to uncover such events.

OVERVIEW OF METHODS OF SEQUENCE ALIGNMENT

Alignment of Pairs of Sequences

Alignment of two sequences is performed using the following methods:

1. Dot matrix analysis
2. The dynamic programming (or DP) algorithm
3. Word or *k*-tuple methods, such as used by the programs FASTA and BLAST, described in Chapter 7.

Unless the sequences are known to be very much alike, the dot matrix method should be used first, because this method displays any possible sequence alignments as diagonals

on the matrix. Dot matrix analysis can readily reveal the presence of insertions/deletions and direct and inverted repeats that are more difficult to find by the other, more automated methods. The major limitation of the method is that most dot matrix computer programs do not show an actual alignment.

The dynamic programming method, first used for global alignment of sequences by Needleman and Wunsch (1970) and for local alignment by Smith and Waterman (1981a), provides one or more alignments of the sequences. An alignment is generated by starting at the ends of the two sequences and attempting to match all possible pairs of characters between the sequences and by following a scoring scheme for matches, mismatches, and gaps. This procedure generates a matrix of numbers that represents all possible alignments between the sequences. The highest set of sequential scores in the matrix defines an optimal alignment. For proteins, an amino acid substitution matrix, such as the Dayhoff percent accepted mutation matrix 250 (PAM250) or blosum substitution matrix 62 (BLOSUM62) is used to score matches and mismatches. Similar matrices are available for aligning DNA sequences.

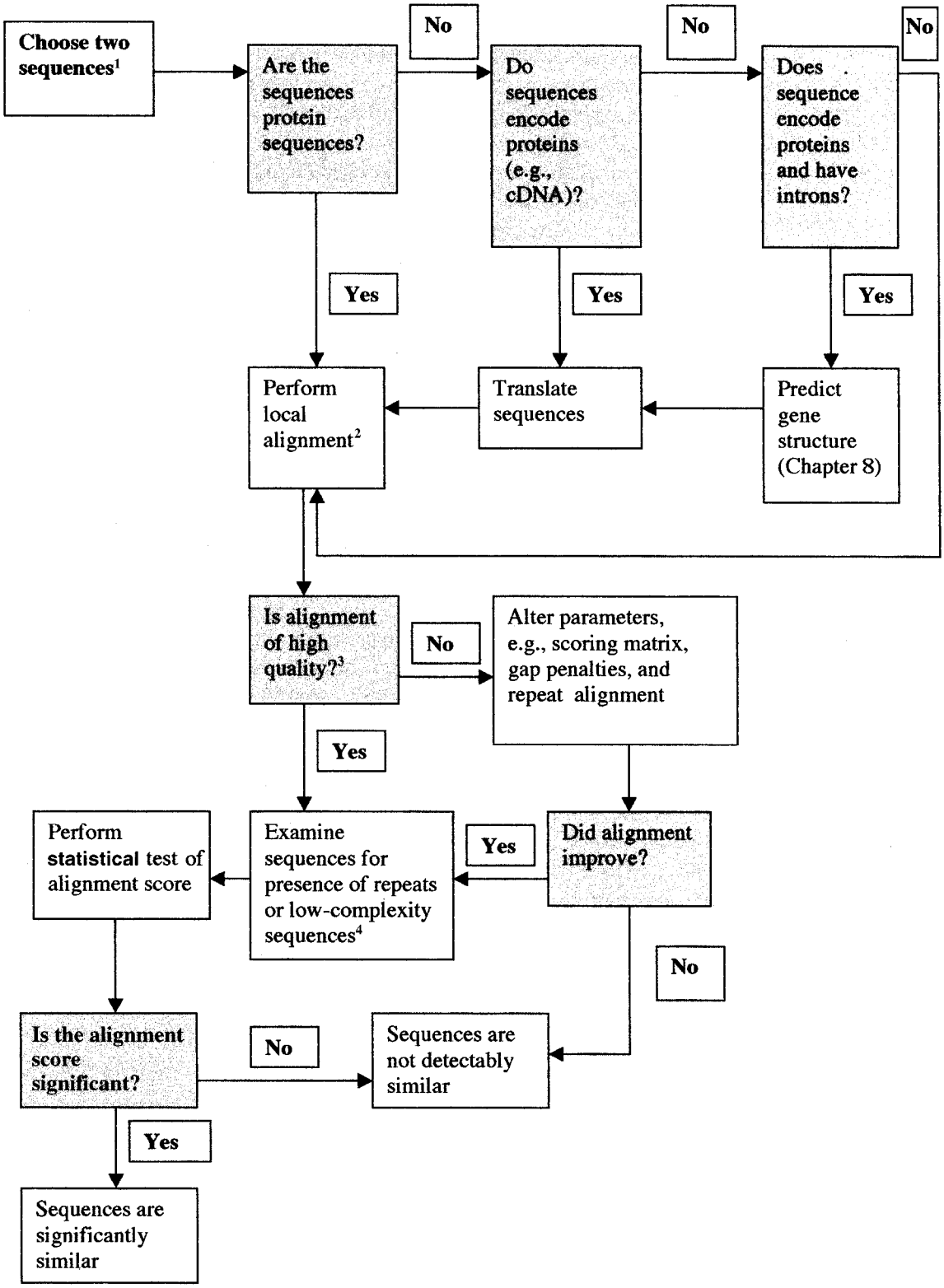
The dynamic programming method is guaranteed in a mathematical sense to provide the optimal (very best or highest-scoring) alignment for a given set of user-defined variables, including choice of scoring matrix and gap penalties. Fortunately, experience with the dynamic programming method has provided much help for making the best choices, and dynamic programming has become widely used. The dynamic programming method can also be slow due to the very large number of computational steps, which increase approximately as the square or cube of the sequence lengths. The computer memory requirement also increases as the square of the sequence lengths. Thus, it is difficult to use the method for very long sequences. Fortunately, computer scientists have greatly reduced these time and space requirements to near-linear relationships without compromising the reliability of the dynamic programming method, and these methods are widely used in the available dynamic programming applications to sequence alignment. Other shortcuts have been developed to speed up the early phases of finding an alignment.

The word or *k*-tuple methods are used by the FASTA and BLAST algorithms (see Chapter 7). They align two sequences very quickly, by first searching for identical short stretches of sequences (called words or *k*-tuples) and by then joining these words into an alignment by the dynamic programming method. These methods are fast enough to be suitable for searching an entire database for the sequences that align best with an input test sequence. The FASTA and BLAST methods are heuristic; i.e., an empirical method of computer programming in which rules of thumb are used to find solutions and feedback is used to improve performance. However, these methods are reliable in a statistical sense, and usually provide a reliable alignment.

Multiple Sequence Alignment

From a multiple alignment of three or more protein sequences, the highly conserved residues that define structural and functional domains in protein families can be identified. New members of such families can then be found by searching sequence databases for other sequences with these same domains. Alignment of DNA sequences can assist in finding conserved regulatory patterns in DNA sequences. Despite the great value of multiple sequence alignments, obtaining one presents a very difficult algorithmic problem. The methods that have been devised are discussed in Chapter 4.

METHODS


Choose two sequences[1]
Are the sequences protein sequences?
No
Do sequences encode proteins (e.g., cDNA)?
No
Does sequence encode proteins and have introns?
No
Yes
Yes
Yes
Perform local alignment[2]
Translate sequences
Predict gene structure (Chapter 8)
Is alignment of high quality?[3]
No
Alter parameters, e.g., scoring matrix, gap penalties, and repeat alignment
Yes
Perform statistical test of alignment score
Examine sequences for presence of repeats or low-complexity sequences[4]
Yes
Did alignment improve?
No
Is the alignment score significant?
No
Sequences are not detectably similar
Yes
Sequences are significantly similar

DOT MATRIX SEQUENCE COMPARISON

A dot matrix analysis is primarily a method for comparing two sequences to look for possible alignment of characters between the sequences, first described by Gibbs and McIntyre (1970). The method is also used for finding direct or inverted repeats in protein and DNA sequences, and for predicting regions in RNA that are self-complementary and that, therefore, have the potential of forming secondary structure. Every laboratory that does sequence analysis should have at least one dot matrix program available. In choosing a program, look for as many of the features described below as possible. The dot matrix should be visible on the computer terminal, thus providing an interactive environment so that different types of analyses may be tried. Use of colored dots can enhance the detection of regions of similarity (Maizel and Lenk 1981). Additional descriptions of the dot matrix method have appeared elsewhere (Doolittle 1986; States and Boguski 1991). The examples given below use the dot matrix module of DNA Strider (version 1.3) on a Macintosh computer. The program DOTTER has interactive features for the UNIX X-Windows environment (Sonnhammer and Durbin 1995; http://www.cgr.ki.se/cgr/groups/sonnhammer/Dotter.html). The Genetics Computer Group programs COMPARE and DOTPLOT also perform a dot matrix analysis. Although not a dot matrix method, the program PLALIGN in the FASTA suite may be used to display the alignments found by the dynamic programming method between two sequences on a graph (http://fasta.bioch.virginia.edu/fasta/fasta_list.html; Pearson 1990). A dot matrix program that may be used with a Web browser is described in Junier and Pagni (2000) (http://www.isrec.isb-sib.ch/java/ dotlet/Dotlet.html).

1. This chart assumes that both sequences are protein sequences or that both are DNA sequences. If one is a DNA sequence, that sequence should be translated and then aligned with the second, protein sequence.
2. The local alignment program, e.g., LALIGN or BESTFIT, usually has a recommended scoring matrix and gap penalty combination. It is important to make sure that the combination is one that is known to produce a confined, local alignment with random (or scrambled) sequences. A global alignment program may also be used with sequences of approximately the same length.
3. For protein sequences, a high-quality alignment is one that includes most of each sequence, a significant proportion (e.g., 25%) of identities throughout the alignment, multiple examples of conservative substitutions (chemically and structurally similar amino acids), and relatively few gaps confined to specific regions of the alignment. A poor-quality alignment includes only a portion of the sequences, has few and widely dispersed identities and conservative substitutions, tends to include regions of low complexity (repeats of same amino acid), and includes gaps that are obviously necessary to obtain the alignment. For DNA sequences, a significant alignment must include long runs of identities and few gaps. For two random or unrelated DNA sequences of length 100 and normal composition (0.25 of each base), the longest run of matches that can be expected is 6 or 7 (see text). A clue as to the significance of an alignment may also be obtained by using an alignment program that gives multiple alternative alignments, e.g., LALIGN. The first alignment found, which will be the highest scoring, should have a much higher score than the following ones, which are designed so that the same sequence positions will not be aligned a second time. Hence, these subsequent alignments should usually be random.
4. The result of this analysis can be a guide for the test of significance that follows. In the test described in this chapter, the second sequence is scrambled and realigned with the first sequence. Scrambling can be done at the level of the individual nucleotide or amino acid, or at the level of words by keeping the composition of short stretches of sequence intact.

Pair-wise Sequence Comparison

The major advantage of the dot matrix method for finding sequence alignments is that all possible matches of residues between two sequences are found, leaving the investigator the choice of identifying the most significant ones. Then, sequences of the actual regions that align can be detected by using one of two other methods for performing sequence alignments, e.g., dynamic programming. These methods are automatic and usually show one best or optimal alignment, even though there may be several different, nearly alike alignments. Alignments generated by these programs can be compared to the dot matrix alignment to determine whether the longest regions are being matched and whether insertions and deletions are located in the most reasonable places.

In the dot matrix method of sequence comparison, one sequence (A) is listed across the top of a page and the other sequence (B) is listed down the left side, as illustrated in Figures 3.4 and 3.5. Starting with the first character in B, one then moves across the page keeping in the first row and placing a dot in any column where the character in A is the same. The second character in B is then compared to the entire A sequence, and a dot is placed in row 2 wherever a match occurs. This process is continued until the page is filled with dots representing all the possible matches of A characters with B characters. Any region of similar sequence is revealed by a diagonal row of dots. Isolated dots not on the diagonal represent random matches that are probably not related to any significant alignment.

Detection of matching regions may be improved by filtering out random matches in a dot matrix. Filtering is achieved by using a sliding window to compare the two sequences. Instead of comparing single sequence positions, a window of adjacent positions in the two

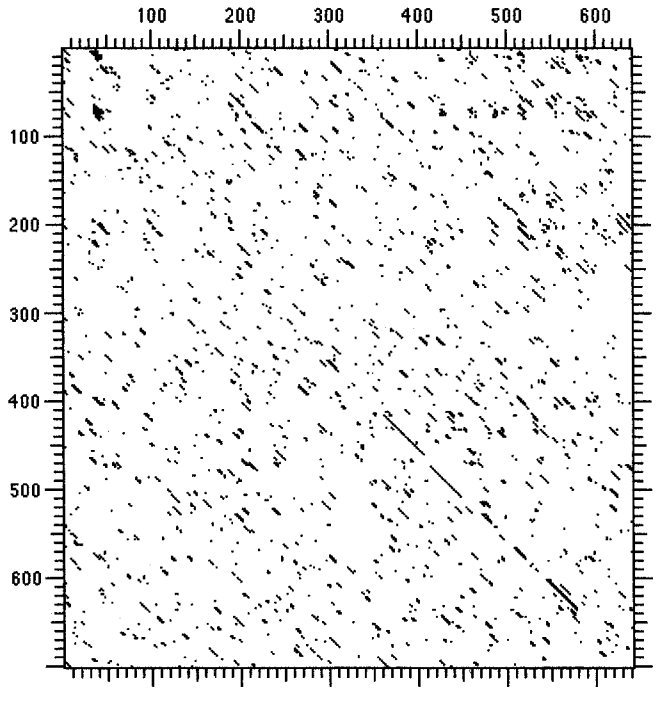


Figure 3.4. Dot matrix analysis of DNA sequences encoding phage λ *cI* (*vertical sequence*) and phage P22 *c2* (*horizontal sequence*) repressors. This analysis was performed using the dot matrix display of the Macintosh DNA sequence analysis program DNA Strider, vers. 1.3. The window size was 11 and the stringency 7, meaning that a dot is printed at a matrix position only if 7 out of the next 11 positions in the sequences are identified.

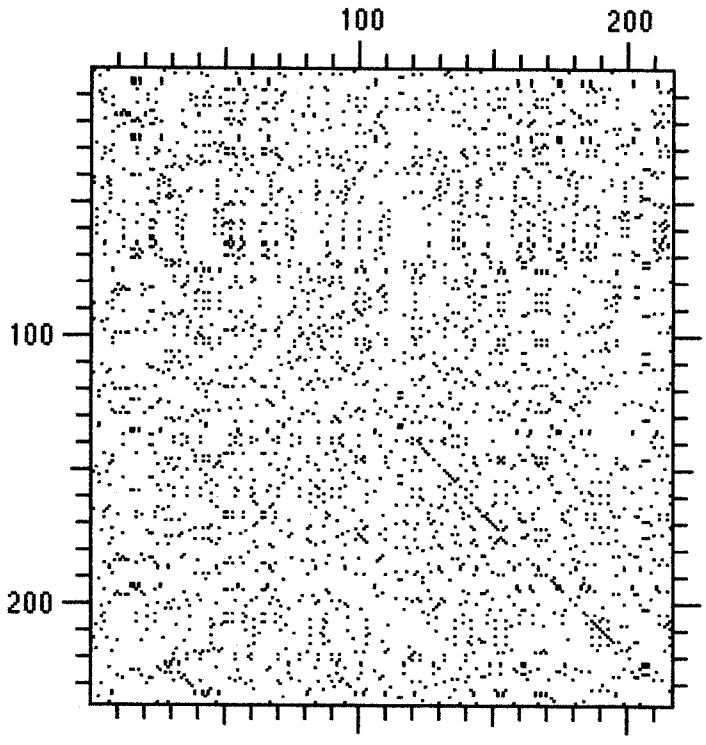


Figure 3.5. Dot matrix analysis of the amino acid sequences of the phage λ *cI* (*horizontal sequence*) and phage P22 *c2* (*vertical sequence*) repressors performed as described in Fig. 3.4. The window size and stringency were both 1.

sequences is compared at the same time, and a dot is printed on the page only if a certain minimal number of matches occur. The window starts at the positions in A and B to be compared and includes characters in a diagonal line going down and to the right, comparing each pair in turn, as in making an alignment. A larger window size is generally used for DNA sequences than for protein sequences because the number of random matches is much greater due to the use of only four DNA symbols as compared to 20 amino acid symbols. A typical window size for DNA sequences is 15 and a suitable match requirement in this window is 10. For protein sequences, the matrix is often not filtered, but a window size of 2 or 3 and a match requirement of 2 will highlight matching regions. If two proteins are expected to be related but to have long regions of dissimilar sequence with only a small proportion of identities, such as similar active sites, a large window, e.g., 20, and small stringency, e.g., 5, should be useful for seeing any similarity. Identification of sequence alignments by the dot matrix method can be aided by performing a count of dots in all possible diagonal lines through the matrix to determine statistically which diagonals have the most matches, and by comparing these match scores with the results of random sequence comparisons (Gibbs and McIntyre 1970; Argos 1987).

An example of a dot matrix analysis between the DNA sequences that encode the *Escherichia coli* phage λ *cI* and phage P22 *c2* repressor proteins is shown in Figure 3.4. With a window of 1 and stringency of 1, there is so much noise that no diagonals can be seen, but, as shown in the figure, with a window of 11 and a stringency of 7, diagonals appear in the lower left. The analysis reveals that there are regions of similarity in the 3′ ends of the coding regions, which, in turn, suggests similarity in the carboxy-terminal domains of the

encoded repressors. Note that sequential diagonals in matrix C do not line up exactly, indicating the presence of extra nucleotides in one sequence (the lambda *cI* gene on the vertical scale). The diagonals shown in the lower part of the matrix reveal a region of sequence similarity in the carboxy-terminal domains of the proteins. A small insertion in the cI protein that is approximately in the middle of this region and shifts the diagonal slightly downward accounts for this pattern.

An example of a dot matrix analysis between the amino acid sequences of the same two *E. coli* phage lambda cI and phage P22 c2 repressor proteins is shown in Figure 3.5. This matrix was filtered by a window of 1 and a stringency of 1. As found with the DNA sequence alignment of the corresponding genes, diagonals shown in the lower part of the matrix reveal a region of sequence similarity in the carboxy-terminal domains of the proteins. The small insertion in the cI protein approximately in the middle of this region which shifts the diagonal slightly downward and which is also observed in the DNA alignment of these corresponding genes is also visible. Note that these windows are much smaller than required for DNA sequence comparisons due to the greater number of possible symbols (20 amino acids) and therefore fewer random matches.

In conclusion, for DNA sequence dot matrix comparisons, use long windows and high stringencies, e.g., 7 and 11, 11 and 15. For protein sequences, use short windows, e.g., 1 and 1, for window and stringency, respectively, except when looking for a short domain of partial similarity in otherwise not-similar sequences. In this case, use a longer window and a small stringency, e.g., 15 and 5, for window and stringency, respectively.

There are three types of variations in the analysis of two protein sequences by the dot matrix method. First, chemical similarity of the amino acid R group or some other feature for distinguishing amino acids may be used to score similarity. Second, a symbol comparison table such as the PAM250 or BLOSUM62 tables may be used (States and Boguski 1991). These tables provide scores for matches based on their occurrence in aligned protein families. These tables are discussed later in this chapter (pages 78 and 85, respectively). When these tables are used, a dot is placed in the matrix only if a minimum similarity score is found. These table values may also be used in a sliding window option, which averages the score within the window and prints a dot only above a certain average score. Finally, several different matrices can be made, each with a different scoring system, and the scores can be averaged. This method should be useful for aligning more distantly related proteins. The scores of each possible diagonal through the matrix are then calculated, and the most significant ones are identified and shown on a computer screen (Argos 1987).

Sequence Repeats

Dot matrix analysis can also be used to find direct and inverted repeats within sequences. Repeated regions in whole chromosomes may be detected by a dot matrix analysis, and an interactive Web-based program has been designed for showing these regions at increasing levels of detail (http://genome-www.stanford.edu/Saccharomyces/SSV/viewer_start.html). Direct repeats may also be found by performing sequence alignments with dynamic programming methods (see next section). When used to align a sequence with itself, the program LALIGN will show alternative possible alignments between the repeated regions; PLALIGN will plot these alignments on a graph similar in appearance to a dot matrix (see http://fasta.bioch.virginia.edu/fasta/fasta-list.html; Pearson 1990). Here, the sequence is analyzed against itself and the presence of repeats is revealed by diagonal rows of dots. A Bayesian method for finding direct repeats is described on page 122. Inverted repeats require special handling and are discussed in Chapters 5 and 8. In Figure 3.6, an example of such an analysis for direct repeats in the amino acid sequence of the human low-density lipoprotein (LDL) receptor is shown. A list of additional proteins with direct repeats is

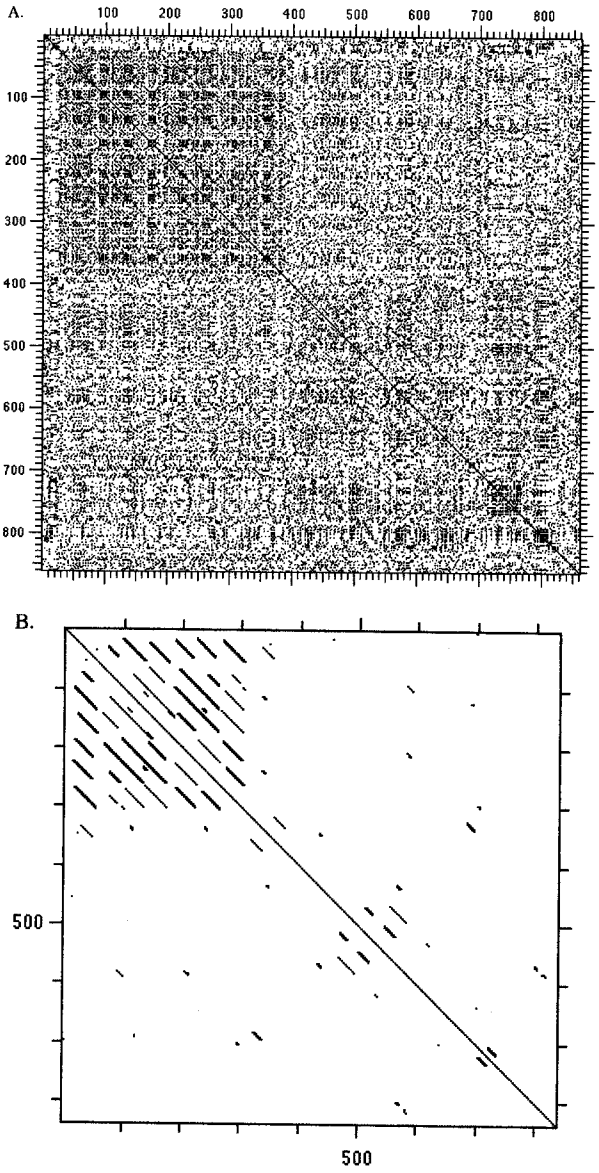


Figure 3.6. Dot matrix analysis of the human LDL receptor against itself using DNA Strider, vers. 1.3, on a Macintosh computer. (*A*) Window 1, Stringency 1. There is a diagonal line from upper left to lower right due to the fact that the same sequence is being compared to itself. The rest of the graph is symmetrical about this line. Other (quite hard to see) lines on either side of this diagonal are also present. These lines indicate repeated sequences perhaps 50 or so long. Patches of high-density dots, e.g., at the position corresponding to position 800 in both sequences representing short repeats of the same amino acid, are also seen. (*B*) Window 23, Stringency 7. The occurrence of longer repeats may be found by using this sliding window. In this example, a dot is placed on the graph at a given position only if 7/23 of the residues are the same. These choices are arbitrary and several combinations may need to be tried. Many repeats are seen in the first 300 positions. A pattern of approximate length 20 and at position 30 is repeated at least six times at positions 70, 100, 140, 180, 230, and 270. Two longer, overlapping repeats of length 70 are also found in this same region starting at positions 70 and 100, and repeated at position 200. Since few of these diagonals remain in new analyses at 11/23 (stringency/window) and all disappear at 15/23, they are not repeats of exactly the same sequence but they do represent an average of about 7/23 matches with no deletions or insertions. The information from the above dot matrix may be used as a basis for listing the actual amino acid repeats themselves by one of the other methods for sequence alignment described below.

given in Doolittle (1986, p. 50), and repeats are also discussed in States and Boguski (1991, p.109). As discussed in Chapters 9 and 10, there are many examples of proteins composed of multiple copies of a single domain.

Repeats of a Single Sequence Symbol

A dot matrix analysis can also reveal the presence of repeats of the same sequence character many times. These repeats become apparent on the dot matrix of a protein sequence against itself as horizontal or vertical rows of dots that sometimes merge into rectangular or square patterns. Such patterns are particularly apparent in the right and lower regions of the dot matrix of the human LDL receptor shown in Figure 3.6 but are also seen throughout the rest of the matrix. The occurrence of such repeats of the same sequence character increases the difficulty of aligning sequences because they create alignments with artificially high scores. A similar problem occurs with regions in which only a few sequence characters are found, called low-complexity regions. Programs that automatically detect and remove such regions from the analysis so that they do not interfere with database similarity searches are discussed in Chapter 7.

DYNAMIC PROGRAMMING ALGORITHM FOR SEQUENCE ALIGNMENT

Dynamic programming is a computational method that is used to align two protein or nucleic acid sequences. The method is very important for sequence analysis because it provides the very best or optimal alignment between sequences. Programs that perform this analysis on sequences are readily available, and there are Web sites that will perform the analysis. However, the method requires the intelligent use of several variables in the program. Thus, it is important to understand how the program works in order to make informed choices of these variables.

The method compares every pair of characters in the two sequences and generates an alignment. This alignment will include matched and mismatched characters and gaps in the two sequences that are positioned so that the number of matches between identical or related characters is the maximum possible. The dynamic programming algorithm provides a reliable computational method for aligning DNA and protein sequences. The method has been proven mathematically to produce the best or optimal alignment between two sequences under a given set of match conditions. Optimal alignments provide useful information to biologists concerning sequence relationships by giving the best possible information as to which characters in a sequence should be in the same column in an alignment, and which are insertions in one of the sequences (or deletions on the other). This information is important for making functional, structural, and evolutionary predictions on the basis of sequence alignments.

Both global and local types of alignments may be made by simple changes in the basic dynamic programming algorithm. A global alignment program is based on the Needleman-Wunsch algorithm, and a local alignment program on the Smith-Waterman algorithm, described below (p. 72). The predicted alignment will be given a score that gives the odds of obtaining the score between sequences known to be related to that obtained by chance alignment of unrelated sequences. There is a method to calculate whether or not an alignment obtained this way is statistically significant. One of the sequences may be scrambled many times and each randomly generated sequence may be realigned with the second sequence to demonstrate that the original alignment is unique. The statistical significance of alignment scores is discussed in detail below (p. 96).

Another feature of the dynamic programming algorithm is that the alignments obtained depend on the choice of a scoring system for comparing character pairs and penalty scores for gaps. For protein sequences, the simplest system of comparison is one based on identity. A match in an alignment is only scored if the two aligned amino acids are identical. However, one can also examine related protein sequences that can be aligned easily and find which amino acids are commonly substituted for each other. The probability of a substitution between any pair of the 20 amino acids may then be used to produce alignments. Recent improvements and experience with the dynamic programming programs and the scoring systems have greatly simplified their use. These enhancements are discussed below and at http://www.bioinformaticsonline.org.

It is important to recognize that several different alignments may provide approximately the same alignment score; i.e., there are alignments almost as good as the highest-scoring one reported by the alignment program. Some programs, e.g., LALIGN, provide several entirely different alignments with different sequence positions matched that can be compared to improve confidence in the best-scoring one. Alignment programs have also been greatly improved in algorithmic design and performance. With the advent of faster machines, it is possible to do a dynamic programming alignment between a query sequence and an entire sequence database and to find the similar sequences in several minutes. Dynamic programming has also been used to perform multiple sequence alignment, but only for a small number of sequences because the complexity of the calculations increases substantially for more than two sequences. Sequence alignment programs are available as a part of most sequence analysis packages, such as the widely used Genetics Computer Group GAP (global alignment) and BESTFIT (local alignment) programs. Sequences can also be pasted into a text area on a guest Web page on a remote host machine that will perform a dynamic programming alignment, and there are also versions of alignment programs that will run on a microcomputer (Table 3.1).

In deciding to perform a sequence alignment, it is important to keep the goal of the analysis in mind. Is the investigator interested in trying to find out whether two proteins have similar domains or structural features, whether they are in the same family with a related biological function, or whether they share a common ancestor relationship? The desired objective will influence the way the analysis is done. There are several decisions to be made along the way, including the type of program, whether to produce a global or local alignment, the type of scoring matrix, and the value of the gap penalties to be used. There are a very large number of amino acid scoring matrices in use (see book Web site), some much more popular than others, and these scoring matrices are designed for different purposes. Some, such as the Dayhoff PAM matrices, are based on an evolutionary model of protein change, whereas others, such as the BLOSUM matrices, are designed to identify members of the same family. Alignments between DNA sequences require similar kinds of considerations. It is often worth the effort to try several approaches to find out which choice of scoring system and gap penalty give the most reasonable result. Fortunately, most alignment programs come with a recommended scoring matrix and gap penalties that are useful for most situations. A more recent development (see Bayesian methods discussed on p. 124) is the simultaneous use of a set of scoring matrices and gap penalties by a method that generates the most probable alignments (see Table 3.1). The final choice as to the most believable alignment is up to the investigator, subject to the condition that reasonable decisions have been made regarding the methods used.

For sequences that are very similar, e.g., >95%, the sequence alignment is usually quite obvious, and a computer program may not even be needed to produce the alignment. As the sequences become less and less similar, the alignment becomes more difficult to produce and one is less confident of the result. For protein sequences, similarity can still be recognized down to a level of approximately 25% amino acid identity. At this level of iden-

Table 3.1. *Web sites for alignment of sequence pairs*

Name of site	Web address	Reference
Bayes block aligner	http://www.wadsworth.org/res&res/bioinfo	Zhu et al. (1998)
BCM Search Launcher: Pairwise sequence alignment[a]	http://dot.imgen.bcm.tmc.edu:9331/seq-search/alignment.html	see Web site
SIM—Local similarity program for finding alternative alignments	http://www.expasy.ch/tools/sim.html	Huang et al. (1990); Huang and Miller (1991); Pearson and Miller (1992)
Global alignment programs (GAP, NAP)	http://genome.cs.mtu.edu/align/align.html	Huang (1994)
FASTA program suite[b]	http://fasta.bioch.virginia.edu/fasta/fasta_list.html	Pearson and Miller (1992); Pearson (1996)
BLAST 2 sequence alignment (BLASTN, BLASTP)[c]	http://www.ncbi.nlm.nih.gov/gorf/bl2.html	Altschul et al. (1990)
Likelihood-weighted sequence alignment (lwa)[d]	http://www.ibc.wustl.edu/servive/lwa.html	see Web site

[a] This server provides access to a number of Web sites offering pair-wise alignments between nucleic acid sequences, protein sequences, or between a nucleic acid and a protein sequence.

[b] The FASTA algorithm normally used for sequence database searches (see Chapter 7) provides an alternative method to dynamic programming for producing an alignment between sequences. Briefly, all short patterns of a certain length are located in both sequences. If multiple patterns are found in the same order in both sequences, these provide the starting point for an alignment by the dynamic programming algorithm. Older versions of FASTA performed a global alignment, but more recent versions perform a local alignment with statistical evaluations of the scores. The program PLFASTA in the FASTA program suite provides a plot of the best matching regions, much like a dot matrix analysis, and thus gives an indication of alternative alignments. The FASTA suite is also available from Genestream at http://vega.igh.cnrs.fr/. Programs include ALIGN (global, Needleman-Wunsch alignment), LALIGN (local, Smith-Waterman alignment), LALIGNO (Smith-Waterman alignment, no end gap penalty), FASTA (local alignment, FASTA method), and PRSS (local alignment with scrambled copies of second sequence to do statistical analysis). Versions of these programs that run with a command-line interface on MS-DOS and Macintosh microcomputers are available by anonymous FTP from ftp.virginia.edu/pub/fasta.

[c] The BLAST algorithm normally used for database similarity searches (Chapter 7) can also be used to align two sequences.

[d] A description of the probabilistic method of aligning two sequences is described in Durbin et al. (1998) and Chapter 4. A related topic, hidden Markov models for multiple sequence alignments, is discussed in Chapter 4.

tity, the relative numbers of mismatched amino acids and gaps in the alignment have to be decided empirically and a decision made as to which gap penalties work the best for a given scoring matrix. Alignment of sequences at this level of identity is called the "twilight zone" of sequence alignment by Doolittle (1981). The alignment program may provide a quite convincing alignment, which suggests that the two sequences are homologous. The statistical significance of the alignment score may then be evaluated, as described later in this chapter.

Description of the Algorithm

Alignment of two sequences without allowing gaps requires an algorithm that performs a number of comparisons roughly proportional to the square of the average sequence length, as in a dot matrix comparison. If the alignment is to include gaps of any length at any position in either sequence, the number of comparisons that must be made becomes astronomical and is not achievable by direct comparison methods. Dynamic programming is a method of sequence alignment that can take gaps into account but that requires a manageable number of comparisons.

The method of sequence alignment by dynamic programming and the proof that the method provides an optimal (highest scoring) alignment are illustrated in Figures 3.7 and 3.8. To understand how the method works, we must first recall what is meant by an align-

```
sequence 1  V  D  S  -  C  Y
sequence 2  V  E  S  L  C  Y
SCORE       4  2  4 -11  9  7   SCORE = SUM OF AMINO ACID PAIR SCORES
(26)                            MINUS SINGLE GAP PENALTY (11) = 15
```

Figure 3.7. Example of scoring a sequence alignment with a gap penalty. The individual alignment scores are taken from an amino acid substitution matrix.

ment, using the two protein sequences shown in Figure 3.7 as an example. The two sequences will be written across the page, one under the other, the object being to bring as many amino acids as possible into register. In some regions, amino acids in one sequence will be placed directly below identical amino acids in the second. In other regions, this process may not be possible and nonidentical amino acids may have to be placed next to each other, or else gaps must be introduced into one of the sequences. Gaps are added to the alignment in a manner that increases the matching of identical or similar amino acids at subsequent portions in the alignment. Ideally, when two similar protein sequences are aligned, the alignment should have long regions of identical or related amino acid pairs and very few gaps. As the sequences become more distant, more mismatched amino acid pairs and gaps should appear.

The quality of the alignment between two sequences is calculated using a scoring system that favors the matching of related or identical amino acids and penalizes for poorly matched amino acids and gaps. To decide how to score these regions, information on the types of changes found in related protein sequences is needed. These changes may be expressed by the following probabilities: (1) that a particular amino acid pair is found in alignments of related proteins; (2) that the same amino acid pair is aligned by chance in the sequences, given that some amino acids are abundant in proteins and others rare; and (3) that the insertion of a gap of one or more residues in one of the sequences (the same as an insertion of the same length in the other sequence), thus forcing the alignment of each partner of the amino acid pair with another amino acid, would be a better choice. The ratio of the first two probabilities is usually provided in an amino acid substitution matrix. Each

```
1.  SCORE OF NEW      =   SCORE OF PREVIOUS +  SCORE OF NEW
    ALIGNMENT             ALIGNMENT (A)        ALIGNED PAIR

V  D  S  -  C  Y          V  D  S  -  C           Y
V  E  S  L  C  Y          V  E  S  L  C           Y

      15              =         8             +     7


II. SCORE OF          =   SCORE OF PREVIOUS +  SCORE OF NEW
    ALIGNMENT (A)         ALIGNMENT (B)        ALIGNED PAIR

V  D  S  -  C             V  D  S  -              C
V  E  S  L  C             V  E  S  L              C

      8               =        -1             +     9

III. REPEAT REMOVING ALIGNED PAIRS UNTIL END OF ALIGNMENT IS REACHED.
```

Figure 3.8. Derivation of the dynamic programming algorithm.

table entry gives the ratio of the observed frequency of substitution between each possible amino acid pair in related proteins to that expected by chance, given the frequencies of the amino acids in proteins. These ratios are called odds scores. The ratios are transformed to logarithms of odds scores, called log odds scores, so that scores of sequential pairs may be added to reflect the overall odds of a real to chance alignment of an alignment. Examples are the Dayhoff PAM250 and BLOSUM62 substitution matrices described below (p. 76). These matrices contain positive and negative values, reflecting the likelihood of each amino acid substitution in related proteins. Using these tables, an alignment of a sequential set of amino acid pairs with no gaps receives an overall score that is the sum of the positive and negative log odds scores for each individual amino acid pair in the alignment. The higher this score, the more significant is the alignment, or the more it resembles alignments in related proteins. The score given for gaps in aligned sequences is negative, because such misaligned regions should be uncommon in sequences of related proteins. Such a score will reduce the score obtained from an adjacent, matching region upstream in the sequences. The score of the alignment in Figure 3.7, using values from the BLOSUM62 amino acid substitution matrix and a gap penalty score of -11 for a gap of length 1, is 26 (the sum of amino acid pair scores) $-11 = 15$. The value of -11 as a penalty for a gap of length 1 is used because this value is already known from experience to favor the alignment of similar regions when the BLOSUM62 comparison matrix is used. Choice of the gap penalty is discussed further below where a table giving suitable choices is presented (see Table 3.10 on p. 113). As shown in the example, the presence of the gap decreases significantly the overall score of the alignment.

Calculating the Odds Score of an Alignment from the Odds Scores of Individual Amino Acid Pairs

Sequence alignment scores are based on the individual scores of all amino acid pairs in the alignment. The odds score for an amino acid pair is the ratio of the observed frequency of occurrence of that pair in alignments of related proteins over the expected frequency based on the proportion of amino acids in proteins. Alignments are built by making possible lists of amino acid pairs and by finding the most likely list using odds scores. To calculate the odds score for an alignment, the odds scores for the individual pairs are multiplied. This calculation is similar to finding the probability of one event AND also a second independent event by multiplying the probabilities (if one event OR another is the choice, then the probabilities are added). Thus, if the odds score of C/C is 7/1 and that of W/W is 50/1, then the probability of C/C and W/W being in the alignment is $7/1 \times 50/1 = 350/1$ (note that the order or position in the alignment does not matter). Usually, log odds scores are used in these calculations, and these scores are added to produce an overall log odds score for the alignment. To perform this optimal alignment using odds scores, the method assumes that the odds score for matching a given pair of sequence positions is not influenced by the odds score of any other matching pair; i.e., that there are no correlations expected among the amino acids found at various sequence positions. Another way of describing this assumption is that the sequences are each being modeled as a Markov chain, with the amino acid found at each position not being influenced by other amino acids in the sequence. Although correlations among sequence positions are expected, since they give rise to structure and function in molecules, this simplifying assumption allows the determination of a reasonable alignment between the sequences.

Although one may be able to align the two short sequences in Figure 3.7 by eye and to place the gap where shown, the dynamic programming algorithm will automatically place gaps in much longer sequence alignments so as to achieve the best possible alignment. The derivation of the dynamic programming algorithm is illustrated in Figure 3.8, using the above alignment as an example. Consider building this alignment in steps, starting with an initial matching aligned pair of characters from the sequences (V/V) and then sequentially adding a new pair until the alignment is complete, at each stage choosing a pair from all the possible matches that provides the highest score for the alignment up to that point. If the full alignment finally reached on the left side of Figure 3.8 (I) has the highest possible or optimal score, then the old alignment from which it was derived (A) by addition of the aligned Y/Y pair must also have been optimal up to that point in the alignment. If this were incorrect, and a different preceding alignment other than A was the highest scoring one, then the alignment on the left would also not be the highest scoring alignment, and we started with that as a known condition. Similarly, in Figure 3.8 (II), alignment A must also have been derived from an optimal alignment (B) by addition of a C/C pair. In this manner, the alignment can be traced back sequentially to the first aligned pair that was also an optimal alignment. One concludes that the building of an optimal alignment in this stepwise fashion can provide an optimal alignment of the entire sequences.

The example in Figure 3.8 also illustrates two of the three choices that can be made in adding to an alignment between two sequences: Match the next two characters in the next positions in each sequence, or match the next character to a gap in the upper sequence. The last possibility, not illustrated, is to add a gap to the lower sequence. This situation is analogous to performing a dot matrix analysis of the sequences, and of either continuing a diagonal or of shifting the diagonal sideway or downward to produce a gap in one of the sequences. An example of using the dynamic programming algorithm to align two short protein sequences is illustrated in Figure 3.9.

Formal Description of the Dynamic Programming Algorithm

The algorithm (Fig. 3.9) may be written in mathematical form, as shown in Figure 3.10. The diagram indicates the moves that are possible to reach a certain matrix position (i,j) starting from the previous row and column at position $(i-1, j-1)$ or from any position in the same row and column.

The following equation describes the algorithm that was illustrated in Figure 3.9. There are three paths in the scoring matrix for reaching a particular position, a diagonal move from position $i-1, j-1$ to position i, j with no gap penalties, or a move from any other position from column j or row i, with a gap penalty that depends on the size of the gap. For two sequences $\mathbf{a} = a_1a_2 \ldots a_n$ and $\mathbf{b} = b_1\, b_2 \ldots b_m$, where $S_{ij} = S(a_1a_2 \ldots a_i, b_1b_2..b_j)$ then (Smith and Waterman 1981a,b)

$$S_{ij} = \max \left\{ \begin{array}{l} S_{i-1,j-1} + s(a_i b_j), \\ \max\limits_{x \geq 1} (S_{i-x,j} - w_x), \\ \max\limits_{y \geq 1} (S_{ij-y} - w_y) \end{array} \right\} \qquad (1)$$

where S_{ij} is the score at position i in sequence **a** and position j in sequence **b**, $s(a_ib_j)$ is the score for aligning the characters at positions i and j, w_x is the penalty for a gap of length x

in sequence **a**, and w_y is the penalty for a gap of length y in sequence **b**. Note that S_{ij} is a type of running best score as the algorithm moves through every position in the matrix. Eventually, when all of the matrix positions (all S_{ij}) have been filled, the best score of the alignment will be found as the highest scoring position in the last row and column (for a global alignment), after correcting for any remaining gap penalties to align the sequence ends, if applicable. To determine an optimal alignment of the sequences from the scoring matrix, a second matrix called the trace-back matrix is used (Fig. 3.9). The trace-back matrix keeps track of the positions in the scoring matrix that contributed to the highest overall score found. The sequence characters corresponding to these high scoring positions may align or may be next to a gap, depending on the information in the trace-back matrix. An example of this procedure can be found on the book Web site.

Use of the dynamic programming method requires a scoring system for the comparison of symbol pairs (nucleotides for DNA sequences and amino acids for protein sequences), and a scheme for insertion/deletion (GAP) penalties. Once those parameters have been set, the resulting alignment for two sequences should always be the same. Scoring matrices are

Figure 3.9. Example of using the dynamic programming algorithm to align sequences a1 a2 a3 a4 and b1 b2 b3 b4.

1. The sequences are written across the top and down the left side of a matrix, respectively, similar to that done in the dot matrix analysis, except that an extra row and column labeled "gap" are added to allow the alignment to begin with a gap of any length in either sequence. The gap rows are filled with penalty scores for gaps of increasing lengths, as indicated. A zero is placed in the upper right box corresponding to no gaps in either sequence.
2. Maximum possible values are calculated for all other boxes below and to the right of the top row and left column, taking into account any sized gap or no gap, using the steps listed in *a* through *d* below. The scores for individual matches a1-b1, a1-b2, etc., are obtained from a scoring matrix (symbol comparison table). To calculate the value for a particular matrix position, trial values are calculated from all moves into that position allowed by the algorithm. The allowed moves are from any position above or to the left of the current position, in the same column or row, or from the upper left diagonal position. The diagonal move attempts to align the sequence characters without introducing a gap. Thus, there is no gap penalty in this case. However, moves from above and to the left will introduce gaps, and thus will require one or more gap penalties to be used. (*a*) s11 is the score for an a1-b1 match added to 0 in the upper left position. According to the algorithm, there are two other possible paths to this position shown by the vertical and horizontal arrows, but they would probably have to give a lower score because they start at a gap penalty and must include an additional gap penalty. (*b*) Trial values for s12 are calculated and the maximum score is chosen. Trial 1 is to add the score for the a1-b2 match to s11 and subtract a penalty for a gap of size 1. The other three trials shown by arrows include gap penalties and so likely cannot yield a higher score than trial 1. (*c*) All possible scores for s21 are calculated by the trial moves indicated. The best score should be obtained by adding the score of an a2-b1 match to s11 since all other moves include gap penalties. (*d*) Trial values of s22 are calculated by considering moves from s11, s21, and s12, and from the top row and left end column. s22 will be the best score of several possible choices, including adding the score for an a2-b2 match to s11, or to s21 less a single gap penalty. Other trials will normally be attempted from other positions above and to the left of this position, but in this case, they will probably not provide a higher score for s22 because they include multiple gap penalties.
3. As the maximum scores for each matrix position are calculated, a record of the paths that produced the highest scores to reach each matrix position is kept. These short paths, which represent extending the alignment to another matching pair, with or without gaps, are recorded in another matrix called the trace-back matrix, illustrated below. For example, if moving from s11 to s21 gave the highest score of all moves to s21, then the corresponding region of the matrix will appear as shown.
4. The paths in the trace-back matrix are joined to produce an alignment. In the example shown, the highest-scoring matrix position in the sequence comparison matrix is located, in this case s44, and the arrows are then traced back as far as possible, generating the path shown. The corresponding alignment A is shown below the matrix. More than one alignment may be possible if there is more than one path from the highest scoring matrix position. As an example, s43 could also be a high-scoring position, generating trace-back alignment B, an alignment that includes a gap opposite a2. Another gap may also be placed opposite b4, which has no matching symbol. Scoring end gaps is optional in the alignment programs. If

Legend continues.

1.

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap				
b2	2 gaps				
b3	3 gaps				
b4	4 gaps				

2a.

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11			
b2	2 gaps				
b3	3 gaps				
b4	4 gaps				

2b.

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11			
b2	2 gaps	s12			
b3	3 gaps				
b4	4 gaps				

2c.

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11	s21		
b2	2 gaps	s12			
b3	3 gaps				
b4	4 gaps				

2d.

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11	s21		
b2	2 gaps	s12	s22		
b3	3 gaps				
b4	4 gaps				

3. Part of trace back matrix

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11	s21	s31	s41
b2	2 gaps	s12	s22	s32	s42
b3	3 gaps	s13	s23	s33	s43
b4	4 gaps	s14	s24	s34	s44

4. Trace back matrix

	gap	a1	a2	a3	a4
gap	0	1 gap	2 gaps	3 gaps	4 gaps
b1	1 gap	s11	s21 B	s31	s41
b2	2 gaps	s12	s22	s32	s42
b3	3 gaps	s13	s23 A	s33	s43
b4	4 gaps	s14	s24	s34	s44

Alignment A: a1 a2 a3 a4
b1 b2 b3 b4

Alignment B: a1 a2 a3 a4 –
b1 – b2 b3 b4

included in this case, alignment B would be disfavored by an additional gap penalty. In addition to this series of alignments, or so-called clump of alignments starting from the highest scoring position, there will be other possible alignments starting from other high-scoring matrix positions, and these may also have multiple pathways through the scoring matrix, each representing a different alignment. Note that these alignments are global alignments because they include the entire sequences.

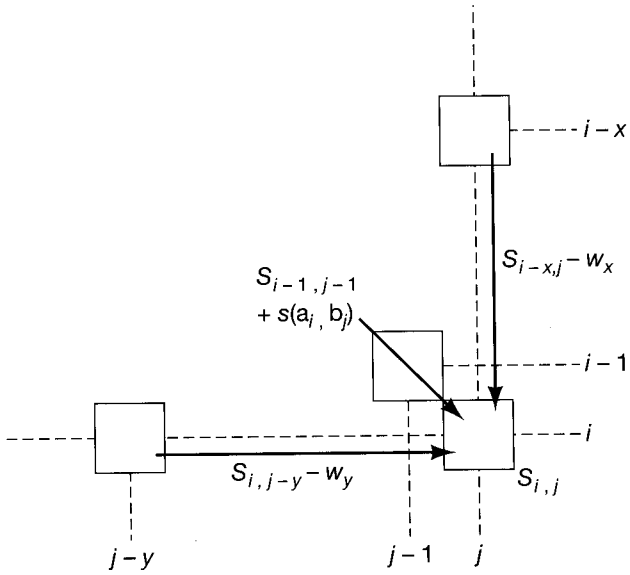


Figure 3.10. Formal description of the dynamic programming algorithm.

described below. The most commonly used ones for protein sequence alignments are the log odds form of the PAM250 matrix and the BLOSUM62 matrix. However, a number of other choices are available.

Dynamic Programming Can Provide Global or Local Sequence Alignments

Global Alignment: Needleman-Wunsch Algorithm

The dynamic programming method as described above gives a global alignment of sequences, as described by Needleman and Wunsch (1970), but was also proven mathematically and extended to include an improved scoring system by Smith and Waterman (1981a,b). The optimal score at each matrix position is calculated by adding the current match score to previously scored positions and subtracting gap penalties, if applicable. Each matrix position may have a positive or negative score, or 0. The Needleman-Wunsch algorithm will maximize the number of matches between the sequences along the entire length of the sequences. Gaps may also be present at the ends of sequences, in case there is extra sequence left over after the alignment. These end gaps are often, but not always, given a gap penalty. The effect of these penalties is illustrated below. An example of a global alignment of two short sequences calculated by hand using the algorithm is shown on the book Web site. The example also reveals that more than one alignment may be equally as likely.

Local Alignment: Smith-Waterman Algorithm

A modification of the dynamic programming algorithm for sequence alignment provides a local sequence alignment giving the highest-scoring local match between two sequences (Smith and Waterman 1981a,b). Local alignments are usually more meaningful than global matches because they include patterns that are conserved in the sequences. They can also be used instead of the Needleman-Wunsch algorithm to match two sequences that may

have a matched region that is only a fraction of their lengths, that have different lengths, that overlap, or where one sequence is a fragment or subsequence of the other. The rules for calculating scoring matrix values are slightly different, the most important differences being (1) the scoring system must include negative scores for mismatches, and (2) when a dynamic programming scoring matrix value becomes negative, that value is set to zero, which has the effect of terminating any alignment up to that point. The alignments are produced by starting at the highest-scoring positions in the scoring matrix and following a trace path from those positions up to a box that scores zero. The mathematical formulation of the dynamic programming algorithm is revised to include a choice of zero as the minimum value at any matrix position. For two sequences $\mathbf{a} = a_1a_2 \ldots a_n$ and $\mathbf{b} = b_1 b_2 \ldots b_n$, where $H_{ij} = H(a_1a_2 \ldots a_i, b_1b_2..b_j)$, then (Smith and Waterman 1981a)

$$H_{ij} = \max \begin{cases} H_{i-1,j-1} + s(a_ib_j), \\ \max\limits_{x \geq 1} (H_{i-x,j} - w_x), \\ \max\limits_{y \geq 1} (H_{i,j-y} - w_y), \\ 0 \end{cases} \quad (2)$$

where H_{ij} is the score at position i in sequence **a** and position j in sequence **b**, $s(a_ib_j)$ is the score for aligning the characters at positions i and j, w_x is the penalty for a gap of length x in sequence **a**, and w_y is the penalty for a gap of length y in sequence **b**.

To illustrate the difference between the Needleman-Wunsch and Smith-Waterman methods, a local alignment of the same two sequences is shown on the book Web site.

Does a Local Alignment Program Always Produce a Local Alignment and a Global Alignment Program Always Produce a Global Alignment?

Although a computer program that is based on the above Smith-Waterman local alignment algorithm is used for producing an optimal alignment, this feature alone does not assure that a local alignment will be produced. The scoring matrix or match and mismatch scores and the gap penalties chosen also influence whether or not a local alignment is obtained. Similarly, a program based on the Needleman-Wunsch algorithm can also return a local alignment depending on the weighting of end gaps and on other scoring parameters. Often, one can simply inspect the alignment obtained to see how many gaps are present. If the matched regions are long and cover most of the sequences and obviously depend on the presence of many gaps, the alignment is global. A local alignment, on the other hand, will tend to be shorter and not include many gaps, just as in the example given on the book Web site. However, these tests are quite subjective, and a more precise method of knowing whether a given program and set of scoring parameters will provide a local or global alignment is required. Looking ahead in the chapter for a moment, the best way of knowing is by looking at what happens when many random or completely unrelated sequences are aligned under the chosen conditions. As the length of the random sequences being aligned increases, the score of a global alignment will just increase proportionally.

This is easy to see. Because a global alignment matches most of the sequence, and the negative mismatch score and gap penalties are deliberately chosen to be small in comparison to match scores in order to provide a long alignment, only matches count and the score has to be proportional to the length.

If using a scoring matrix, a matrix that gives on the average a positive score to each aligned position, combined with a small enough gap penalty to allow extension of the alignment through poorly matched regions, will give a global alignment. Conversely, for the local alignment, a negative mismatch score and gap penalties are chosen to balance the positive score of a match and to prevent the alignment growing into regions that do not match very well. The scoring matrix in this case will on the average give a negative value to the matched positions, and the gap penalty will be large enough to prevent gaps from extending the alignment. The local alignment score of random sequences does not increase proportionally to sequence length, because the positive score of matches is offset by the mismatch and penalty scores. In this case, it may be shown by theory and experiment that the score of local random alignments increases much more slowly, and proportionally to the logarithm of the product of the sequence lengths. It is this different behavior of the alignment score of random sequences with length that distinguishes global and local alignments.

One may well ask, Does it really matter whether I use a sequence alignment program based on the global alignment algorithm or one based on the local alignment algorithm? The answer is that sometimes both methods will provide the same alignment with the same scoring system and sometimes they will not. The most reasonable approach is to use a program based on the appropriate algorithm for the analysis at hand, and then to choose the scoring system carefully. Small changes in the scoring system can abruptly change an alignment from a local to a global one. There are even examples in the bioinformatics literature where this feature of alignment scoring systems has been overlooked. The rest of this chapter is designed to provide a suitable guide for making the right choices.

Additional Development and Use of the Dynamic Programming Algorithm for Sequence Alignments

Use of Distance Scores for Sequence Alignment

As originally designed by Needleman and Wunsch and Smith and Waterman, the dynamic programming algorithm was used for sequence alignments scored on the basis of the similarity or identity of sequence characters. An alternative method is to score alignments based on differences between sequences and sequence characters; i.e., how many changes are required to change one sequence into another. Using this measure, the greater the distance between sequences, the greater the evolutionary time that has elapsed since the sequences diverged from a common ancestor. Hence, distance scores provide a more biologically natural way to compare sequences than do similarity scores. Using a distance scoring scheme, Sellers (1974, 1980) showed that the dynamic programming method could be used to provide an alignment that highlighted the evolutionary changes. Smith et al. (1981) and Smith and Waterman (1981b) showed that alignments based on a similarity scoring scheme could give a similar alignment. This analysis is discussed further on the book Web site. Conversion between distance and similarity scores is discussed in Chapter 6.

Improvement in Speed and Memory Requirement for the Dynamic Programming Algorithm

The dynamic programming methods for sequence alignments originally required between $n \times m$ and $n \times m^2$ steps and storage in several matrices of size $n \times m$, where n is the length of the shorter sequence (Needleman and Wunsch 1970; Waterman et al. 1976; Smith and Waterman 1981a). On the book Web site, a series of improvements in this algorithm that reduced the number of steps and amount of memory required are described. These steps include: (1) a decreased number of steps in the alignment algorithm by Gotoh (1982); (2) a reduction in the amount of memory required to a linear function of sequence length (Myers and Miller 1988); (3) ability to find near-optimal alignments (Chao et al. 1994) and to align long sequences (Schwartz et al. 1991); and (4) ability to find the best-scoring alternative alignments that do not include alignments of the same sequence positions (Waterman and Eggert 1987; Huang et al. 1990; Huang and Miller 1991).

The alignment programs listed in Table 3.1 include these features.

An alternative global alignment is found by giving the matrix position that begins with an alignment score of zero, and then all matrix positions that are affected by this change are recalculated. The next highest matrix score and the path leading to it provide an alternative alignment of the sequences that does not include the same sequence matches as were present in the original alignment (Waterman and Eggert 1987). Alternative local alignments are found by a more complex algorithm (the SIM algorithm) that includes the improvements listed above (Huang et al. 1990; Huang and Miller 1991).

Examples of Global and Local Alignments

An example of global and local alignments between two phage repressor proteins using the Genetics Computer Group (GCG) programs GAP (Needleman-Wunsch algorithm) and BESTFIT (Smith-Waterman algorithm) is shown in Figure 3.11. Note that the proteins are 58% similar in the carboxy-terminal domain, which is the region required for protein–protein interactions and a self-cleavage function that leads to phage induction. In these GCG implementations of the Needleman-Wunsch and Smith-Waterman algorithms, the alignments found in the carboxy-terminal domain are identical. However, the Smith-Waterman method (B) only reports the most alike regions, as expected by the focus on a local alignment strategy. In contrast, the Needleman-Wunsch method shows the entire alignment of the sequences but reports a lower score of similarity due to the longer alignment.

LALIGN (Fig. 3.12) is an implementation of the SIM algorithm for finding multiple unique (nonintersecting) alignments in DNA and protein sequences (Huang and Miller 1991) distributed in the FASTA package from W. Pearson. The program is also available on Web sites (see Table 3.1). Two features of these alignments are noteworthy: First, the highest-scoring alignment is similar to that found by the GAP program using a different amino acid substitution matrix and different gap penalties, with some minor variations in the more dissimilar regions and extension of the alignment farther into the amino-terminal domains. Second, by design, the alternative alignments never align the same amino acids and, in this example, the second and third alignments score much lower than the first one. These observations that strongly aligning regions are not significantly influenced by the scoring system, and that alternative high-scoring alignments are not possible, add convincing support that the initial alignment represents true similarity between these sequences. Another example of an alignment of these same sequences using ALIGN with a different scoring system is given on page 116.

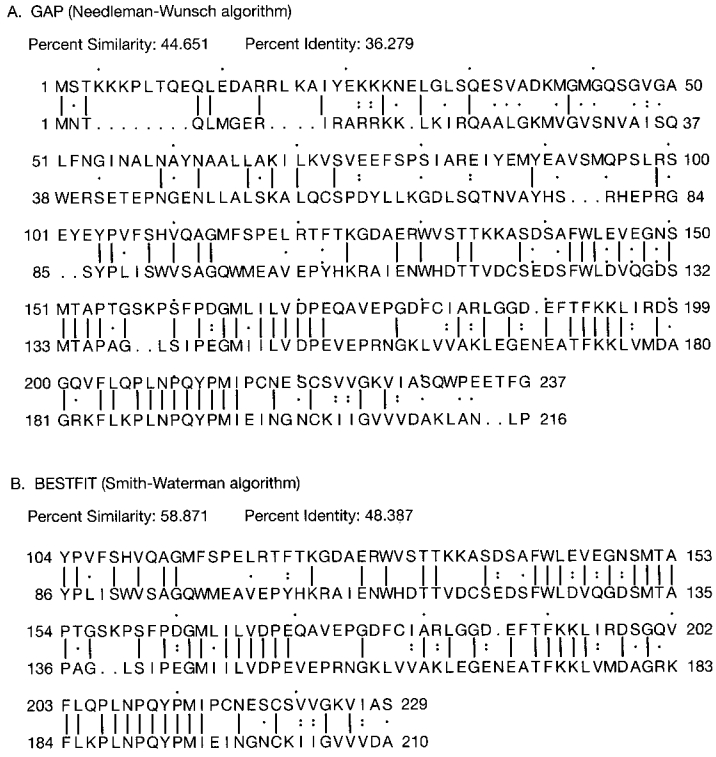


Figure 3.11. Example of local alignment of phage λ *cI* and phage P22 *c2* repressors by dynamic programming using the GCG GAP (Needleman-Wunsch algorithm) and BESTFIT (Smith-Waterman algorithm) programs. The log odds form of the PAM120 amino acid substitution matrix was used. PAM120 is optimal for proteins that are ~40% similar. The alignment reveals that the proteins are similar in the carboxy-terminal domain. The penalty for opening a gap in one of the sequences is 11 and for extending the gap 8; these were the default values assigned by the programs. Gaps at the unaligned ends of sequences were also weighted. In the program output, percent identity indicates the number of identical amino acids in the alignment, and percent similarity, the number of similar amino acids. Similar amino acids are defined by high-scoring matches between the amino acid pairs in the substitution matrix, and were defined at the time the program was run. The most similar pairs were indicated by a ':', less similar pairs by a '.' and unrelated pairs by a space, ' ', between the amino acid pairs. Although these dynamic programming programs provide a single optimal alignment, it is important to realize that a series of alignments are usually possible. Other programs, such as ALIGN in the FASTA set (Table 3.1 ALIGN-SITES), provide a user-specified number of alignments (see Fig. 3.12). Additionally, the alignments depend on the method used by the program to convert the traceback matrix into an alignment. GCG programs GAP and BESTFIT provide a method for printing two extremes of alignment, depending on whether gaps are favored in one sequence or the other. These options are called high road and low road.

USE OF SCORING MATRICES AND GAP PENALTIES IN SEQUENCE ALIGNMENTS

Amino Acid Substitution Matrices

Protein chemists discovered early on that certain amino acid substitutions commonly occur in related proteins from different species. Because the protein still functions with these substitutions, the substituted amino acids are compatible with protein structure and function. Often, these substitutions are to a chemically similar amino acid, but other changes also occur. Yet other substitutions are relatively rare. Knowing the types of changes that are most and least common in a large number of proteins can assist with predicting alignments for any set of protein sequences, as illustrated in Figure 3.13. If related

```
LALIGN finds the best local alignments between two sequences
 version 2.0u64 March 1998
Please cite:
 X. Huang and W. Miller (1991) Adv. Appl. Math. 12:373-381

 Comparison of:
(A) lamc1.pro  LAMC1  REFORMAT of: cipro.pro  from: 1 - 237
(B) p22c2.pro  P22C2  REFORMAT of: p22c2.pro  from: 1 - 216
 using matrix file: pam250.mat, gap penalties: -12/-2

  34.0% identity in 206 aa overlap; score:  338

          30        40        50        60        70        80
LAMC1   KKNELGLSQESVADKMGMGQSGVGALFNGINALNAYNAALLAKILKVSVEEFSPSIAREI
        ....: ..:.......:... ....  .. .. :. :   :.: :. : . .  .  ...
P22C2   RRKKLKIRQAALGKMVGVSNVAISQWERSETEPNGENLLALSKALQCSPDYLLKGDLSQT
              20        30        40        50        60        70

           90       100       110       120       130       140
LAMC1   YEMYEAVSMQPSLRSEYEYPVFSHVQAGMFSPELRTFTKGDAERWVSTTKKASDSAFWLE
          :..   ...     .::..: : :: .  .. .. : . :.: .:: . :...:::.
P22C2   NVAYHSRHEPRG-----SYPLISWVSAGQWMEAVEPYHKRAIENWHDTTVDCSEDSFWLD
             80              90        100       110       120

         150       160       170       180       190       200
LAMC1   VEGNSMTAPTGSKPSFPDGMLILVDPEQAVEPGDFCIARLGGD-EFTFKKLIRDSGQVFL
        :.:.:::::.:   :.:.::.:::::: .   :.. .:.:.:. : ::::..:.:. ::
P22C2   VQGDSMTAPAG--LSIPEGMIILVDPEVEPRNGKLVVAKLEGENEATFKKLVMDAGRKFL
        130       140       150       160       170       180

           210       220       230
LAMC1   QPLNPQYPMIPCNESCSVVGKVIASQ
        .::::::::: :..:...: :....
P22C2   KPLNPQYPMIEINGNCKIIGVVVDAK
          190       200       210

----------

  17.8% identity in 90 aa overlap; score:  37

          20        30        40        50        60        70
LAMC1   RRLKAIYEKKNELGLSQESVAD-KMGMGQSGVGALFNGINALNAYNAALLAKILKVSVE
        ..::   .   . .:.:. .... ... .... ..:..  .::.   . ::   :. .
P22C2   KKLKIRQAALGKMVGVSNVAISQWERSETEPNGENLLALSKALQCSPDYLLKGDLSQTNV
            20        30        40        50        60        70

            80        90       100
LAMC1   EF-SPSIAREIYEMYEAVSMQPSLRSEYEY
        .. :.  .:. : . . ::  . . .   :
P22C2   AYHSRHEPRGSYPLISWVSAGQWMEAVEPY
            80        90       100

----------

  40.0% identity in 15 aa overlap; score:  36

      220       230
LAMC1   SCSVVGKVIASQWPE
        :..... : :.:: :
P22C2   SYPLISWVSAGQWME
            90
```

Figure 3.12. Example of LALIGN program for finding multiple local alignments of two protein sequences. Three independent alignments of the phage λ and P22 repressors are shown. The amino acid substitution matrix used was the log odds form of the Dayhoff PAM250 matrix provided with the program, with a gap opening penalty of −12 and a gap extension penalty of −2.

protein sequences are quite similar, they are easy to align, and one can readily determine the single-step amino acid changes. If ancestor relationships among a group of proteins are assessed, the most likely amino acid changes that occurred during evolution can be predicted. This type of analysis was pioneered by Margaret Dayhoff (1978).

Amino acid substitution matrices or symbol comparison tables, as they are sometimes called, are used for such purposes. Although the most common use of such tables is for comparison of protein sequences, other tables of nucleic acid symbols are also used for comparison of nucleic acid sequences in order to accommodate ambiguous nucleotide

	Alignment			
sequence A	Tyr	Cys	Asp	Ala
sequence B	Phe	Met	Glu	Gly
BLOSUM62 matrix value	3	−1	2	0

Total score for alignment of sequence A with sequence B
= 3 − 1 + 2 + 0 = 4

Figure 3.13. Use of amino acid substitution matrix to evaluate an alignment of two protein sequences. The score for each amino acid pair (Tyr/Phe, etc.) is looked up in the BLOSUM62 matrix. Each value represents an odds score, the likelihood that the two amino acids will be aligned in alignments of similar proteins divided by the likelihood that they will be aligned by chance in an alignment of unrelated proteins. In a series of individual matches in an alignment, these odds scores are multiplied to give an overall odds score for the alignment itself. For convenience, odds scores are converted to log odds scores so that the values for amino acid pairs in an alignment may be summed to obtain the log odds score of the alignment. In this case, the logarithms are calculated to the base 2 and multiplied by 2 to give values designated as half-bits (a bit is the unit of an odds score that has been converted to a logarithm to the base 2). The value of 4 indicates that the 4 amino acid alignment is $2^{(4/2)}$ = 4-fold more likely than expected by chance.

characters or models of expected sequence changes during different periods of evolutionary time that vary scoring of transitions and transversions.

In the amino acid substitution matrices, amino acids are listed both across the top of a matrix and down the side, and each matrix position is filled with a score that reflects how often one amino acid would have been paired with the other in an alignment of related protein sequences. The probability of changing amino acid A into B is always assumed to be identical to the reverse probability of changing B into A. This assumption is made because, for any two sequences, the ancestor amino acid in the phylogenetic tree is usually not known. Additionally, the likelihood of replacement should depend on the product of the frequency of occurrence of the two amino acids and on their chemical and physical similarities. A prediction of this model is that amino acid frequencies will not change over evolutionary time (Dayhoff 1978).

Dayhoff Amino Acid Substitution Matrices (Percent Accepted Mutation or PAM Matrices)

This family of matrices lists the likelihood of change from one amino acid to another in homologous protein sequences during evolution. There is presently no other type of scoring matrix that is based on such sound evolutionary principles as are these matrices. Even though they were originally based on a relatively small data set, the PAM matrices remain a useful tool for sequence alignment. Each matrix gives the changes expected for a given period of evolutionary time, evidenced by decreased sequence similarity as genes encoding the same protein diverge with increased evolutionary time. Thus, one matrix gives the changes expected in homologous proteins that have diverged only a small amount from each other in a relatively short period of time, so that they are still 50% or more similar. Another gives the changes expected of proteins that have diverged over a much longer period, leaving only 20% similarity. These predicted changes are used to produce optimal alignments between two protein sequences and to score the alignment. The assumption in this evolutionary model is that the amino acid substitutions observed over short periods of

evolutionary history can be extrapolated to longer distances. The BLOSUM matrices (see below) are based on scoring substitutions found over a range of evolutionary periods and reveal that substitutions are not always as predicted by the PAM model.

In deriving the PAM matrices, each change in the current amino acid at a particular site is assumed to be independent of previous mutational events at that site (Dayhoff 1978). Thus, the probability of change of any amino acid **a** to amino acid **b** is the same, regardless of the previous changes at that site and also regardless of the position of amino acid **a** in a protein sequence. Amino acid substitutions in a protein sequence are thus viewed as a Markov model (see also hidden Markov models in Chapter 4), characterized by a series of changes of state in a system such that a change from one state to another does not depend on the previous history of the state. Use of this model makes possible the extrapolation of amino acid substitutions observed over a relatively short period of evolutionary time to longer periods of evolutionary time.

To prepare the Dayhoff PAM matrices, amino acid substitutions that occur in a group of evolving proteins were estimated using 1572 changes in 71 groups of protein sequences that were at least 85% similar. Because these changes are observed in closely related proteins, they represent amino acid substitutions that do not significantly change the function of the protein. Hence they are called "accepted mutations," defined as amino acid changes "accepted" by natural selection. Similar sequences were first organized into a phylogenetic tree, as illustrated in Figure 1.1 in Chapter 1. The number of changes of each amino acid into every other amino acid was then counted. To make these numbers useful for sequence analysis, information on the relative amount of change for each amino acid was needed.

Relative mutabilities were evaluated by counting, in each group of related sequences, the number of changes of each amino acid and by dividing this number by a factor, called the exposure to mutation of the amino acid. This factor is the product of the frequency of occurrence of the amino acid in that group of sequences being analyzed and the total number of all amino acid changes that occurred in that group per 100 sites. This factor normalizes the data for variations in amino acid composition, mutation rate, and sequence length. The normalized frequencies were then summed for all sequence groups. By these scores, Asn, Ser, Asp, and Glu were the most mutable amino acids, and Cys and Trp were the least mutable.

The above amino acid exchange counts and mutability values were then used to generate a 20×20 mutation probability matrix representing all possible amino acid changes. Because amino acid change was modeled by a Markov model, the mutation at each site being independent of the previous mutations, the changes predicted for more distantly related proteins that have undergone N mutations could be calculated. By this model, the PAM1 matrix could be multiplied by itself N times, to give transition matrices for comparing sequences with lower and lower levels of similarity due to separation of longer periods of evolutionary history. Thus, the commonly used PAM250 matrix represents a level of 250% of change expected in 2500 my. Although this amount of change seems very large, sequences at this level of divergence still have about 20% similarity. For example, alanine will be matched with alanine 13% of the time and with another amino acid 87% of the time.

Do not confuse this mutation probability form of the PAM250 matrix with the log odds form of the matrix described below.

The percentage of remaining similarity for any PAM matrix can be calculated by summing the percentages for amino acids not changing (Ala versus Ala, etc.) after multiplying each by the frequency of that amino acid pair in the database (e.g., 0.089 for Ala) (Dayhoff 1978). The PAM120, PAM80, and PAM60 matrices should be used for aligning sequences that are 40%, 50%, and 60% similar, respectively. Simulations by George et al. (1990) have shown that, as predicted, the PAM250 matrix provides a better-scoring alignment than lower-numbered PAM matrices for distantly related proteins of 14–27% similarity.

PAM matrices are usually converted into another form, called log odds matrices. The odds score represents the ratio of the chance of amino acid substitution by two different hypotheses—one that the change actually represents an authentic evolutionary variation at that site (the numerator), and the other that the change occurred because of random sequence variation of no biological significance (the denominator). Odds ratios are converted to logarithms to give log odds scores for convenience in multiplying odds scores of amino acid pairs in an alignment by adding the logarithms (Fig. 3.13).

Example: Calculations for obtaining the log odds score for changes between Phe and Tyr at an evolutionary distance of 250 PAMs

1. Of 1572 observed amino acid changes, there were 260 changes between Phe and Tyr. These numbers were multiplied by (1) the relative mutability of Phe (see text), and (2) the fraction of Phe to Tyr changes over all changes of Phe to any other amino acid (since Phe to Tyr and Tyr to Phe changes are not distinguished in the original mutation counts, sums of changes are used to calculate the fraction) to obtain a mutation probability score of Phe to Tyr. A similar score was obtained for changes of Phe to each of the other 18 amino acids, and also for the calculated probability of not changing at all. The resulting 20 scores were summed and divided by a normalizing factor such that their sum represented a probability of change of 1%, as illustrated in Table 3.2.

   In this matrix, the score for changing Phe to Tyr was 0.0021, as opposed to a score of Phe not changing at all of 0.9946, as shown in Table 3.2. These calculations were repeated for Tyr changing to any other amino acid. The score for changing Tyr to Phe was 0.0028, and that of not changing Tyr was 0.9946 (not shown). These scores were placed in the PAM1 matrix, in which the overall probability of each amino acid changing to another is ~1%, and that of each not changing is ~99%.
2. The above PAM1 matrix was multiplied by itself 250 times to obtain the distribution of changes expected for 250 PAMs of evolutionary change. These changes can include both forward changes to another amino acid and reverse changes to a former one. At this distance, the probability of change of Phe to Tyr was 0.15 as opposed to a probability of 0.32 of no change in Phe. The corresponding probabilities for Tyr to Phe at 250 PAMs were 0.20 and 0.31 for no change.
3. The log odds values for changes between Phe and Tyr were then calculated. The Phe-Tyr score in the 250 PAM matrix, 0.15, was divided by the frequency of Phe in the sequence data, 0.040, to give the relative frequency of change. This ratio, $0.15/0.04 = 3.75$, was converted to a logarithm to the base 10 ($\log_{10}3.75 = 0.57$) and multiplied by 10 to remove fractional values ($0.57 \times 10 = 5.7$). Similarly, the Tyr to Phe score is $0.20/0.03 = 6.7$, and the logarithm of this number is $\log_{10}6.7 = 0.83$, and multiplied by 10 ($0.83 \times 10 = 8.3$). The average of 5.7 and 8.3 is 7, the number entered in the log odds table for changes between Phe and Tyr at 250 PAMs of evolutionary distance.

   The log odds from the PAM250 matrix, which is sometimes referred to as the mutation data matrix (MDM) at 250 PAMs and also as MDM_{78}, is shown in Figure 3.14. The log odds scores in this table lie within the range of −8 to +17. A value of 0 indicates that the frequency of the substitution between a matched pair of amino acids in related proteins is as expected by chance; a value less than 0 or greater than 0 indicates that the frequency is less than or greater than that expected by chance, respectively. Using such a matrix, a high positive score

between two amino acids means that the pair is more likely to be found aligned in sequences that are derived from a common ancestor, i.e., homologous, than in unrelated or nonhomologous sequences. The highest-scoring replacements are for amino acids whose side chains are chemically similar, as might be expected if the amino acid substitution is not to impede function. In the original data, the largest number of observed changes (83) was between Asp (D) and Glu (E). This number is reflected as a log odds score of +3 in the MDM. Many changes were not observed. For example, there were no changes between Gly (G) and Trp (W), resulting in a score of −7 in the table.

Table 3.2. *Normalized probability scores for changing Phe to any other amino acid (or of not changing) at PAM1 and PAM250 evolutionary distances*

Amino acid change	PAM1	PAM250
Phe to Ala	0.0002	0.04
Phe to Arg	0.0001	0.01
Phe to Asn	0.0001	0.02
Phe to Asp	0.0000	0.01
Phe to Cys	0.0000	0.01
Phe to Gln	0.0000	0.01
Phe to Glu	0.0000	0.01
Phe to Gly	0.0001	0.03
Phe to His	0.0002	0.02
Phe to Ile	0.0007	0.05
Phe to Leu	0.0013	0.13
Phe to Lys	0.0000	0.02
Phe to Met	0.0001	0.02
Phe to Phe	0.9946	0.32
Phe to Pro	0.0001	0.02
Phe to Ser	0.0003	0.03
Phe to Thr	0.0001	0.03
Phe to Trp	0.0001	0.01
Phe to Tyr	0.0021	0.15
Phe to Val	0.0001	0.05
SUM[a]	1.0000	1.00

[a]Approximate since scores are rounded off.

The multiplication of two PAM1 matrices to give a PAM2 matrix. Only three rows and columns are shown for illustrative purposes.

	aa1	aa2	aa3 →
aa1	a	b	c
aa2	d	e	f
aa3	g	h	i
↓			

×

	aa1	aa2	aa3 →
aa1	a	b	c
aa2	d	e	f
aa3	g	h	i
↓			

=

	aa1	aa2	aa3 →
aa1	A	B	C
aa2	D	E	F
aa3	G	H	I
↓			

$A = a^2 + bd + cg + \ldots$

$B = ab + be + ch + \ldots$

$C = ac + bf + ci + \ldots$

$D = da + ed + fg + \ldots$, etc.

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	12																				C
S	0	2																			S
T	−2	1	3																		T
P	−3	1	0	6																	P
A	−2	1	1	1	2																A
G	−3	1	0	−1	1	5															G
N	−4	1	0	−1	0	0	2														N
D	−5	0	0	−1	0	1	2	4													D
E	−5	0	0	−1	0	0	1	3	4												E
Q	−5	−1	−1	0	0	−1	1	2	2	4											Q
H	−3	−1	−1	0	−1	−2	2	1	1	3	6										H
R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	6									R
K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5								K
M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6							M
I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5						I
L	−6	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	6					L
V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4				V
F	−4	−3	−3	−5	−4	−5	−4	−6	−5	−5	−2	−4	−5	0	1	2	−1	9			F
Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10		Y
W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Figure 3.14. The log odds form (the mutation data matrix or MDM) of the PAM250 scoring matrix. Amino acids are grouped according to the chemistry of the side group: (C) sulfhydryl, (STPAG) small hydrophilic, (NDEQ) acid, acid amide and hydrophilic, (HRK) basic, (MILV) small hydrophobic, and (FYW) aromatic. Each matrix value is calculated from an odds score, the probability that the amino acid pair will be found in alignments of homologous proteins divided by the probability that the pair will be found in alignments of unrelated proteins by random chance. The logarithm of these ODDS scores to the base 10 is multiplied by 10 and then used as the table value (see text for details). Thus, +10 means the ancestor probability is greater, 0 that the probabilities are equal, and −4 that the alignment is more often a chance one than due to an ancestor relationship. Because these numbers are logarithms, they may be added to give a combined probability of two or more amino acid pairs in an alignment. Thus, the probability of aligning two Ys in an alignment YY/YY is 10 + 10 = 20, a very significant score, whereas that of YY with TP is −2 −5 = − 7, a rare and unexpected alignment between homologous sequences.

At one time, the PAM250 scoring matrix was modified in an attempt to improve the alignment obtained. All scores for matching a particular amino acid were normalized to the same mean and standard deviation, and all amino acid identities were given the same score to provide an equal contribution for each amino acid in a sequence alignment (Gribskov and Burgess 1986). These modifications were included as the default matrices for the GCG sequence alignment programs in versions 8 and earlier and are optional in later versions. They are not recommended because they will not give an optimal alignment that is in accord with the evolutionary model.

Choosing the Best PAM Scoring Matrices for Detecting Sequence Similarity. The ability of PAM scoring matrices to distinguish statistically between chance and biologically meaningful alignments has been analyzed using a recently developed statistical theory for sequences (Altschul 1991) that is discussed later in this chapter. As discussed above, each PAM matrix is designed to score alignments between sequences that have diverged by a particular degree of evolutionary distance. Altschul (1991) has examined how well the PAM matrices actually can distinguish proteins that have diverged to a greater or lesser extent, when these proteins are subjected to a local alignment.

Initially, when using a scoring matrix to produce an alignment, the amount of similarity between sequences may not be known. However, the ungapped alignment scores obtained are maximal when the correct PAM matrix, i.e., the one corresponding to the degree of similarity in the target sequences, is used (Altschul 1991). Altschul (1991) has also examined the ability of PAM matrices to provide a reliable enough indication of an ungapped local alignment score between sequences on an initial attempt of alignment. For sequence alignments, the PAM200 matrix is able to detect a significant ungapped alignment of 16–62 amino acids whose score is within 87% of the optimal one. Alternatively, several combinations, such as PAM80 and PAM250 or PAM120 and PAM350, can also be used. Altschul (1993) has also proposed using a single matrix and adjusting a statistical parameter in the scoring system to reach more distantly related sequences, but this change would primarily be for database searches.

Scoring matrices are also used in database searches for similar sequences. The optimal matrices for these searches have also been determined (see book Web site and Chapter 7). It is important to remember that these predictions assume that the amino acid distributions in the set of protein families used to make the scoring matrix are representative of all families that are likely to be encountered. The original PAM matrices represent only a small number of families. Scoring matrices obtained more recently, such as the BLOSUM matrices, are based on a much larger number of protein families. BLOSUM matrices are not based on a PAM evolutionary model in which changes at large evolutionary distance are predicted by extrapolation of changes found at small distances. Matrix values are based on the observed frequency of change in a large set of diverse proteins. As is discussed on the book Web site, the BLOSUM scoring matrices (especially BLOSUM62) appear to capture more of the distant types of variations found in protein families.

In addition to the aforementioned differences among PAM scoring matrices for scoring alignments of more- or less-related proteins, the ability of each PAM matrix to discriminate real local alignments from chance alignments also varies. To calculate the ability of the entire matrix to discriminate related from unrelated sequences (H, the relative entropy), the score for each amino acid pair s_{ij} (in units of $\log_2$, called bits) is multiplied by the probability of occurrence of that pair in the original dataset, q_{ij} (Altschul 1991). This weighted score is then summed over all of the amino acid pairs to produce a score that represents the ability of the average amino acid pair in the matrix to discriminate actual from chance alignments.

$$H = \sum_{i=1}^{20} \sum_{j=1}^{i} q_{ij} \times s_{ij} \qquad (3)$$

In information theory, this score is called the average mutual information content per pair, and the sum over all pairs is the relative entropy of the matrix (termed H). The relative entropy will be a small positive number. For the PAM250 matrix the number is +0.36, for PAM120, +0.98, and for PAM160, +0.70. In general, all other factors being equal, the higher the value of H for a scoring matrix, the more likely it is to be able to distinguish real from chance alignments.

Analysis of the Dayhoff Model of Protein Evolution as Used in PAM Matrices. As outlined above, the Dayhoff model of protein evolution is a Markov process. In this model, each amino acid site in a protein can change at any time to any of the other 20 amino acids with probabilities given by the PAM table, and the changes that occur at each site are independent of the amino acids found at other sites in the protein and depend only on the cur-

rent amino acid at the site. The assumptions that underlie the method of constructing the Dayhoff scoring matrix have been challenged (for discussion, see George et al. 1990; States and Boguski 1991). First, it is assumed that each amino acid position is equally mutable, whereas, in fact, sites vary considerably in their degree of mutability. Mutagenesis hot spots are well known in molecular genetics, and variations in mutability of different amino acid sites in proteins are well known.

The more conserved amino acids in similar proteins from different species are ones that play an essential role in structure and function and the less conserved are in sites that can vary without having a significant effect on function. Thus, there are many factors that influence both the location and types of amino acid changes that occur in proteins. Wilbur (1985) has tested the Markov model of evolution (see box, below) and has shown that it can be valid if certain changes are made in the way that the PAM matrices are calculated.

Test of Markov Model of Evolution in Proteins

To test the model, Wilbur addressed a major criticism of the PAM scoring matrix, namely that the frequency of amino acid changes that require two nucleotide changes is higher than would be expected by chance. About 20% of the observed amino acid changes require more than a single mutation for the necessary codon changes. This fraction is far greater than would be expected by chance.

To correct for changes that require at least two mutations, Wilbur recalculated the PAM1 matrix using only amino acid substitution data from 150 amino acid pairs that are accountable by single mutations. To accomplish this calculation, he used a refined mathematical model that provided a more precise measure of the rate of substitution. He then estimated frequencies of the other 230 amino acid substitutions reachable only by at least two mutations, and compared these frequencies to the values calculated by Dayhoff, who had assumed these were single-step changes. If these numbers agreed, argued Wilbur, then the PAM model used to produce the Dayhoff matrix is a reliable one. In fact, the Dayhoff values exceeded the two-step model values by a factor of about 117. One source of discrepancy was the assumption that the two-step changes were a linear function of evolutionary time over short evolutionary periods of 1 PAM (average time of 1 PAM = 10 my), whereas, because two mutations are required to make the change, a quadratic function is expected. With this correction made to the Dayhoff calculations for amino acid substitutions requiring two mutations, agreement with the two-step model improved about 10-fold, leaving another 11.7-fold unaccounted for.

Wilbur analyzed the remainder by the covarion hypothesis (Fitch and Markowitz 1970; Miyamoto and Fitch 1995), in which it is assumed that only a certain fraction of amino acid sites in a protein are variable and that one site influences another. Thus, a change in one site may influence the variability of others. This model seems to be reasonable from many biological perspectives. The prediction of this hypothesis is that the frequency of two-step changes would be overestimated because we did not take into account the failure of many sites to be mutable. Using a reasonable estimate of 0.3 for the fraction of the sites that could change, the effect on the Dayhoff calculations for frequencies of two-step changes would be 3.3-fold. The remaining discrepancy in the 11.7-fold ratio between Dayhoff values and two-step values may be attributable to variations in mutation rates from site to site, or to the exclusion of certain amino acids at a particular site. In conclusion, Wilbur (1985) has shown that the Dayhoff model for protein evolution appears to give predictable and consistent

results, but that frequencies of change between amino acids that require two mutational steps must be calculated as a two-step process. Failure to do so generates errors due to variations in site-to-site mutability. George et al. (1990) have counterargued that it has never been demonstrated that two independent mutations must occur, each becoming established in a population before the next appears.

A further criticism of the PAM scoring matrices is that they are not more useful for sequence alignment than simpler matrices, such as one based on a chemical grouping of amino acid side chains. Although alignment of related proteins is straightforward and quite independent of the symbol comparison scoring scheme, alignments of less-related proteins are much more speculative (Feng et al. 1985). These matrices and the BLOSUM matrices have been very useful for finding more distantly related sequences (George et al. 1990). There have been recent changes in the way that members of protein families are identified (see Chapters 4 and 9). Once a family has been identified, family-specific scoring matrices can be produced, and there is no point in using these general matrices. As described in Chapter 4, a scoring matrix representing a section of aligned sequences with no gaps, or a matrix representing a section of aligned sequences with matches, mismatches, and gaps (a profile), are the best tools to search for more family members.

Another criticism of the PAM matrix is that constructing phylogenetic relationships prior to scoring mutations has limitations, due to the difficulty of determining ancestral relationships among sequences, a topic discussed in Chapter 6. Early on in the Dayhoff analysis, the evolutionary trees were estimated by a voting scheme for the branches in the tree, each node being estimated by the most abundant amino acid in distal parts of the tree. Once available, the PAM matrices were used to estimate the evolutionary distance between proteins, given the amount of sequence similarity. Such data can be used to produce a tree based on evolutionary distances (Chapter 6). This circular analysis of using alignments to score amino acid changes and then to use the matrices to produce new alignments has also been criticized. However, no method has yet been devised in any type of sequence analysis for completely circumventing this problem. Evidence that the values in the scoring matrix are insensitive to changes in the phylogenetic relationships has been provided (George et al. 1990).

Finally, the Dayhoff PAM matrices have been criticized because they are based on a small set of closely related proteins. The Dayhoff data set has been augmented to include the 1991 protein database (Gonnet et al. 1992; Jones et al. 1992). The ability of the Dayhoff matrices to identify homologous sequences has also been extensively compared to that of other scoring matrices. These comparisons are discussed on the book Web site.

Blocks Amino Acid Substitution Matrices (BLOSUM)

The BLOSUM62 substitution matrix (Henikoff and Henikoff 1992) is widely used for scoring protein sequence alignments. The matrix values are based on the observed amino acid substitutions in a large set of ~2000 conserved amino acid patterns, called blocks. These blocks have been found in a database of protein sequences representing more than 500 families of related proteins (Henikoff and Henikoff 1992) and act as signatures of these protein families. The BLOSUM matrices are thus based on an entirely different type of sequence analysis and a much larger data set than the Dayhoff PAM matrices.

These protein families were originally identified by Bairoch in the Prosite catalog. This catalog provides lists of proteins that are in the same family because they have a similar biochemical function. For each family, a pattern of amino acids that are characteristic of that function is provided. Henikoff and Henikoff (1991) examined each Prosite family for the presence of ungapped amino acid patterns (blocks) that were present in each family and that could be used to identify members of that family. To locate these patterns, the sequences of each protein family were searched for similar amino acid patterns by the MOTIF program of H. Smith (Smith et al. 1990), which can find patterns of the type aa1 d1 aa2 d2 aa3, where aa1 and aa2 are conserved amino acids and d1 and d2 are stretches of intervening sequence up to 24 amino acids long located in all sequences. These initial patterns were organized into larger ungapped patterns (blocks) between 3 and 60 amino acids long by the Henikoffs' PROTOMAT program (http://www.blocks.fhcrc.org). Because these blocks were present in all of the sequences in each family, they could be used to identify other members of the same family. Thus, the family collections were enlarged by searching the sequence databases for more proteins with these same conserved blocks.

The blocks that characterized each family provided a type of multiple sequence alignment for that family. The amino acid changes that were observed in each column of the alignment could then be counted. The types of substitutions were then scored for all aligned patterns in the database and used to prepare a scoring matrix, the BLOSUM matrix, indicating the frequency of each type of substitution. As previously described for the PAM matrices, BLOSUM matrix values were given as logarithms of odds scores of the ratio of the observed frequency of amino acid substitutions divided by the frequency expected by chance. An example of the calculations is shown in Figure 3.15.

This procedure of counting all of the amino acid changes in the blocks, however, can lead to an overrepresentation of amino acid substitutions that occur in the most closely related members of each family. To reduce this dominant contribution from the most alike sequences, these sequences were grouped together into one sequence before scoring the amino acid substitutions in the aligned blocks. The amino acid changes within these clustered sequences were then averaged. Patterns that were 60% identical were grouped together to make one substitution matrix called BLOSUM60, and those 80% alike to make another matrix called BLOSUM80, and so on. As with the PAM matrices, these matrices differ in the degree to which the more common amino acid pairs are scored relative to the less common pairs. Thus, when used for aligning protein sequences, they provide a greater or lesser distinction between the more common and less common amino acid pairs. The ability of these different BLOSUM matrices to distinguish real from chance alignments and to identify as many members as possible of a protein family has been determined (Henikoff and Henikoff 1992).

Two types of analyses were performed: (1) an information content analysis of each matrix, as was described above for the PAM matrices, and (2) an actual comparison of the ability of each matrix to find members of the same families in a database search, discussed below. As the clustering percentage was increased, the ability of the resulting matrix to distinguish actual from chance alignments, defined as the relative entropy of the matrix or the average information content per residue pair (see above), also increased. As clustering increased from 45% to 62%, the information content per residue increased from ~0.4 to 0.7 bits per residue, and was ~1.0 bits at 80% clustering. However, at the same time, the number of blocks that contributed information decreased by 25% between no clustering and 62% clustering. BLOSUM62 represents a balance between information content and data size. The BLOSUM62 matrix is shown in Figure 3.16.

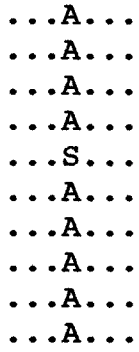


Figure 3.15. Derivation of the matrix values in the BLOSUM62 scoring matrix. As an example of the calculations, if a column in one of the blocks consisted of 9 A and 1 S amino acids, the following is true for this data set (see Henikoff and Henikoff 1992).

1. Since the original sequence from which the others were derived is not known, each column position has to be considered a possible ancestor of the other nine columns. Hence, there are $8+7+6 \ldots +1 = 36$ possible AA pairs (f_{AA}) and 9 possible AS pairs (f_{AS}) to be compared.
2. There are $20+19+18+ \ldots +1 = 210$ possible amino acid pairs.
3. The frequency of occurrence of an AA pair, $q_{AA} = f_{AA}/(f_{AA} + f_{AS}) = 36/(36+9) = 0.8$, and that of an AS pair, $q_{AS} = f_{AS}/(f_{AA} + f_{AS}) = 9/(36+9) = 0.2$.
4. The expected frequency of A being in a pair, $p_A = (q_{AA} + q_{AS}/2) = 0.8 + 0.2/2 = 0.9$, and that of $p_S = q_{AS}/2 = 0.1$.
5. The expected frequency of occurrence of AA pairs, $e_{AA} = p_A \times p_A = 0.9 \times 0.9 = 0.81$, and that of AS, $e_{AS} = 2 \times p_S \times p_A = 2 \times 0.9 \times 0.1 = 0.18$.
6. The matrix entry for AA will be calculated from the ratio of the occurrence frequency to the expected frequency. For AA, ratio $= q_{AA}/ e_{AA} = 0.8/0.81 = 0.99$, and for AS, ratio $= q_{AS}/ e_{AS} = 0.2/0.18 = 1.11$.
7. Both ratios are converted to logarithms to the base 2 and then multiplied by 2 (1/2 bit units). Matrix entry for AA, $s_{AA} = \log_2(q_{AA}/ e_{AA}) = -0.04$, and for AS, $s_{AS} = \log_2(q_{AS}/ e_{AS}) = 0.30$. These logarithms are both rounded to $1\ ^1/_2$ bit unit.

Henikoff and Henikoff (1993) have prepared a set of interval BLOSUM matrices that represent the changes observed between more closely related or more distantly related representatives of each block. Rather than representing the changes observed in very alike sequences up to sequences that were *n*% alike to give a BLOSUM-*n* matrix, the new BLOSUM-*nm* matrix represented the changes observed in sequences that were between *n*% alike and *m*% alike. The idea behind these matrices was to have a set of matrices corresponding to amino acid changes in sequence blocks that are separated by different evolutionary distances.

Comparison of the PAM and BLOSUM Amino Acid Substitution Matrices

There are several important differences in the ways that the PAM and BLOSUM scoring matrices were derived, and these differences should be appreciated in order to interpret the results of protein sequence alignments obtained with these matrices. First, the PAM matrices are based on a mutational model of evolution that assumes amino acid changes occur as a Markov process, each amino acid change at a site being independent of previous changes at that site. Changes are scored in sequences that are 85% similar after predicting

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	−1	4																			S
T	−1	1	5																		T
P	−3	−1	−1	7																	P
A	0	1	0	−1	4																A
G	−3	0	−2	−2	0	6															G
N	−3	1	0	−2	−2	0	6														N
D	−3	0	−1	−1	−2	−1	1	6													D
E	−4	0	−1	−1	−1	−2	0	2	5												E
Q	−3	0	−1	−1	−1	−2	0	0	2	5											Q
H	−3	−1	−2	−2	−2	−2	1	−1	0	0	8										H
R	−3	−1	−1	−2	−1	−2	0	−2	0	1	0	5									R
K	−3	0	−1	−1	−1	−2	0	−1	1	1	−1	2	5								K
M	−1	−1	−1	−2	−1	−3	−2	−3	−2	0	−2	−1	−1	5							M
I	−1	−2	−1	−3	−1	−4	−3	−3	−3	−3	−3	−3	−3	1	4						I
L	−1	−2	−1	−3	−1	−4	−3	−4	−3	−2	−3	−2	−2	2	2	4					L
V	−1	−2	0	−2	0	−3	−3	−3	−2	−2	−3	−3	−2	1	3	1	4				V
F	−2	−2	−2	−4	−2	−3	−3	−3	−3	−3	−1	−3	−3	0	0	0	−1	6			F
Y	−2	−2	−2	−3	−2	−3	−2	−3	−2	−1	2	−2	−2	−1	−1	−1	−1	3	7		Y
W	−2	−3	−2	−4	−3	−2	−4	−4	−3	−2	−2	−3	−3	−1	−3	−2	−3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Figure 3.16. The BLOSUM62 amino acid substitution matrix. The amino acids in the table are grouped according to the chemistry of the side group: (C) sulfhydryl, (STPAG) small hydrophilic, (NDEQ) acid, acid amide, and hydrophilic, (HRK) basic, (MILV) small hydrophobic, and (FYW) aromatic. Each entry is the logarithm of the odds score, found by dividing the frequency of occurrence of the amino acid pair in the BLOCKS database (after sequences 62% or more in similarity have been clustered) by the likelihood of an alignment of the amino acids by random chance. The denominator in this ratio is calculated from the frequency of occurrence of each of the two individual amino acids in the BLOCKS database and provides a measure of a chance alignment of the two amino acids. The actual/expected ratio is expressed as a log odds score in so-called half-bit units, obtained by converting the odds ratio to a logarithm to the base 2, and then multiplying by 2. A zero score means that the frequency of the amino acid pair in the database is as expected by chance, a positive score that the pair is found more often than by chance, and a negative score that the pair is found less often than by chance. The accumulated score of an alignment of several amino acids in two sequences may be obtained by adding up the respective scores of each individual pair of amino acids. As with the PAM250-derived matrix, the highest-scoring matches are between amino acids that are in the same chemical group, and the very highest-scoring matches are for cysteine–cysteine matches and for matches among the aromatic amino acids. Compared to the PAM160 matrix, however, the BLOSUM62 matrix gives a more positive score to mismatches with the rare amino acids, e.g., cysteine, a more positive score to mismatches with hydrophobic amino acids, but a more negative score to mismatches with hydrophilic amino acids (Henikoff and Henikoff 1992).

a phylogenetic history of the changes in each family. Thus, the PAM matrices are based on prediction of the first changes that occur as proteins diverge from a common ancestor during evolution of a protein family. Matrices that may be used to compare more distantly related proteins are then derived by extrapolation from these short-term changes, assuming that these more distant changes are a reflection of the short-term changes occurring over and over again. For each longer evolutionary interval, each amino acid can change to any other with the same frequency as observed in the short term. In contrast, the BLOSUM matrices are not based on an explicit evolutionary model. They are derived from considering all amino acid changes observed in an aligned region from a related family of proteins, regardless of the overall degree of similarity between the protein sequences. However, these

proteins are known to be related biochemically and, hence, should share common ancestry. The evolutionary model implied in such a scheme is that the proteins in each family share a common origin, but closer versus distal relationships are ignored, as if they all were derived equally from the same ancestor, called a starburst model of protein evolution (see Chapter 6). Second, the PAM matrices are based on scoring all amino acid positions in related sequences, whereas the BLOSUM matrices are based on substitutions and conserved positions in blocks, which represent the most alike common regions in related sequences. Thus, the PAM model is designed to track the evolutionary origins of proteins, whereas the BLOSUM model is designed to find their conserved domains.

Other Amino Acid Scoring Matrices

In addition to the Dayhoff PAM, and related Gonnet et al. (1992), Benner et al. (1994), and Jones et al. (1992) matrices and the BLOSUM matrices, a number of other amino acid substitution matrices have been used for producing protein sequence alignments, and several representative ones are listed in Table 3.3. For a more complete list and comparison, see Vogt et al. (1995). These tables vary from a comparison of simple chemical properties of amino acids to a complex analysis of the substitutions found in secondary structural domains of proteins. Because most of these tables are designed to align proteins on the basis of some such feature of the amino acids, and not on an evolutionary model, they are not particularly suitable for evolutionary analysis. They can be very useful, however, for discovering structural and functional relationships, or family relationships among proteins. A sequence alignment program that uses a combination of these tables has been found to be particularly useful for detecting distant protein relationships (Argos 1987; Rechid et al. 1989). There have been extensive comparisons of the usefulness of various amino acid substitution matrices for aligning sequences, for finding similar sequences in a protein sequence database, or for aligning similar sequences based on structure that are described on the book Web site.

Table 3.3. *Criteria used in amino acid scoring matrices for sequence alignments*

1. Simple identity, which scores only identical amino acids as a match and all others as a mismatch.
2. Genetic code changes, which score the minimum number of nucleotide changes to change a codon for one amino acid into a codon for another, due to Fitch (1966), and also with added information based on structural similarity of amino acid side chains (Feng et al. 1985). A similar matrix based on the assumption that genetic code is the only factor influencing amino acid substitutions has been produced (Benner et al. 1994).
3. Matrices based on chemical similarity of amino acid side chains, molecular volume, and polarity and hydrophobicity of amino acid side chains (see Vogt et al. 1995).
4. Amino acid substitutions in structurally aligned three-dimensional structures (Risler et al. 1988; matrix JO93, Johnson and Overington 1993). A similar matrix was described by Henikoff and Henikoff (1993). Sander and Schneider (1991) prepared a similar matrix based on these same substitutions but augmented by substitutions found in proteins which are so similar to the structure-solved group that they undoubtedly have the same three-dimensional structure.
5. Gonnet et al. (1994) have prepared a 400 × 400 dipeptide substitution matrix for aligning proteins based on the possibility that amino acid substitutions at a particular site are influenced by neighboring amino acids, and thus that the environment of an amino acid plays a role in protein evolution.
6. Jones et al. (1994) have prepared a scoring matrix specifically for transmembrane proteins. This matrix was prepared using an analysis similar to that used for preparing the original Dayhoff PAM matrices, and therefore provides an estimate of evolutionary distances among members of this class of proteins.

Nucleic Acid PAM Scoring Matrices

Just as amino acid scoring matrices have been used to score protein sequence alignments, nucleotide scoring matrices for scoring DNA sequence alignments have also been developed. The DNA matrix can incorporate ambiguous DNA symbols (see Table 2.1) and information from mutational analysis, which reveals that transitions (substitutions between the purines A and G or between the pyrimidines C and T) are more probable than transversions (substitutions between purine to pyrimidine or pyrimidine to purine) (Li and Graur 1991). These substitution matrices may be used to produce global or local alignments of DNA sequences.

States et al. (1991) have developed a series of nucleic acid PAM matrices based on a Markov transition model similar to that used to generate the Dayhoff PAM scoring matrices. Although designed to improve the sensitivity of similarity searches of sequence databases, these matrices also may be used to score nucleic acid alignments. The advantage of using these matrices is that they are based on a defined evolutionary model and that the statistical significance of alignment scores obtained by local alignment programs may be evaluated, as described later in this chapter.

To prepare these DNA PAM matrices, a PAM1 mutation matrix representing 99% sequence conservation and one PAM of evolutionary distance (1% mutations) was first calculated. For a model in which all mutations from any nucleotide to any other are equally likely, and in which the four nucleotides are present at equal frequencies, the four diagonal elements of the PAM1 matrix representing no change are 0.99 whereas the six other elements representing change are 0.00333 (Table 3.4). The values are chosen so that the sum of all possible changes for a given nucleotide in the PAM1 matrix is 1% ($3 \times 0.00333 = 0.00999$). For a biased mutation model in which a given transition is threefold more likely than a transversion (Table 3.4), the off-diagonal matrix elements corresponding to the one possible transition for each nucleotide are 0.006 and those for the two possible transversions are 0.002, and the sum for each nucleotide is again 1% ($0.006 + 0.002 + 0.002 = 0.01$).

As with the amino acid matrices, the above matrix values are then used to produce log odds scoring matrices that represent the frequency of substitutions expected at increasing

Table 3.4. *Nucleotide mutation matrix for an evolutionary distance of 1 PAM, which corresponds to a probability of a change at each nucleotide position of 1%*

A. Model of uniform mutation rates among nucleotides

	A	G	T	C
A	0.99			
G	0.00333	0.99		
T	0.00333	0.00333	0.99	
C	0.00333	0.00333	0.00333	0.99

B. Model of threefold higher transitions than transversions

	A	G	T	C
A	0.99			
G	0.006	0.99		
T	0.002	0.002	0.99	
C	0.002	0.002	0.006	0.99

Values are frequency of change at each site, or of no change for all base combinations.

evolutionary distances. In terms of an alignment, the probability (s_{ij}) of obtaining a match between nucleotides i and j, divided by the random probability of aligning i and j, is given by

$$s_{ij} = \log (p_i M_{ij} / p_i p_j) \tag{4}$$

where M_{ij} is the value in the mutation matrix given in Table 3.4, and p_i and p_j are the fractional composition of each nucleotide, assumed to be 0.25. The base of the logarithm can be any value, corresponding to multiplying every value in the matrix by the same constant. With such scaling variations, the ability of the matrix to distinguish among significant and chance alignments will not be altered. The resulting tables with s_{ij} expressed in units of bits (logarithm to the base 2) and rounded off to the nearest whole integer are shown in Table 3.5.

From these PAM1 matrices, additional log odds matrices at an evolutionary distance of n PAMs may be obtained by multiplying the PAM1 matrix by itself n times. The ability of each matrix to distinguish real from random nucleotide matches in an alignment, designated H, measured in bit units ($\log_2$) can be calculated using the equation

$$H = \sum_{i,j} p_i p_j s_{ij} 2^{s_{ij}} \tag{5}$$

where the s_{ij} scores are also expressed in bit units. In Table 3.6 are shown the log odds values of the match and mismatch scores for PAM matrices at increasing evolutionary distances, assuming a uniform rate of mutation among all nucleotides. Also shown is the percentage of nucleotides that will be changed at that distance. The identity score will be 100 minus this value. This percentage is not as great as the PAM score due to expected back-mutation over longer time periods. Also shown are the H scores of the matrices at each PAM value.

Table 3.5. *Nucleotide substitution matrix at 1 PAM of evolutionary distance*

A. Model of uniform mutation rates among nucleotides

	A	G	T	C
A	2			
G	−6	2		
T	−6	−6	2	
C	−6	−6	−6	2

B. Model of threefold higher transitions than transversions

	A	G	T	C
A	2			
G	−5	2		
T	−7	−7	2	
C	−7	−7	−5	2

Units are log odds scores obtained as described in the text.

Table 3.6. *Properties of nucleic acid substitution matrices assuming a uniform rate of mutation among nucleotides*

PAM distance	Percentage difference	Match score (bits)	Mismatch score (bits)	Average information per position (bits)
10	9.4	1.86	−3.00	1.40
25	21.3	1.66	−1.82	0.92
50	36.5	1.34	−1.04	0.47
100	55.2	0.84	−0.44	0.13
125	60.8	0.65	−0.30	0.07

The following points may be made:

1. If comparing sequences that are quite similar, it is better to use a lower scoring matrix because the information content of the small PAM matrices is relatively higher. As discussed earlier for lower-numbered Dayhoff PAM matrices for more-alike protein sequences, a more optimal alignment will be obtained.
2. As the PAM distance increases, the mismatch scores in the biased mutational model in Table 3.7 become positive and appear as conservative substitutions. Thus, the bias model can provide considerably more information than the uniform mutation model when aligning sequences that are distantly related (>30% different) and may be used for this purpose (States et al. 1991).
3. The scoring matrices at large evolutionary distances provide very little information per aligned nucleotide pair. When sequences have so little similarity, a much longer alignment is necessary to be significant.

As with amino acid scoring matrices, the average information content shown is only achieved by using the scoring matrix that matches the percentage difference between the sequences. For example, for sequences that are 21% different (79% identical), the matrix at 25 PAM distance should be used. One cannot know ahead of time what the percentage similarity or difference between two sequences actually is until an alignment is done, thus a trial alignment must first be done. States et al. (1991) have calculated how efficient a given scoring matrix is at achieving the highest possible score in aligning two sequences that vary in their levels of similarity. Once the initial similarity score has been obtained with these matrices, a more representative score can be obtained by using another PAM matrix designed specifically for sequences at that level of similarity.

Gap Penalties

The inclusion of gaps and gap penalties is necessary in order to obtain the best possible alignment between two sequences. A gap opening penalty for any gap (g) and a gap exten-

Table 3.7. *Properties of nucleic acid substitution matrices assuming transitions are threefold more frequent than transversions*

PAM distance	Percentage difference	Match score (bits)	Transition score (bits)	Transversion score (bits)	Average information per position (bits)
10	9.3	1.86	−2.19	−3.70	1.42
25	21.0	1.66	−1.06	−2.46	0.96
50	35.8	1.36	−0.37	−1.60	0.54
100	53.7	0.89	0.06	−0.86	0.19
150	62.9	0.57	0.16	−0.52	0.08

sion penalty for each element in the gap (r) is most often used, to give a total gap score w_x, according to the equation

$$w_x = g + rx \tag{6}$$

where x is the length of the gap. Note that in some formulations of the gap penalty, the equation $w_x = g + r\,(x - 1)$ is used. Thus, the gap extension penalty is not added to the gap opening penalty until the gap size is 2. Although this difference does not affect the alignment obtained, one needs to distinguish which method is being used by a particular computer program if the correct results are to be obtained. In the former case, the penalty for a gap of size 1 is $g + x$, whereas in the latter case this value is g. The values for these penalties have to be chosen to balance the scores in the scoring matrix that is used. Thus, the Dayhoff log odds matrix at PAM250 is expressed in units of $\log_{10}$, which is approximately 1/3 bits, but if this matrix were converted to 1/2 bits, the same gap penalties would no longer be appropriate.

If too high a gap penalty is used relative to the range of scores in the substitution matrix, gaps will never appear in the alignment. Conversely, if the gap penalty is too low compared to the matrix scores, gaps will appear everywhere in the alignment in order to align as many of the same characters as possible. Fortunately, most alignment programs will suggest gap penalties that are appropriate for a given scoring matrix in most situations. In the GCG and FASTA program suites, the scoring matrix itself is formatted in a way that includes default gap penalties. Examples of the values of g and r used by various alignment programs are shown on the book Web site. When deciding gap penalties for local alignment programs, another consideration is that the penalties should be large enough to provide a local alignment of the sequences. Examples of suitable values are given in Table 3.10 on p. 114. Altschul and Gish (1996) and Pearson (1996, 1998) have found that use of appropriate gap penalties will provide an improved local alignment based on statistical analysis. These studies are described in detail in the following section.

Mathematician Peter Sellers (1974) showed that if sequence alignment was formulated in terms of distances instead of similarity between sequences, a biologically more appealing interpretation of gaps is possible. The distance is the number of changes that must be made to convert one sequence into the other and represents the number of mutations that will have occurred following separation of the genes during evolution; the greater the distance, the more distantly related are the sequences in evolution. In this case, substitution produces a positive score of 1. Notice that the distance score plus the similarity score for an alignment is equal to 1. Sellers proved that this distance formulation of sequence alignment has a desirable mathematical property that also makes evolutionary sense. If three sequences, **a**, **b**, and **c**, are compared using the above scoring scheme, the distance score as defined above is described as a metric that satisfies the triangle inequality relationship

$$d(\mathbf{a},\mathbf{b}) + d(\mathbf{b},\mathbf{c}) \geq d(\mathbf{a},\mathbf{c}) \tag{7}$$

where $d(\mathbf{a},\mathbf{b})$ is the distance between sequences **a** and **b**, and likewise for the other two d values. Expressed another way, if the three possible distances between three sequences are obtained, then the distance between any first pair plus that for any second pair cannot underscore the third pair. Violating this rule would not be consistent with the expected evolutionary origin of the sequences. To satisfy the metric requirement, the scoring of individual matches, mismatches, and gaps must be such that in an alignment of two iden-

tical sequences **a** and **a**′, $d(\mathbf{a},\mathbf{a}')$ must equal 0 and for two totally different sequences **b** and **b**′, $d(\mathbf{b},\mathbf{b}')$ must equal 1. For any other two sequences **a** and **b**, $d(\mathbf{a},\mathbf{b}) = d(\mathbf{b},\mathbf{a})$. Hence, it is important that the distance score for changing one sequence character into a second is the same as the converse score for changing the second into the first, if the distance score of the alignment is to remain a metric and to make evolutionary sense. The above relationships were shown by Sellers to be true for gaps of length 1 in a sequence alignment. He also showed that the smallest number of steps required to change one sequence into the other could be calculated by the dynamic programming algorithm. The method was similar to that discussed above for the Needleman-Wunsch global and Smith-Waterman local alignments, except that these former methods found the maximum similarity between two sequences, as opposed to the minimum distance found by the Sellers analysis.

Subsequently, Smith et al. (1981) and Smith and Waterman (1981a,b) showed that gaps of any length could also be included in an alignment and still provide a distance metric for the alignment score. In this formulation, the gap penalty was required to increase as a function of the gap length. The argument was made that a single mutational event involving a single gap of n residues should be more likely to have occurred than n single gaps. Thus, to increase the likelihood of such gaps of length >1 being found, the penalty for a gap of length n was made smaller than the score for n individual gaps. The simplest way of implementing this feature of the gap penalty was to have the gap score w_x be a linear function of gap length by consisting of two parts, a larger gap opening penalty (g) and a smaller gap extension penalty (r) for each extra position in the gap, or $w_x = g + rx$, where x is the length of the gap, as described above. This type of gap penalty is referred to as an affine gap penalty in the literature. Any other formula for scoring gap penalties should also work, provided that the score increases with length of the gap but that the score is less than x individual gaps. Scoring of gaps by the above linear function of gap length has now become widely used in sequence alignment. However, more complex gap penalty functions have been used (Miller and Myers 1988).

Penalties for Gaps at the Ends of Alignments

Sequence alignments are often produced that include gaps opposite nonmatching characters at the ends of an alignment. These gaps may be given the same penalty score as gaps inside of the alignment or, alternatively, they may not be given any penalty score. End gaps were an important component in the mathematical formulation of both the similarity and distance methods of sequence alignment for producing both global and local alignments. Failure to include them in distance calculations can result in a failure to obtain distance scores that make evolutionary sense (Smith et al. 1981). Examples of using or of not using end gap penalties in the Needleman-Wunsch alignment are shown on the book Web site. Without scoring end alignments, gaps may be liberally placed at the ends of alignments by the dynamic programming algorithm to increase the matching of internal characters, as opposed to including these gaps as a part of the overall alignment.

If comparing sequences that are homologous and of about the same length, it makes a great deal of sense to include end gap penalties to achieve the best overall alignment. For sequences that are of unknown homology or of different lengths, it may be better to use an alignment that does not include end gap penalties (States and Boguski 1991). If one sequence is expected to be contained within the other, it is reasonable to include end gap penalties only for the shorter sequence. However, for any test alignment, these end penalties should be included in at least one alignment to assure that they do not have an effect. It is also important to use alignment programs that include them as an option.

Parametric Sequence Alignments

Computer methods that find a range of possible alignments in response to varying the scoring system used for matches, mismatches, and gaps, called parametric sequence comparisons (Waterman et al. 1992; Waterman 1994 and references therein), have been developed. There is also an effort to use scores such that the results of global and local types of sequence alignments provide consistent results. For example, if two sequences are similar along their entire lengths, both global and local methods should provide the same alignment. The program Xparal (Gusfield and Stelling 1996), which can perform this type of analysis, is available from http://theory.cs.ucdavis.edu/~stevenk. The program runs on a UNIX environment under X-Windows. When provided with two sequences and some of the alignment parameters, such as gap score, the program displays graphically the types of possible alignments when the remaining parameters are varied. Another sequence alignment program that performs parametric sequence alignment is the Bayes block aligner, discussed below (p. 124).

Effects of Varying Mismatched Gap Penalties on Local Alignment Scores

Vingron and Waterman (1994) have reviewed the effect of varying the parameters of the scoring system on the alignment of random DNA and protein sequences. To simplify the number of parameters, a constant penalty for any size gap was used. If a very high mismatch penalty is used relative to a positive score for a match, with zero gap penalty, the local alignment of these sequences will not include any gaps and is defined as the longest common subsequence. The global alignment with the same scoring parameters will have no mismatches but will have many gaps so placed as to maximize the matches, and the score will be positive. In this case, the score of the local alignment of the sequences is predicted to increase linearly with the length of the sequences being compared.

Another case of varying alignment is penalizing gaps heavily. Then the best scoring local alignment between the sequences will be one that optimizes the score between matches and mismatches, without any gaps. If both mismatches and gaps are heavily penalized, the resulting alignment will also be a local alignment that contains the longest region of exact matches. In the above two cases, the alignment score of the highest-scoring local alignment will increase as the logarithm of the length of the sequences. Under these same conditions, the score of the corresponding global alignment between the sequences will be negative. The transition between a linear and logarithmic dependence of the local similarity score on sequence length occurs when the score of the corresponding global alignment is zero. When both the mismatch and gap penalties are varied between zero and a high negative score, the number of possible alignments of random DNA sequences is very large.

Three general conclusions can be drawn from this theoretical study of random sequence alignments: (1) Use of high mismatch and gap penalties that are greater than a match score will find local alignments, of which there are relatively few in number; (2) when the penalty for a mismatch is greater than twice the score for a match, the gap penalty becomes the decisive parameter in the alignment; and (3) for a mismatch penalty less than twice the score of a gap and a wide range of gap penalties, there are a large number of possible alignments that depend on both the mismatch and gap penalty scores.

Distinguishing local from global alignments has an important practical application. A local alignment is rarely produced between random sequences. Accordingly, the significance of a local alignment between real sequences may be readily calculated, as described below. In contrast, the significance of a global alignment is difficult to determine since a global alignment is readily produced between random sequences.

Optimal Combinations of Scoring Matrices and Gap Penalties for Finding Related Proteins

The usefulness of combinations of scoring matrices and gap penalties for identifying related proteins, including distantly related ones, has been compared (Feng et al. 1985; Doolittle 1986; Henikoff and Henikoff 1993; Pearson 1995, 1996, 1998; Agarwal and States 1998; Brenner et al. 1998). The method generally used is to start with a database of protein sequences organized into families, either based on sequence similarity or structural similarity (described in Chapters 7 and 9, respectively). A member of a family is then selected and used as a query sequence in a search of the entire database from which the sequence came, using a database similarity search method (FASTA, BLAST, SSEARCH), as described in Chapter 7. These methods basically use the dynamic programming algorithm and a choice of scoring matrix and gap penalties to produce alignment scores. Details of these studies are described on the book Web site.

In summary, the following general observations have been made: (1) Some scoring matrices are superior to others at finding related proteins based on either sequence or structure. For example, matrices prepared by examining the full range of amino acid substitutions in families of related proteins, such as the BLOSUM62 matrix, perform better than matrices based on variations in closely related proteins that are extrapolated to produce matrices for more distantly related sequences, such as the Dayhoff PAM250 matrix. (2) Gap penalties that for a given scoring matrix are adjusted to produce a local alignment are the most suitable. (3) To identify related sequences, the significance of the alignment scores should be estimated, as described in the following section.

These methods provide the means to demonstrate sequence similarity in even the most distantly related proteins. For closely related proteins, a PAM-type scoring matrix that matches the evolutionary separation of the sequences may provide a higher-scoring alignment, as described on page 82. Another set of studies has suggested that a global alignment algorithm in combination with scoring matrices that have all positive values and suitable gap penalties can be used to align proteins that have limited sequence similarity (i.e., 25% identity) but that have similar structure (Vogt et al. 1995; Abagyan and Batalov 1997).

ASSESSING THE SIGNIFICANCE OF SEQUENCE ALIGNMENTS

One of the most important recent advances in sequence analysis is the development of methods to assess the significance of an alignment between DNA or protein sequences. For sequences that are quite similar, such as two proteins that are clearly in the same family, such an analysis is not necessary. A significance question arises when comparing two sequences that are not so clearly similar but are shown to align in a promising way. In such a case, a significance test can help the biologist to decide whether an alignment found by the computer program is one that would be expected between related sequences or would just as likely be found if the sequences were not related. The significance test is also needed to evaluate the results of a database search for sequences that are similar to a sequence by the BLAST and FASTA programs (Chapter 7). The test will be applied to every sequence matched so that the most significant matches are reported. Finally, a significance test can also help to identify regions in a single sequence that have an unusual composition suggestive of an interesting function. Our present purpose is to examine the significance of sequence alignment scores obtained by the dynamic programming method.

Originally, the significance of sequence alignment scores was evaluated on the basis of the assumption that alignment scores followed a normal statistical distribution. If sequences are randomly generated in a computer by a Monte Carlo or sequence shuffling method, as in generating a sequence by picking marbles representing four bases or 20

amino acids out of a bag (the number of each type is proportional to the frequency found in sequences), the distribution may look normal at first glance. However, further analysis of the alignment scores of random sequences will reveal that the scores follow a different distribution than the normal distribution called the Gumbel extreme value distribution (see p. 104). In this section, we review some of the earlier methods used for assessing the significance of alignments, then describe the extreme value distribution, and finally discuss some useful programs for this type of analysis with some illustrative examples.

The statistical analysis of alignment scores is much better understood for local alignments than for global alignments. Recall that the Smith-Waterman alignment algorithm and the scoring system used to produce a local alignment are designed to reveal regions of closely matching sequence with a positive alignment score. In random or unrelated sequence alignments, these regions are rarely found. Hence, their presence in real sequence alignments is significant, and the probability of their occurring by chance alignment of unrelated sequences can be readily calculated. The significance of the scores of global alignments, on the other hand, is more difficult to determine. Using the Needleman-Wunsch algorithm and a suitable scoring system, there are many ways to produce a global alignment between any pair of sequences, and the scores of many different alignments may be quite similar. When random or unrelated sequences are compared using a global alignment method, they can have very high scores, reflecting the tendency of the global algorithm to match as many characters as possible. Thus, assessment of the statistical significance of a global alignment is a much more difficult task. Rather than being used as a strict test for sequence homology, a global alignment is more appropriately used to align sequences that are of approximately the same length and already known to be related. The method will conveniently show which sequence characters align. One can then use this information to perform other types of analyses, such as structural modeling or an evolutionary analysis.

Significance of Global Alignments

In general, global alignment programs use the Needleman-Wunsch alignment algorithm and a scoring system that scores the average match of an aligned nucleotide or amino acid pair as a positive number. Hence, the score of the alignment of random or unrelated sequences grows proportionally to the length of the sequences. In addition, there are many possible different global alignments depending on the scoring system chosen, and small changes in the scoring system can produce a different alignment. Thus, finding the best global alignment and knowing how to assess its significance is not a simple task, as reflected by the absence of studies in the literature.

Waterman (1989) provided a set of means and standard deviations of global alignment scores between random DNA sequences, using mismatch and gap penalties that produce a linear increase in score with sequence length, a distinguishing feature of global alignments. However, these values are of limited use because they are based on a simple gap scoring system. Abagyan and Batalov (1997) suggested that global alignment scores between unrelated protein sequences followed the extreme value distribution, similar to local alignment scores. However, since the scoring system that they used favored local alignments, these alignments they produced may not be global but local (see below). Unfortunately, there is no equivalent theory on which to base an analysis of global alignment scores as there is for local alignment scores. For zero mismatch and gap penalties, which is the most extreme condition for a global alignment giving the longest subsequence common to two sequences, the score between two random or unrelated sequences P is proportional to sequence length n, such that $P \simeq cn$ (Chvátal and Sankoff 1975), but it has not proven possible to calculate the proportionality constant c (Waterman and Vingron 1994a).

To evaluate the significance of a Needleman-Wunsch global alignment score, Dayhoff (1978) and Dayhoff et al. (1983) evaluated Needleman-Wunsch alignment scores for a large number of randomized and unrelated but real protein sequences, using their log odds scoring matrix at 250 PAMs and a constant gap penalty. The distribution of the resulting random scores matched a normal distribution. On the basis of this analysis, the significance of an alignment score between two apparently related sequences A and B was determined by obtaining a mean and standard deviation of the alignment scores of 100 random permutations or shufflings of A with 100 of B, conserving the length and amino acid composition of each. If the score between A and B is significant, the authors specify that the real score should be at least 3–5 standard deviations greater than the mean of the random scores. This level of significance means that the probability that two unrelated sequences would give such a high score is 1.35×10^{-3} (3 S.D.s) and 2.87×10^{-6} (5 S.D.s). In evaluating an alignment, two parameters were varied to maximize the alignment score: First, a constant called the matrix bias was added to each value in the scoring matrix and, second, the gap penalty was varied. The statistical analysis was then performed after the score between A and B had been maximized. Recall that the log odds PAM250 matrix values vary from -7 to 17 in units of 1/3 bits. The bias varied from 2 to 20 and had the effect of increasing the score by the bias times the number of alignment positions where one amino acid is matched to another. As a result, the alignment frequently decreases in length because there are fewer gaps, assuming the gap penalty is not also changed. It was these optimized alignments on which the significance test was performed. Feng et al. (1985) used the same method to compare the significance of alignment scores obtained by using different scoring matrices. They used 25–100 pairs of randomized sequences for each test of an alignment.

There are several potential problems with this approach, some of which apply to other methods as well. First, the method is expensive in terms of the number of computational steps, which increase at least as much as the square of sequence length because many Needleman-Wunsch alignments must be done. However, this problem is much reduced with the faster computers and more efficient algorithms of today. Second, if the amino acid composition is unusual, and if there is a region of low complexity (for example, many occurrences of one or two amino acids), the analysis will be oversimplified. Third, when natural sequences were compared more closely, the patterns found did not conform to a random set of the basic building blocks of sequences but rather to a random set of sequence segments that were varying. Consider use of the 26-letter alphabet in English sentences. Alphabet letters do not appear in any random order in these sentences but rather in a vocabulary of meaningful words. What happens if sentences, which are made up of words, are compared? On the one hand, if just the alphabet composition of many sentences is compared, not much variation is seen. On the other hand, if words are compared, much greater variation is found because there are many more words than alphabet characters. If random sequences are produced from segments of sequences, rather than from individual residues, more variation is observed, more like that observed when unrelated natural sequences are compared. The increased variation found among natural sequences is not surprising when one thinks of DNA and proteins as sources of information. For example, protein-encoding regions of DNA sequences are constrained by the genetic code and by amino acid patterns that produce functional domains in proteins.

Lipman et al. (1984) analyzed the distribution of scores among 100 vertebrate nucleic acid sequences and compared these scores with randomized sequences prepared in different ways. When the randomized sequences were prepared by shuffling the sequence to conserve base composition, as was done by Dayhoff and others, the standard deviation was approximately one-third less than the distribution of scores of the natural sequences. Thus, natural sequences are more variable than randomized ones, and using such randomized

sequences for a significance test may lead to an overestimation of the significance. If, instead, the random sequences were prepared in a way that maintained the local base composition by producing them from overlapping fragments of sequence, the distribution of scores has a higher standard deviation that is closer to the distribution of the natural sequences. The conclusion is that the presence of conserved local patterns can influence the score in statistical tests such that an alignment can appear to be more significant than it actually is. Although this study was done using the Smith-Waterman algorithm with nucleic acids, the same cautionary note applies for other types of alignments. The final problem with the above methods is that the correct statistical model for alignment scores was not used. However, these earlier types of statistical analysis methods set the stage for later ones.

The GCG alignment programs have a RANDOMIZATION option, which shuffles the second sequence and calculates similarity scores between the unshuffled sequence and each of the shuffled copies. If the new similarity scores are significantly smaller than the real alignment score, the alignment is considered significant. This analysis is only useful for providing a rough approximation of the significance of an alignment score and can easily be misleading.

Dayhoff (1978) and Dayhoff et al. (1983) devised a second method for testing the relatedness of two protein sequences that can accommodate some local variation. This method is useful for finding repeated regions within a sequence, similar regions that are in a different order in two sequences, or a small conserved region such as an active site. As used in a computer program called RELATE (Dayhoff 1978), all possible segments of a given length of one sequence are compared with all segments of the same length from another. An alignment score using a scoring matrix is obtained for each comparison to give a score distribution among all of the segments. A segment comparison score in standard deviation units is calculated as the difference between the value for real sequences minus the average value for random sequences divided by the standard deviation of the scores from the random sequences. A version of the program RELATE that runs on many computer platforms is included with the FASTA distribution package by W. Pearson. An example of the output of the RELATE program for the phage λ and P22 repressor sequences is shown in Table 3.8. This program also calculates a distribution based on the normal distribution, thus it provides only an approximate indication of the significance of an alignment.

Modeling a Random DNA Sequence Alignment

The above types of analyses assume that alignment scores between random sequences follow a normal distribution that can be used to test the significance of a score between two test sequences. For a number of reasons, mathematicians were concerned that this statistical model might not be correct. Let's start by creating two aligned random DNA sequences by drawing pairs of marbles from a large bag filled with four kinds of labeled marbles. The marbles are in equal proportions and labeled A, T, G, and C to represent an assumed equal representation of the four nucleotides in DNA. Now consider the probability of removing 10 identical pairs representing 10 columns in an alignment between two random sequences. The probability of removing an identical pair (an A and another A) is $1/4 \times 1/4$, but there are 4 possible identical pairs (A/A, C/C, G/G, and T/T), so that the probability of removing any identical pair is $4 \times 1/4 \times 1/4 = 1/4$ and that for removing 6 identical pairs is $(1/4)^6 = 2.4 \times 10^{-4}$. The probability of drawing a mismatched pair is $1 - 1/4 = 3/4$, and that of drawing 6/6 mismatched pairs $(3/4)^6 = 0.178$. Most random alignments produced in this manner will have a mixture of a few matches and many mismatches.

The calculations are a little more complex if the four nucleotides are not equally represented, but the results will be approximately the same. The probability of drawing the same

Table 3.8. *Distribution of alignment scores produced by program RELATE*

```
< -120      0 :
  -115      0 :
  -110      0 :
  -105      0 :
  -100      0 :
   -95      0 :
   -90      0 :
   -85      0 :
   -80      0 :
   -75      0 :
   -70      0 :
   -65      9 :.
   -60     69 :===
   -55    293 :===============
   -50    932 :==================================================
   -45   1868 :====================================================================
   -40   3214 :====================================================================
   -35   4784 :====================================================================
   -30   5858 :====================================================================
0  -25   6091 :====================================================================
   -20   5384 :====================================================================
   -15   4470 :====================================================================
1  -10   2960 :====================================================================
    -5   2076 :====================================================================
     0   1131 :====================================================================
2    5    590 :================================
    10    288 :===============
3   15    154 :========
    20     67 :===
    25     34 :=
4   30     18 :.
    35     10 :.
5   40      1 :.
    45      0 :
    50      0 :
6   55      0 :
    60      0 :
    65      0 :
7   70      0 :
    75      0 :
8   80      0 :
    85      0 :
    90      0 :
    95      0 :
   100      0 :
   105      0 :
   110      0 :
   115      0 :
   120      0 :
   125      0 :
>  125      0 :
40301 comparisons of window: 25, mean score: -27.3 (13.34)
matrix file: PAM250
29 segments >= 4 sd above mean
```

The sequences of two phage repressors were broken down into overlapping 25-amino-acid segments, and all 40,301 combinations of these segments were compared. The first column gives the approximate location of the number of standard deviations (13.34) from the mean score of −27.3. The second column is increasing ranges of the alignment score, and the third, the number of segment alignment scores, that fall within the range. Twenty-nine scores were greater than 3 standard deviations from the mean. Thus, these two sequences share segments that are significantly more related than the average segment, and the proteins share strong regions of local similarity. In such cases of strong local similarity, a local alignment program such as LFASTA, PLFASTA, or LALIGN can provide the alignments and a more detailed statistical analysis, as described below. Graph is truncated on right side.

pair is p, where $p = p_A^2 + p_C^2 + p_G^2 + p_T^2$, where p_X is the proportion of nucleotide X. p is an important parameter to remember for the discussion below. An even more complicated situation is when the two random sequences to align have different nucleotide distributions. One way would be to use an average p for the two sequences. This example illustrates the difficulty of modeling sequence alignments between two different organisms that have a different base composition.

The above model is not suitable for predicting the number of sequentially matched positions between random sequences of a given length. To estimate this number, a DNA sequence alignment may also be modeled by coin-tossing experiments (Arratia and Waterman 1989; Arratia et al. 1986, 1990). Random alignments will normally comprise mixtures of matches and mismatches, just as a series of coin tosses will produce a mixture of heads and tails. The chance of producing a series of matches in a sequence alignment with no mismatches is similar to the chance of tossing a coin and coming up with a series of only heads. The numbers of interest are the highest possible score that can be obtained and the probability of obtaining such a score in a certain number of trials. In such models, coins are usually considered to be "fair" in that the probability of a head is equal to that of a tail. The coin in this example has a certain probability p of scoring a head (H) and $q = 1 - p$ of scoring a tail (T). The longest run of heads R has been shown by Erdös and Rényi to be given by $\log_{1/p}(n)$. If $p = 0.5$ as for a normal coin, then the base of the logarithm is $1/p = 2$. For the example of $n = 100$ tosses, then $R = \log_2 100 = \log_e 100/\log_e 2 = 4.605/0.693 = 6.65$.

To use the coin model, an alignment of two random sequences $\mathbf{a} = a_1, a_2, a_3 \text{---} a_n$ and $\mathbf{b} = b_1, b_2, b_3 \text{---} b_n$, each of the same length n is converted to a series of heads and tails. If $a_i = b_i$ then the equivalent toss result is an H, otherwise the result is a T. The following example illustrates the conversion of an alignment to a series of H and T tosses.

$$\begin{array}{l} a_1\ a_2\ a_3 \text{ --- } a_n \text{----> H T H ---} \\ b_1\ b_2\ b_3 \text{ --- } b_n \quad \text{where } a_1 = b_1 \text{ and } a_3 = b_3 \text{ only} \end{array} \tag{8}$$

The longest run of matches in the alignment is now equivalent to the longest run of heads in the coin-tossing sequence, and it should be possible to use the Erdös and Rényi law to predict the longest run of matches. This score, however, only applies to one particular alignment of random sequences, such as generated above by the marble draw. In performing a sequence alignment, two sequences are in effect shifted back and forth with respect to each other to find regions that can be aligned. In addition, the sequences may be of different lengths. If two random sequences of length m and n are aligned in this same manner, the same law still applies but the length of the predicted match is $\log_{1/p}(mn)$ (Arratia et al. 1986). If $m = n$, the longest run of matches is doubled. Thus, for DNA sequences of length 100 and $p = 0.25$ (equal representation of each nucleotide), the longest expected run of matches is $2 \times \log_{1/p}(n) = 2 \times \log_4 100 = 2 \times \log_e 100 / \log_e 4 = 2 \times 4.605 / 1.386 = 6.65$, the same number as in the coin-tossing experiment. This number corresponds to the longest subalignment that can be expected between two random sequences of this length and composition.

A more precise formula for the expectation value or mean of the longest match M and its variance has been derived (Arratia et al. 1986; Waterman et al. 1987; Waterman 1989).

$$E(M) \simeq \log_{1/p}(mn) + \log_{1/p}(q) + \gamma \log(e) - 1/2 \tag{9}$$

$$\text{Var}\,[M(n,m)] \simeq [\pi \log_{1/p}(e)]^2/6 + 1/12 \qquad (10)$$

where $\gamma = 0.577$ is Euler's number and $q = 1 - p$. Note that Equation 9 can be simplified

$$E\,(M) \simeq \log_{1/p}(Kmn) \qquad (11)$$

where K is a constant that depends on the base composition.

Equation 11 also applies when there are k mismatches in the alignment, except that another term $= k \log_{1/p} \log_{1/p}(qmn)$ appears in the equation (Arratia et al. 1986). K, the constant in Equation 11, depends on k. The log log term is small and can be replaced by a constant (Mott 1992), and simulations also suggest that it is not important (Altschul and Gish 1996). Altschul and Gish (1996) have found a better match to Equation 11 when the length of each sequence is reduced by the expected length of a match. In the example given above with two sequences of length 100, the expected length of a match was 6.65. As the sequences slide align each other, it is not possible to have overlaps on the ends that are shorter than 7 because there is not enough sequence remaining. Hence, the effective length of the sequences is $100 - 7 = 93$ (Altschul and Gish 1996). This correction is also used for the calculation of statistical significance by the BLAST algorithm discussed in Chapter 7.

Equation 11 is fundamentally important for calculating the statistical significance of alignment scores. Basically, it states that as the lengths of random or unrelated sequences increase, the mean of the highest possible local alignment scores will be proportional to the logarithm of the product of the sequence lengths, or twice the logarithm of the sequence length if the lengths are equal (since $\log(nn) = 2 \log n$). Equation 10 also predicts a constant variance among scores of random or unrelated sequences, and this prediction is also borne out by experiment. It is important to emphasize once again that this relationship depends on the use of scoring parameters appropriate for a local alignment algorithm, such as 1 for a match and -0.9 for a mismatch, or a scoring matrix that scores the average aligned position as negative, and also upon the use of sufficiently large gap penalties. This type of scoring system gives rise to positive scoring regions only rarely. The significance of these scores can then be estimated as described herein.

Another way of describing the result in Equation 11 uses a different parameter, λ, where $\lambda = \log_e(1/p)$ (Karlin and Altschul 1990)

$$E\,(M) = [\log_e\,(Kmn)]\,/\,\lambda \qquad (12)$$

Recall that p is the probability of a match between the same two characters, given above as 1/4 for matching a random pair of DNA bases, assuming equal representation of each base in the sequences. p may also be calculated as the probability of a match averaged over scoring matrix and sequence composition values. Instead, it is λ that is more commonly used with scoring matrix values. The calculation of λ and also of K is described below and in more detail on the book Web site.

It is more useful in sequence analysis to use alignment scores instead of lengths for comparing alignments. The expected or mean alignment length between two random sequences given by Equations 11 and 12 can be easily converted to an alignment score just by using match and mismatch or scoring matrix values along with some simple normalization procedures. Thus, in addition to predicting length, these equations can also predict the mean

or expected value of the alignment scores $E(S)$ between random sequences of lengths m and n. Assessing statistical significance then boils down to calculating the probability that an alignment score between two random or unrelated sequences will actually go above $E(S)$. Hence, the expected score or mean extreme score is

$$E(S) = [\log_e (Kmn)] / \lambda \tag{13}$$

Another important mathematical result bearing on this question was that the number of matched regions that exceeds the mean score $E(S)$ in Equation 13 could be predicted by the Poisson distribution where the mean x of the Poisson distribution is given by $E(S)$ (Waterman and Vingron 1994b). The Poisson distribution applies when the probability of success in a single trial is small, but the number of trials is large (as in comparing many pairs of random sequences or a test sequence to many scrambled versions of a second sequence) so that some trials end in success but others do not. Some alignments do not reach the expected score, but others will reach or even exceed that score. The Poisson distribution gives the probability P_n of the number of successes, i.e., 0, 1, 2, 3 . . . when the average number is x and is given by the formula $P_n = e^{-x} x^n / n!$. The probability that no score from many test alignments will exceed x is therefore approximated by $(P_0 = e^{-x})$. The probability that at least one score exceeds x is $1 - P_0$ and is given by $P(S > x) = 1 - e^{-x}$, so that

$$\begin{aligned} P(S < x) &\simeq \exp(-E(S)) \\ &\simeq \exp(-Kmne^{-\lambda x}) \end{aligned} \tag{14}$$

$$P(S > x) \simeq 1 - \exp(-Kmne^{-\lambda x}) \tag{15}$$

Equation 15 estimates the probability of a score greater than x between two random sequences and is identical to the extreme value distribution described below. The Poisson approximation provides a very convenient way to estimate K and λ from alignment scores between many random or unrelated sequences by using the fraction of alignments that have a score less than value x (see book Web site).

Alignments with Gaps

It was predicted on mathematical grounds and shown experimentally that a similar type of analysis holds for sequence alignments that include gaps (Smith et al. 1985). Thus, when Smith et al. (1985) optimally aligned a large number of unrelated vertebrate and viral DNA sequences of different lengths (n and m) and their complements to each other, using a dynamic programming local alignment method that allowed for a score of +1 for matches, −0.9 for mismatches, and −2 for a single gap penalty (longer gaps were not considered in order to simplify the analysis), a plot of the similarity score (S) versus the $\log_1/p(nm)$ produced a straight line with approximately constant variance. This result is as expected in the above model except that with the inclusion of gaps, the slope was increased and was of the form

$$S_{mean} = 2.55 (\log_i/p(mn)) - 8.55 \quad (16)$$

with constant standard deviation $\sigma = 1.78$. This result was then used to calculate how many standard deviations were between the predicted mean and variance of the local alignment scores for unrelated sequences and the scores for test pairs of sequences. If the actual alignment score exceeded the predicted S_{mean} by several standard deviations, then the alignment score should be significant. For example, the expected score between two unrelated sequences of lengths 2948 and 431, average $p = 0.279$, was $S_{mean} = 2.55 \times \log_{1/0.279}(2948 \times 431) - 8.99 = 2.55 \times (\log_e(2948 \times 431)/\log_e(1/0.279)) - 8.99 = 2.55 \times 14.1 / 1.28 - 8.99 = 28.1 - 8.99 = 19.1$. The actual optimal alignment score between the two real sequences of these lengths was 37.20, which exceeds the alignment score expected for random sequences by $(37.20 - 19.1) / 1.78 = 10.2\sigma$. Is this number of standard deviations significant? Smith et al. (1985) and Waterman (1989) suggested the use of a conservative statistic known as Chebyshev's inequality, which is valid for many probability distributions: The probability that a random variable exceeds its mean is less than or equal to the square of 1 over the number of standard deviations from the mean. In this example where the actual score is 10 standard deviations above the mean, the probability is $(1/10)^2 = 0.01$.

Waterman (1989) has noted that for low mismatch and gap penalties, e.g., +1 for matches, −0.5 for mismatches, and −0.5 for a single gap penalty, the predicted alignment scores between random sequences as estimated above are not accurate because the score will increase linearly with sequence length instead of with the logarithm of the length. The linear relationship arises when the alignment is more global in nature, and the logarithmic relationship when it is local. Waterman (1989) has fitted alignment scores from a large number of randomly generated DNA sequences of varying lengths to either the predicted $\log(n)$ or n linear relationships expected for low- and high-valued mismatch and gap penalties. The results provide the mean and standard deviation of an alignment score for several scoring schemes, assuming a constant gap penalty.

With further mathematical analysis, it became apparent that the expected scores between alignment of random and unrelated sequences follow a distribution called the Gumbel extreme value distribution (Arratia et al. 1986; Karlin and Altschul 1990). This type of distribution is typical of values that are the highest or best score of a variable, such as the number of heads only expected in a coin toss discussed previously. Subsequently, S. Karlin and S. Altschul (1990, 1993) further developed the use of this distribution for evaluating the significance of ungapped segments in comparisons between a test sequence and a sequence database using the BLAST program (for review, see Altschul et al. 1994). The method is also used for evaluating the statistical features of repeats and amino acid patterns and clusters in the same sequence (Karlin and Altschul 1990; Karlin et al. 1991). The program SAPS developed by S. Karlin and colleagues at Stanford University and available at http://ulrec3.unil.ch/software/software.html provides this type of analysis. The extreme value distribution is now widely used for evaluating the significance of the score of local alignments of DNA and protein sequence alignments, especially in the context of database similarity searches.

The Gumbel Extreme Value Distribution

When two sequences have been aligned optimally, the significance of a local alignment score can be tested on the basis of the distribution of scores expected by aligning two random sequences of the same length and composition as the two test sequences (Karlin and

Altschul 1990; Altschul et al. 1994; Altschul and Gish 1996). These random sequence alignment scores follow a distribution called the extreme value distribution, which is somewhat like a normal distribution with a positively skewed tail in the higher score range. When a set of values of a variable are obtained in an experiment, biologists are used to calculating the mean and standard deviation of the entire set assuming that the distribution of values will follow the normal distribution. For sequence alignments, this procedure would be like obtaining many different alignments, both good and bad, and averaging all of the scores. However, biologically interesting alignments are those that give the highest possible scores, and lower scores are not of interest. The experiment, then, is one of obtaining a set of values, and then of using only the highest value and discarding the rest. The focus changes from the statistical approach of wanting to know the average of scores of random sequences, to one of knowing how high a value will be obtained next time another set of alignment scores of random sequences is obtained.

The distribution of alignment scores between random sequences follows the extreme value distribution, not the normal distribution. After many alignments, a probability distribution of highest values will be obtained. The goal is to evaluate the probability that a score between random or unrelated sequences will reach the score found between two real sequences of interest. If that probability is very low, the alignment score between the real sequences is significant and the sequence similarity score is significant.

The probability distribution of highest values in an experiment, the extreme value distribution, is compared to the normal probability distribution in Figure 3.17. The equations giving the respective y coordinate values in these distributions, Y_{ev} and Y_n, are

$$Y_{ev} = \exp[-x - e^{-x}] \quad \text{for the extreme value distribution} \qquad (17)$$

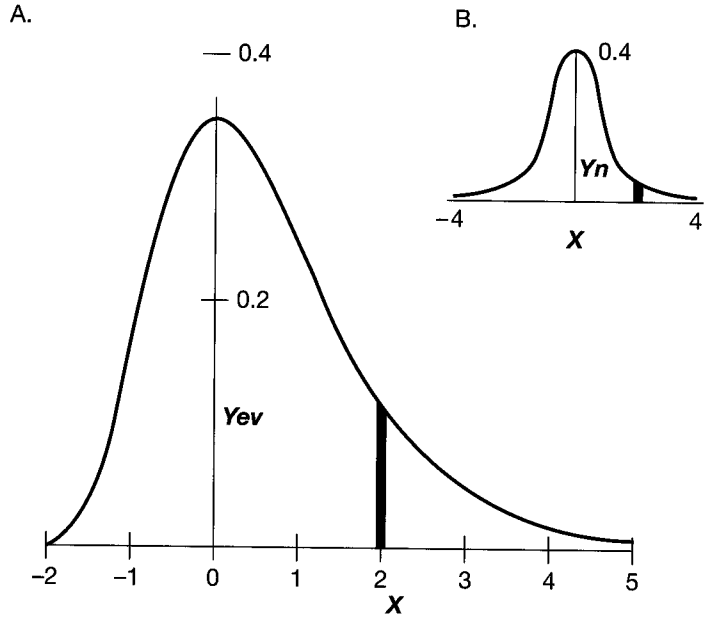


Figure 3.17. Probability values for the extreme value distribution (*A*) and the normal distribution (*B*). The area under each curve is 1.

$$Y_n = 1/\sqrt{(2\pi)} \exp [(-x^2)/2] \quad \text{for the normal distribution} \tag{18}$$

The area under both curves is 1. The normal curve is symmetrical about the expectation value or mean at $x = 0$, such that the area under the curve below the mean (0.5) is the same as that above the mean (0.5) and the variance σ^2 is 1. The probability of a particular value of x for the normal distribution is obtained by calculating the area under curve B, usually between $-x$ and $+x$. For $x = 2$, often used as an indication of a significant deviation from the mean, the area between -2 and $+2$ is 0.9544. For the extreme value distribution, the expectation value or mean of x is the value of the Euler-Mascheroni constant, 0.57722 . . . and the variance of x, σ^2, is the value of $\pi^2 / 6 = 1.6449$. The probability that score S will be less than value x, $P(S < x)$, is obtained by calculating the area under curve A from $-\infty$ to x, by integration of Equation 17 giving

$$P(S < x) = \exp [-e^{-x}] \tag{19}$$

and the probability of $S \geq x$ is 1 minus this probability

$$P(S \geq x) = 1 - \exp[-e^{-x}] \tag{20}$$

For the extreme value distribution, the area below $x = 0$, which represents the peak or mode of the distribution, is $1/e$ or 0.368 of the total area of 1, and the area above the mean is $1 - 0.368 = 0.632$. At a value of $x = 2$, $Yev = 0.118$ and $P(S < 2) = \exp [-e^{-2}] = 0.873$. Thus, just over 0.87 of the area under the curve is found below $x = 2$. An area of 0.95 is not reached until $x = 3$. The difference between the two distributions becomes even greater for larger values of x. As a result, for a variable whose distribution comes from extreme values, such as random sequence alignment scores, the score must be greater than expected from a normal distribution in order to achieve the same level of significance.

The above equations are modified for use with scores obtained in an analysis. For a variable x that follows the normal distribution, values of x are used to estimate the mean m and standard deviation σ of the distribution, and the probability curve given by Equation 18 then becomes

$$Yn = 1/(\sigma\sqrt{(2\pi)}) \exp [-(x - m)^2/2\sigma^2] \tag{21}$$

The probability of a particular value of x can be estimated by using m and σ to estimate the number of standard deviations from the mean, Z, where $Z = (x - m)/\sigma$. Similarly, Equations 17 and 20 can be modified to accommodate the extreme values such as sequence alignment scores

$$P(S \geq x) = 1 - \exp [-e^{-\lambda(x - u)}] \tag{22}$$

where u is the mode, highest point, or characteristic value of the distribution, and λ is the decay or scale parameter. As is apparent in Equation 22, λ converts the experimentally measured values into standard values of x after subtraction of the mode from each score.

It is quite straightforward to calculate u and λ, and several methods using alignment scores are discussed on the book Web site. There is an important relationship between u and λ, and the mean and standard deviation of a set of extreme values. The mean and standard deviation do not only apply to the normal distribution, but in fact are mathematically defined for any probability distribution. The mean of any set of values of a variable may always be calculated as the sum of the values divided by their number. The mean m or expected value of a variable x, $E(x)$, is defined as the first moment of the values of the variable around the mean. From this definition, the mean is that number from which the sum of deviations to all values is zero. The standard deviation σ^2 is the second moment of the values about the mean and is the sum of the squares of the devations from the mean divided by the number of observations less one $(n - 1)$. The mean $\underline{x}$ and standard deviation σ of a set of extreme values can be calculated in the same way, and then u and λ can be calculated using the following equations derived by mathematical evaluation of the first and second moments of the extreme value distribution (Gumbel 1962; Altschul and Erickson 1986).

$$\lambda = \pi / (\sigma\sqrt{6}) = 1.2825 / \sigma \tag{23}$$

$$u = \underline{x} - \gamma / \lambda = \underline{x} - 0.4500\,\sigma \tag{24}$$

where γ was already introduced. Equation 23 is derived from the ratio of the standard deviations σ^2 of the two distributions in Figure 3.17, or 1 to $\pi^2 / 6$. Equation 24 is derived from the observation that the mode or the EV distribution (zero in Fig. 3.17) has the value of γ less than the mean. However, the value of γ must be scaled by the ratio of the standard deviations. Hence γ / λ is subtracted from the mean. This method of calculating u and λ from means and standard deviations is called the method of moments.

As with the normal distribution, z scores may be calculated for each extreme value x, where $z = (x - m) / \sigma$ is the number of standard deviations from the mean m to each score. z scores are used by the FASTA, version 3, programs distributed by W. Pearson (1998). Equation 22 may be written in a form that directly uses z scores to evaluate the probability that a particular score Z exceeds a value z,

$$P(Z > z) = 1 - exp(-e^{-1.2825\,z\,-\,0.5772}) \tag{25}$$

For sequence analysis, u and λ depend on the length and composition of the sequences being compared, and also on the particular scoring system being used. They can be calculated directly or estimated by making many alignments of random sequences or shuffled natural sequences, using a scoring system that gives local alignments. The parameters will change when a different scoring system is used. Examples of programs that calculate these values are given below.

For alignments that do not include any gaps, u and λ may be calculated from the scoring matrix. The scaling factor λ is calculated as the value of x, which satisfies the condition

$$\Sigma\, p_i p_j e^{s_{ij}x} = 1 \tag{26}$$

where p_i and p_j are the respective fractional representations of residues i and j in the sequences, and s_{ij} is the score for a match being i and j, taken from a log odds scoring matrix. u, the characteristic value of the distribution, is given by (Altschul and Gish 1996)

$$u = (\ln Kmn) / \lambda \tag{27}$$

where m and n are the sequence lengths and K is a constant that can also be calculated from the values of p_i and s_{ij}. Note that this value originates from the coin toss analysis that gave rise to Equation 14. Combining Equations 25 and 27 eliminates u and gives the following relationship

$$\begin{aligned} P(S \geq x) &= 1 - \exp\left[-e^{-\lambda(x-u)}\right] \\ &= 1 - \exp\left[-e^{-\lambda(x-(\ln Kmn)/\lambda)}\right] \\ &= 1 - \exp\left[-e^{-\lambda x + \ln Kmn}\right] \qquad (28) \\ &= 1 - \exp\left[-Kmn\, e^{-\lambda x}\right] \qquad (29) \end{aligned}$$

To facilitate calculations, a sequence alignment score S may also be normalized to produce a score S'. The effect of normalization is to change the score distribution into the form shown above in Figure 3.17 with $u = 0$ and $\lambda = 1$. From Equation 28, S' is calculated by

$$S' = \lambda S - \ln Kmn \tag{30}$$

The probability of $P(S' > x)$ is then given by Equation 20 with $S = S'$

$$P(S' \geq x) = 1 - \exp\left[-e^{-x}\right] \tag{31}$$

The probability of a particular normalized score may then be readily calculated. This capability depends on a determination of the λ and K to calculate the normalized scores S' by Equation 30.

The probability function $P(S' \geq x)$ decays exponentially in x as x increases and $P(S' \geq x) = 1 - \exp[-e^{-x}] \rightarrow e^{-x}$. Consequently, an important approximation for Equations 29 and 31 for the significant part of the extreme value distribution where $x > 2$ is shown in Equations 32 and 33. Note that the replacement equations are single and not double exponentials.

$$P(S \geq x) \approx Kmn\, e^{-\lambda x} \tag{32}$$

$$P(S' \geq x) \approx e^{-x} \tag{33}$$

Table 3.9. *Approximation of $P(S' \geq x)$ by e^{-x}*

x	$1-\exp[-e^{-x}]$	e^{-x}
0	0.63	1
1	0.308	0.368
2	0.127	0.135
3	0.0486	0.0498
4	0.0181	0.0183

A comparison of probability calculations using this approximation instead of that given in Equation 31 is shown in Table 3.9. For $x > 2$, the estimates differ by less than 2%. The estimate given in Equation 32 also provides a quicker method for estimating the significance of an alignment score.

A Quick Determination of the Significance of an Alignment Score

Scoring matrices are most useful for statistical work if they are scaled in logarithms to the base 2 called bits. Scaling the matrices in this fashion does not alter their ability to score sequence similarities, and thereby to distinguish good matches from poor ones, but does allow a simple estimation of the significance of an alignment. The actual alignment may then be calculated by summing the matrix values for each of the aligned pairs, using matrix values in bit units. If the actual alignment score in bits is greater than expected for alignment of random sequences, the alignment is significant.

For a typical amino acid scoring matrix and protein sequence, K = 0.1 and λ depends on the values of the scoring matrix. If the log odds matrix is in units of bits as described above, then $\lambda = \log_e 2 = 0.693$, and the following simplified form of Equation 32 may be derived (Altschul 1991) by taking logarithms to the base 2 and setting p as the probability of the scores of random or unrelated alignments reaching a score of S or greater

$$\begin{aligned}\log_2 p &= \log_2 (Kmn\, e^{-\lambda S}) \\ &= \log_2 (Kmn) + \log_2 (e^{-\lambda S}) \\ &= \log_2 (Kmn) + (\log_e (e^{-\lambda S}))/\log_e 2 \\ &= \log_2 (Kmn) - \lambda S/\log_e 2 \\ &= \log_2 (Kmn) - S \end{aligned} \tag{34}$$

then S, the score corresponding to probability P, may be obtained by rearranging terms of Equation 34 as follows

$$\begin{aligned} S &= \log_2 (Kmn) - \log_2 P \\ &= \log_2 (K/P) + log_2 (nm) \end{aligned} \tag{35}$$

Since for most scoring matrices $K \simeq 0.1$ and choosing $P = 0.05$, the first term is 1, and the second term in Equation 35 becomes the most important one for calculating the score (Altschul 1991), thus giving

$$S \simeq \log_2 (nm) \tag{36}$$

Example: Using the Extreme Value Distribution to Calculate the Significance of a Local Alignment

Suppose that two sequences approximately 250 amino acids long are aligned by the Smith-Waterman local alignment algorithm using the PAM250 matrix and a high gap score to omit gaps from the alignment, and that the following alignment is found.

```
FWLEVEGNSMTAPTG
FWLDVQGDSMTAPAG
```

1. By Equation 36, a significant alignment between unrelated or random sequences will have a score of $S \simeq \log_2(nm) = \log_2(250 \times 250) = 16$ bits.
2. The score of the above actual alignment is 75 using the scores in the Dayhoff mutation data matrix (MDM) that provides log odds scores at 250 PAMs evolutionary distance.
3. A correction to the alignment score must be made because the MDM table at 250 PAMs is not in bit units but in units of logarithm to the base 10, multiplied by 10. These MDM scores actually correspond to units of 1/3 bits ([MDM score in units of $\log_{10}$] $\times$ 10 = [MDM score in bits of $\log_2 \times \log_2 10$] / 10 = [MDM score in units of $\log_{10} \times 10$] $\times$ 0.333). Thus, the score of the alignment in bits is 75/3 = 25 and 9 bits greater than the 16 expected by chance. Therefore, this alignment score is highly significant.
4. Altschul and Gish (1996) have provided estimates of $K = 0.09$ and $\lambda = 0.229$ for the PAM250 scoring matrix, for a typical amino acid distribution and for an alignment score based on using a very high gap penalty. By Equations 3.30 and 3.31, $S' = 0.229 \times 75 - \ln(0.09 \times 250 \times 250) = 17.18 - 8.63 = 8.55$ bits, and $P(S' \geq 8.55) = 1 - \exp[-e^{-8.55}] = 1.9 \times 10^{-4}$. Thus, the chance that an alignment between two random sequences will achieve a score greater than or equal to 75 using the MDM matrix is 1.9×10^{-4}. Note that the calculated S' of 8.55 bits in step 4 is approximately the same as the 9 bits calculated by the simpler method in step 3.
5. The probability may also be calculated by the approximation given in Equation 3.33 $P(S' > x) \simeq e^{-x} = e^{-8.55} = 1.9 \times 10^{-4}$.

The Importance of the Type of Scoring Matrix for Statistical Analyses

Using a log odds matrix in bit units simplifies estimation of the significance of an alignment. The Dayhoff PAM matrices, the BLOSUM matrices, and the nucleic acid PAM scoring matrices are examples of this type. Such matrices are also useful for finding local alignments because the matrix includes both positive and negative values. Another important feature of the log odds form of the scoring matrix is that this design is optimal for assessing statistical significance of alignment scores. A set of matrices, each designed to detect similarity between sequences at a particular level, is best for this purpose. Use of a matrix that is designed for aligning sequences that have a particular level of similarity (or evolutionary distance) assures the highest-scoring alignment and therefore the very best estimate of significance. Thus, lower-numbered PAM matrices are most suitable for aligning sequences that are more similar. In the above example, the Dayhoff PAM250 matrix designed for sequences that are 20% similar was used to align sequences that are approxi-

mately 20% identical and 50% similar (identities plus common replacements in the alignment). Using a lower PAM120 matrix produces a slightly higher score for this alignment, and thus increases the significance of the alignment score.

Another important parameter of the scoring matrix for statistical purposes is the expected value of the average amino acid pair, calculated as shown in Equation 37. This value should be negative if alignment scores for the matrix are to be used for statistical tests, as performed in the above example. Otherwise, in any aligned pair of sequences the scores will increase with length faster than the logarithm of the length. Not all scoring matrices will meet this requirement. To calculate the expected score (E), the score for each amino acid pair (s_{ij}) is multiplied by the fractional occurrences of each amino acid (p_i and p_j). This weighted score is then summed over all of the amino acid pairs. The expected values of the log odds matrices such as the Dayhoff PAM, BLOSUM, JTT, JO93, PET91, and Gonnet92 matrices all meet this statistical requirement.

$$E = \sum_{i=1}^{20} \sum_{j=1}^{i} p_i q_i s_{ij} \tag{37}$$

For example, for the PAM120 matrix in one-half bits $E = -1.64$ and for PAM160 in one-half bits, $E = -1.14$. Thus, scores obtained with these matrices may be used in the above statistical analysis. Ungapped alignment scores obtained using the BLOSUM62 matrix may also be subject to a significance test, as described above for the PAM matrices. The test is valid because the expect score for a random pair of amino acids is negative ($E = -0.52$). Because the matrix is in half-bit units, the alignment is significant when a score exceeds $16/0.52 \approx 32$ half-bits.

To assist in keeping track of information, scoring matrices have appeared in a new format suitable for use by many types of programs. An example is given in Figure 3.18. The matrix includes: (1) the scale of the matrix and the value of the statistical parameter λ; (2) E, the expect score of the average amino acid pair in the matrix, which if negative assures that local alignments will be emphasized (Eq. 37); (3) H, the information content or entropy of the matrix (Eq. 3) giving the ability of the matrix to discriminate related from unrelated sequence alignments, not shown here; and (4) suitable gap penalties. The BLOSUM matrices are also available in this same format.

Significance of Gapped, Local Alignments

When random sequences of varying lengths are optimally aligned with the Smith-Waterman dynamic programming algorithm using an appropriate scoring matrix and gap penalties, the distribution of scores also matches the extreme value distribution (Altschul and Gish 1996). Similarly, in optimally aligning a given sequence to a database of sequences, and after removing the high scores of the closely related sequences, the scores of the unrelated sequences also follow this distribution (Altschul et al. 1994; Pearson 1996, 1998). In these and other cases, optimal scores are found to increase linearly with log (n), where n is the sequence length. Equation 33 predicts that the optimal alignment score (x) expected between two random or unrelated sequences should be proportional to the logarithm of the product of the sequence lengths, $x \simeq \log_2(nm)$. If the sequence lengths are approximately equal, $n \simeq m$, then x should be proportional to $\log_2(n^2) = 2\log_2(n)$, and the predicted score should also increase linearly with $\log(n)$. $\log_2(n)$ is equivalent to $\log(n)$ because, to change the base of a logarithm, one merely multiplies by a constant. In comparing one sequence of length m to a sequence database of length n, m is a constant and

```
#
# This matrix was produced by "pam" Version 1.0.6 [28-Jul-93]
#
# PAM 120 substitution matrix, scale = ln(2)/2 = 0.346574  [1/2 bits]
#
# Expected score = -1.64, Entropy = 0.979 bits
#
# Lowest score = -8, Highest score = 12
#
   A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  3 -3 -1  0 -3 -1  0  1 -3 -1 -3 -2 -2 -4  1  1  1 -7 -4  0  0 -1 -1 -8
R -3  6 -1 -3 -4  1 -3 -4  1 -2 -4  2 -1 -5 -1 -1 -2  1 -5 -3 -2 -1 -2 -8
N -1 -1  4  2 -5  0  1  0  2 -2 -4  1 -3 -4 -2  1  0 -4 -2 -3  3  0 -1 -8
D  0 -3  2  5 -7  1  3  0  0 -3 -5 -1 -4 -7 -3  0 -1 -8 -5 -3  4  3 -2 -8
C -3 -4 -5 -7  9 -7 -7 -4 -4 -3 -7 -7 -6 -6 -4  0 -3 -8 -1 -3 -6 -7 -4 -8
Q -1  1  0  1 -7  6  2 -3  3 -3 -2  0 -1 -6  0 -2 -2 -6 -5 -3  0  4 -1 -8
E  0 -3  1  3 -7  2  5 -1 -1 -3 -4 -1 -3 -7 -2 -1 -2 -8 -5 -3  3  4 -1 -8
G  1 -4  0  0 -4 -3 -1  5 -4 -4 -5 -3 -4 -5 -2  1 -1 -8 -6 -2  0 -2 -2 -8
H -3  1  2  0 -4  3 -1 -4  7 -4 -3 -2 -4 -3 -1 -2 -3 -3 -1 -3  1  1 -2 -8
I -1 -2 -2 -3 -3 -3 -3 -4 -4  6  1 -3  1  0 -3 -2  0 -6 -2  3 -3 -3 -1 -8
L -3 -4 -4 -5 -7 -2 -4 -5 -3  1  5 -4  3  0 -3 -4 -3 -3 -2  1 -4 -3 -2 -8
K -2  2  1 -1 -7  0 -1 -3 -2 -3 -4  5  0 -7 -2 -1 -1 -5 -5 -4  0 -1 -2 -8
M -2 -1 -3 -4 -6 -1 -3 -4 -4  1  3  0  8 -1 -3 -2 -1 -6 -4  1 -4 -2 -2 -8
F -4 -5 -4 -7 -6 -6 -7 -5 -3  0  0 -7 -1  8 -5 -3 -4 -1  4 -3 -5 -6 -3 -8
P  1 -1 -2 -3 -4  0 -2 -2 -1 -3 -3 -2 -3 -5  6  1 -1 -7 -6 -2 -2 -1 -2 -8
S  1 -1  1  0  0 -2 -1  1 -2 -2 -4 -1 -2 -3  1  3  2 -2 -3 -2  0 -1 -1 -8
T  1 -2  0 -1 -3 -2 -2 -1 -3  0 -3 -1 -1 -4 -1  2  4 -6 -3  0  0 -2 -1 -8
W -7  1 -4 -8 -8 -6 -8 -8 -3 -6 -3 -5 -6 -1 -7 -2 -6 12 -2 -8 -6 -7 -5 -8
Y -4 -5 -2 -5 -1 -5 -5 -6 -1 -2 -2 -5 -4  4 -6 -3 -3 -2  8 -3 -3 -5 -3 -8
V  0 -3 -3 -3 -3 -3 -3 -2 -3  3  1 -4  1 -3 -2 -2  0 -8 -3  5 -3 -3 -1 -8
B  0 -2  3  4 -6  0  3  0  1 -3 -4  0 -4 -5 -2  0  0 -6 -3 -3  4  2 -1 -8
Z -1 -1  0  3 -7  4  4 -2  1 -3 -3 -1 -2 -6 -1 -1 -2 -7 -5 -3  2  4 -1 -8
X -1 -2 -1 -2 -4 -1 -1 -2 -2 -1 -2 -2 -2 -3 -2 -1 -1 -5 -3 -1 -1 -1 -2 -8
* -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8 -8  1
```

Figure 3.18. Example of BLASTP format of the Dayhoff MDM giving log odds scores at 120 PAMs. Note that the matrix has mirror-image copies of the same score on each side of the main diagonal. Besides the standard single-letter amino acid symbols, there are four new symbols, B, Z, X, *. B is the frequency-weighted average of entries for D and N pairs, Z similarly for Q and E entries, X similarly for all pairs in each row, and * is the lowest score in the matrix for matches with any other sequence character that may be present.

the predicted score should increase linearly as $\log(n)$. This $\log(n)$ relationship has been found in several studies of the distribution of optimal local alignment scores that have included gap penalties (Smith et al. 1985; Arratia et al. 1986; Collins et al. 1988; Pearson 1996, 1998; for additional references, see Altschul et al. 1994). Thus, the same statistical methods described above for assessing the significance of ungapped alignment scores may also be used for gapped alignment scores. Methods for calculating the parameters K and λ for a given combination of scoring matrix methods and gap penalties are described on the book Web site.

Methods for Calculating the Parameters of the Extreme Value Distribution

In the analysis by Altschul and Gish (1996), 10,000 random amino acid sequences of variable lengths were aligned using the Smith-Waterman method and a combination of the scoring matrix and a reasonable set of gap penalties for the matrix. The scores found by this method followed the same extreme value distribution predicted by the underlying statistical theory. Values of K and λ were then estimated for each combination by fitting the data to the predicted extreme value distribution. Some representative results are shown in

Table 3.10. Readers should consult Tables V–VII in Altschul and Gish (1996) for a more detailed list of the gap penalties tested.

Altschul and Gish (1996) have cautioned users of these statistical parameters. First, the parameters were generated by alignment of random sequences that were produced assuming a particular amino acid distribution, which may be a poor model for some proteins. Second, the accuracy of λ and K cannot be estimated easily. Finally, for gap costs that give values of $H < 0.15$, the optimal alignment length is a significant fraction of the sequence lengths and produces a source of error called the edge effect. The effect occurs when the expected length of an alignment is a significant fraction of the sequence length, and, as discussed earlier, alignments between sequences that overlap at their ends cannot be completed. The expected length is then subtracted from the sequence length before λ is estimated. If no such correction is done, λ may be overestimated.

These values for gap penalties should also not be construed to represent the best choice for a given pair of sequences or the only choices, simply because the statistical parameters are available. The process of choosing a gap penalty remains a matter of reasoned choice. In trying the effects of varying the gap penalty, it is important to recognize that as the gap penalty is lowered, the alignments produced will have more gaps and will eventually change from a local to a global type of alignment, even though a local alignment program is being used. In contrast, higher H values are generated by a very large gap penalty and produce alignments with no gaps (Table 3.10), thus suggesting an increased ability to discriminate between related and unrelated sequences. In this respect, Altschul and Gish (1996) note that beyond a certain point increasing the gap

Table 3.10. *Statistical parameters for combination of scoring matrices and affine gap penalties*

Scoring matrix	Gap opening penalty[b]	Gap extension penalty[b]	K	λ	H[c]
BLOSUM50	∞[a]	0-∞	0.232	0.11	0.34
BLOSUM50	15	8–15	0.09	0.222	0.31
BLOSUM50	11	8–11	0.05	0.197	0.21
BLOSUM50	11	1	—	—	—
BLOSUM62	∞[a]	0-∞	0.318	0.13	0.40
BLOSUM62	12	3–12	0.1	0.305	0.38
BLOSUM62	8	7–8	0.06	0.270	0.25
BLOSUM62	7	1	—	—	—
PAM250	∞[a]	0-∞	0.229	0.09	0.23
PAM250	15	5–15	0.06	0.215	0.20
PAM250	10	8-10	0.031	0.175	0.11
PAM250	11	1	—	—	—

Dashes indicate that no value can be calculated because the relationship between alignment score and sequence length is linear and not logarithmic, indicating that the alignment is global, not local, in character. Statistical significance may not be calculated for these gap penalty-scoring matrix combinations. The corresponding values for gap penalties define approximate lower limits that should be used.

[a] A value of ∞ for gap penalty will produce alignments with no gaps.

[b] The penalty for a gap opening of length 1 is the value of the gap opening penalty shown. The gap extension penalty is not added until the gap length is 2. Make sure that the alignment program uses this same scheme for scoring gaps. The extension penalty is shown over a range of values; values within this range did not change K and λ.

[c] The entropy in units of the natural logarithm.

extension penalty does not change the parameters, indicating that most gaps in their simulations are probably of length 1. However, reducing the gap penalty can also allow an alignment to be extended and create a higher scoring alignment. Eventually, however, the optimal local alignment score between unrelated sequences will lose the log length relationship with sequence length and become a linear function. At this point, gap penalties are no longer useful for obtaining local alignments and the above statistical relationships are no longer valid.

The higher the H value, the better the matrix can distinguish related from unrelated sequences. The lower the value of H, the longer the expected alignment. These conditions may be better if a longer alignment region is required, such as testing a structural or functional model of a sequence by producing an alignment. Conversely, scoring parameters giving higher values of H should produce shorter, more compact alignments. If $H < 0.15$, the alignments may be very long. In this case, the sequences have a shorter effective length since alignments starting near the ends of the sequences may not be completed. This edge effect can lead to an overestimation of λ but was corrected for in the above table (Altschul and Gish 1996).

Unfortunately, the above method for calculating the significance of an alignment score may not be used to test the significance of a global alignment score. The theory does not apply when these same substitution matrices are used for global alignments. Transformation of these matrices by adding a fixed constant value to each entry or by multiplying each value by a constant has no effect on the relative scores of a series of global alignments. Hence, there is no theoretical basis for a statistical analysis of such scores as there is for local alignments (Altschul 1991).

As discussed in Chapter 7, two programs are commonly used for database similarity searches: FASTA and BLAST. These programs both calculate the statistical significance of the higher scores found with similar sequences, but the types of analyses used to determine the statistical significance of these scores are somewhat different. BLAST uses the value of K and λ found by aligning random sequences and Equation 29, where n and m are shortened to compensate for inability of ends to align. FASTA calculates the statistical significance using the distribution of scores with unrelated sequences found during the database search. In effect, the mean and standard deviation of the low scores found in a given length range are calculated. These scores represent the expected range of scores of unrelated sequences for that sequence length (recall that the local alignment scores increase as the logarithm of the sequence length). The number of standard deviations to the high scores of related sequences in the same length range (z score) is then determined. The significance of this z score is then calculated according to the extreme value distribution expected of the z scores, given in Equation 25. This method is discussed in greater detail in Chapter 7. Pearson (1996) showed that these two methods are equally useful in database similarity searches for detecting sequences more distantly related to the input query sequence.

Pearson (1996) has also determined the influence of scoring matrices and gap penalties on alignment scores of moderately related and distantly related protein sequences in the same family. For two examples of moderately related sequences, the choice of scoring matrix and gap penalties (gap opening penalty followed by penalty for each additional gap position) did not matter, i.e., BLOSUM50 $-12/-2$, BLOSUM62 $-8/-2$, Gonnet93 $-10/-2$, and PAM250 -12, -2 all produced statistically significant scores. The scores of distantly related proteins in the same family depended more on the choice of scoring matrix and gap penalty, and some scores were significant and others were not. Pearson recommends using caution in evaluating alignment scores using only one particular combination of scoring matrix and gap penalties. He also suggests that using a

larger gap penalty, e.g., −14, −2 with BLOSUM50, can increase the selectivity of a database search for similarity (fewer sequences known to be unrelated will receive a significant alignment score).

A difficulty encountered by FASTA in calculating statistical parameters during a database search is that of distinguishing unrelated from related sequences, because only scores of unrelated sequences must be used. As score and sequence length information is accumulated during the search, the scores will include high, intermediate, and sometimes low scores of sequences that are related to the query sequence, as well as low scores and sometimes intermediate and even high scores of unrelated sequences. As an example, a high score with an unrelated database sequence can occur because the database sequence has a region of low complexity, such as a high proportion of one amino acid. Regardless of the reason, these high scores must be pruned from the search if accurate statistical estimates are to be made. Pearson (1998) has devised several such pruning schemes, and then determined the influence of the scheme on the success of a database search at demonstrating statistically significant alignment scores among members of the same protein family or superfamily. However, no particular scheme proved to be better than another.

Example: Use of the Above Principles to Estimate the Significance of a Smith-Waterman Local Alignment Score

The alignment shown in step 1 in the next example box is a local alignment between the phage λ and P22 repressor protein sequences used previously. The alignment is followed by a statistical analysis of the score in steps 2 and 3. To perform this analysis, the second sequence (the P22 repressor sequence) was shuffled 1000 times and realigned with the first sequence to create a set of random alignments. Two types of shuffling are available: first, a global type of shuffling in which random sequences are assembled based on amino acid composition and, second, a local one in which the random sequences are assembled by random selection of an amino acid from a sliding window of length n in the original sequence in order to preserve local amino acid composition as described on page 98 (an example of a global analysis is shown in step 2). The distribution of scores in each case was fitted to the extreme value distribution (Altschul and Gish 1996) to obtain estimates of λ and K to be used in the estimation of significance.

The program and parameters used were LALIGN (see Table 3.1 , p. 66), which produces the highest-scoring n independent alignments and which was described previously (p. 75), and the scoring matrix BLOSUM50 with a gap opening penalty of −12 and −2 for extra positions in the gap, with end gaps weighted. These programs do not presently have windows or Web page interfaces, and must be run using command line options.

The program PRSS performs a statistical analysis based on the correct statistical distribution of alignment scores, as shown below. PRSS version 3 (PRSS3) gives the results as z scores.

Example: Estimation of Statistical Significance of a Local Alignment Score

1. Optimal alignment of phage λ and P22 repressor sequences using the program LALIGN. The command line used was lalign -f -12 -g -2 lamc1.pro p22c2.pro 3> results.doc. The -f and -g flags indicate the gap opening and extension parameters to be used, and are followed by the sequence files in FASTA format, then a request for 3 alignments. No scoring matrix was specified and the default BLOSUM50 matrix was therefore used. Program output is directed to the file results.doc, as indicated by the symbol >. The alignment shown is the highest-scoring or optimal one using this scoring matrix and these gap penalties. The next two alignments reported were only 9 and 15 amino acids long and each one had a score of 35 (not shown). As discussed in the text, these alignments are produced by repeatedly erasing the previous alignment from the dynamic programming matrix and then rescoring the matrix to find the next best alignment. The fact that the first alignment has a much higher score than the next two is an indication that (1) there are no other reasonable alignments of these sequences and (2) the first alignment score is highly significant.

```
LALIGN finds the best local alignments between two sequences
 version 2.0u64 March 1998
Please cite:
 X. Huang and W. Miller (1991) Adv. Appl. Math. 12:373-381

 Comparison of:
(A) lamc1.pro  LAMC1  REFORMAT of: cipro.pro  check: -1  from: 1  - 237 aa
(B) p22c2.pro  P22C2  REFORMAT of: p22  check: 4729  from: 1  to  - 216 aa
 using matrix file: blosum50.mat, gap penalties: -12/-2

  36.1% identity in 208 aa overlap; score:  401 [1/2 bits]

             30        40        50        60        70        80
LAMC1   KKNELGLSQESVADKMGMGQSGVGALFNGINALNAYNAALLAKILKVSVEEFSPSIAREI
        ....: . : ...  .:...  ...     . .   :. :   :.: :. : . . . .
P22C2   RRKKLKIRQAALGKMVGVSNVAISQWERSETEPNGENLLALSKALQCSPDYLLKGDLSQT
              20        30        40        50        60        70

             90       100       110       120       130       140
LAMC1   YEMYEAVSMQPSLRSEYEYPVFSHVQAGMFSPELRTFTKGDAERWVSTTKKASDSAFWLE
          :..   .:  :. :  :..: :.::..    .. . :   : : .::   :...:::.
P22C2   NVAYHS-RHEP--RGSY--PLISWVSAGQWMEAVEPYHKRAIENWHDTTVDCSEDSFWLD
              80                90       100       110       120

            150       160       170        180       190       200
LAMC1   VEGNSMTAPTGSKPSFPDGMLILVDPEQAVEP--GDFCIARLGGD-EFTFKKLIRDSGQV
        :.:.:::::.:   :.:.::.:::::  :::   : .  .:.: :.  : ::::. :.:.
P22C2   VQGDSMTAPAGL--SIPEGMIILVDPE--VEPRNGKLVVAKLEGENEATFKKLVMDAGRK
        130       140       150          160       170       180

              210       220       230
LAMC1   FLQPLNPQYPMIPCNESCSVVGKVIASQ
        ::.:::::::::  :  .:...: :. ..
P22C2   FLKPLNPQYPMIEINGNCKIIGVVVDAK
            190       200       210
```

2. Statistical analysis with program PRSS using a global shuffling strategy. The program prompts for input information and requests the name of a file for saving output. The second sequence has been shuffled 1000 times conserving amino acid composition, and realigned to the first sequence. The distribution of scores is shown. Fitting the extreme value distribution to these scores provides an estimate of λ and K needed for performing the statistical estimate by Equation 31. Recent versions of PRSS estimate these parameters by the method of maximum likelihood estimation (Mott 1992; W. Pearson, pers. comm.) described on the book Web site.

```
 lamc1.pro, 237 aa vs p22c2.pro

       s-w  est
< 24    0    0:
  26    0    0:
  28    3    1:*==
  30   13    6:=====*=======
  32   27   21:====================*======
  34   68   50:==================================================*
  36   98   84:==================================================*
  38  128  111:==================================================*
  40  129  123:==================================================*
  42  105  121:==================================================*
  44  110  108:==================================================*
  46   63   91:==================================================*
  48   75   72:==================================================*
  50   35   56:==================================                 *
  52   48   42:==========================================*======
  54   30   32:============================== *
  56   19   23:==================   *
  58   17   16:===============*=
  60    6   13:======      *
  62    7    9:======= *
  64    7    6:=====*=
  66    2    5:==  *
  68    4    3:==*=
  70    0    2: *
  72    1    2:=*
  74    0    1:*
  76    1    1:*
  78    2    1:*=
  80    0    0:
  82    0    0:
  84    0    0:
  86    1    0:=
  88    1    0:=
  90    0    0:
  92    0    0:
  94    0    0:
> 96    0    0: O
 216000 residues in  1000 sequences,
 BLOSUM50 matrix, gap penalties: -12,-2
 unshuffled s-w score: 401; shuffled score range: 30 - 89
Lambda: 0.16931 K: 0.020441; P(401)= 3.7198e-27
For 1000 sequences, a score >=401 is expected 3.72e-24 times
```

The above method does not necessarily ensure that the choice of scoring matrix and gap penalties provides a realistic set of local alignment scores. In the comparable situation of matching a test sequence to a database of sequences, the scores also follow the extreme value distribution. For this situation, Mott (1992) has explained that for local alignments the end point of the alignment should on the average be half-way along the query sequence, and for global alignments, the end point should be beyond that half-way point. Pearson (1996) has pointed out that the presence of known, unrelated sequences in the upper part of the curve where $E > 1$ (see Chapter 7) can be an indication of an inappropriate scoring system.

The Statistical Significance of Individual Alignment Scores between Sequences and the Significance of Scores Found in a Database Search Are Calculated Differently

In performing a database search between a query sequence and a sequence database, a new comparison is made for each sequence in the database. Alignment scores between unrelated sequences are employed by FASTA to calculate the parameters of the extreme value distribution. The probability that scores between unrelated sequences could reach as high as those found for matched sequences can then be calculated (Pearson 1998). Similarly, in the database similarity search program BLAST, estimates of the statistical parameters are calculated based on the scoring matrix and sequence composition. The parameters are then used to calculate the probability of finding conserved patterns by chance alignment of unrelated sequences (Altschul et al. 1994). When performing such database searches, many trials are made in order to find the most strongly matching sequences.

As more and more comparisons between unrelated sequences are made, the chance that one of the alignment scores will be the highest one yet found increases. The probability of finding a match therefore has to be higher than the value calculated for a score of one sequence pair. The length of the query sequence is about the same as it would be in a normal sequence alignment, but the effective database sequence is very large and represents many different sequences, each one a different test alignment. Theory shows that the Poisson distribution should apply (Karlin and Altschul 1990, 1993; Altschul et al. 1994), as it did above for estimating the parameters of the extreme value distribution from many alignments between random sequences.

The probability of observing, in a database of D sequences, no alignments with scores higher than the mean of the highest possible local alignment scores s is given by e^{-Ds}, and that of observing at least one score s is $P \simeq 1 - e^{-Ds}$. For the range of values of P that are of interest, i.e., $P < 0.1$, $P = Ds$. If two sequences are aligned by PRSS as given in the above example, and the significance of the alignment is calculated, two scores must be considered. The probability of the score may first be calculated using the estimates of λ and K. Thus, in the phage repressor alignment, $P(s > 401) = 3.7. \times 10^{-27}$. However, to estimate the EV parameters, 1000 shuffled sequences were compared, and the probability that one of those sequences would score as high as 401 is given by Ds, or $1000 \times 3.7 \times 10^{-27} = 3.7 \times 10^{-24}$. These numbers are also shown in the statistical estimates computed by PRSS. Finally, if the score had arisen from a database search of 50,000 sequences, the probability of a score of 401 among this many sequence alignments is 5×10^{-19}, still a small number, but 50,000 larger than that for a single comparison. These probability calculations are used for reporting the significance of scores with database sequences by FASTA and BLAST, as described in Chapter 7.

SEQUENCE ALIGNMENT AND EVOLUTIONARY DISTANCE ESTIMATION BY BAYESIAN STATISTICAL METHODS

A recent development in sequence alignment methods is the use of Bayesian statistical methods to produce alignments between pairs of sequences (Zhu et al. 1998) and to calculate distances between sequences (Agarwal and States 1996). Before discussing these methods further, we provide some introductory comments about Bayesian probability.

Introduction to Bayesian Statistics

Bayesian statistical methods differ from other types of statistics by the use of conditional probabilities. These probabilities are used to derive the joint probability of two events or conditions. An example of a conditional probability is $P(B|A)$, meaning the probability of B, given A, whereas $P(B)$ is the probability of B, regardless of the value of A. Suppose that A can have two states, A1 and A2, and that B can also have two states, B1 and B2, as shown in Table 3.11. These states might, for instance, correspond to two allelic states of two genes. Then, $P(B) = P(B1) + P(B2) = 1$ and $P(A) = P(A1) + P(A2) = 1$. Suppose, further, that the probability $P(B1) = 0.3$ is known. Hence $P(B2) = 1 - 0.3 = 0.7$. In our genetic example, each probability might correspond to the frequency of an allele, for which p and q are often used. These probabilities $P(B1)$, etc., can be placed along the right margins of the table as the respective sum of each row or column and are referred to as the marginal probabilities.

Interest is now focused on filling in the missing data in the middle two columns of the table. The probability of A1 and B1 occurring together (the value to be entered in row B1 and column A1) is called the joint probability, $P(B1 \text{ and } A1)$ (also denoted $P[B1, A1]$). The marginal probability $P(A1)$ is also missing. The available information up to this point, called the prior information, is not enough to calculate the joint probabilities. With additional data on the co-occurrence of A1 with B1, etc., these joint probabilities may be derived by Bayes' rule. Suppose that the conditional probabilities $P(A1|B1) = 0.8$ and $P(A2|B2) = 0.70$ are known, the first representing, for example, the proportion of a population with allele B1 that also has allele A1. First, note that $P(A1|B1) + P(A2|B1) = 1$, and hence that $P(A2|B1) = 1.0 - 0.8 = 0.2$. Similarly, $P(A1|B2) = 1.0 - 0.70 = 0.3$. Then the joint probabilities and other conditional probabilities may be calculated by Bayes' rule, illustrated using the joint probability for A1 and B1 as an example.

$$P(A1 \text{ and } B1) = P(B1)\,P(A1|B1) \tag{38}$$

$$P(A1 \text{ and } B1) = P(A1)\,P(B1|A1) \tag{39}$$

Thus, $P(A1 \text{ and } B1) = P(B1) \times P(A1|B1) = 0.3 \times 0.8 = 0.24$, and $P(A2 \text{ and } B2) = P(B2) \times P(A2|B2) = 0.7 \times 0.7 = 0.49$. The other joint probabilities may be calculated by subtraction; e.g., $P(A1 \text{ and } B2) = P(B1) - P(A1 \text{ and } B1) = 0.30 - 0.24 = 0.06$. To calculate

Table 3.11. *Prior information for a Bayes analysis*

	A1	A2	
B1			0.3
B2			0.7
			1.0

Table 3.12. *Completed table of joint and marginal probabilities*

	A1	A2	
B1	0.24	0.06	0.3
B2	0.21	0.49	0.7
	0.45	0.55	1.0

$P(A1)$ and $P(A2)$, the joint probabilities in each column may be added, thereby completing the additions to the table, and shown in Table 3.12.

However, note that $P(A1)$ may also be calculated in the following manner,

$$\begin{aligned} P(A1) &= P(A1 \text{ and } B1) + P(A1 \text{ and } B2) \\ &= P(B1)\,P(A1|B1) + P(B2)\,P(A1|B2) \end{aligned} \tag{40}$$

Other conditional probabilities may be calculated from Equations 38 and 39 by rearranging terms and by substituting Equation 40, and the following form of Bayes' rule may be derived,

$$\begin{aligned} P(B2|A1) &= P(A1 \text{ and } B2) / P(A1) \\ &= P(B2)\,P(A1|B2) / P(A1) \\ &= P(B2)\,P(A1|B2) / [P(B1)\,P(A1|B1) + P(B2)\,P(A1|B2)] \end{aligned} \tag{41}$$

Using Equation 41, $P(B2|A1) = 0.7 \times 0.30 / [0.3 \times 0.80 + 0.7 \times 0.3] = 0.467$, and also $P(B1|A1) = 1.0 - 0.467 = 0.533$. Such calculated probabilities are called posterior probabilities or posteriors, as opposed to the prior probabilities or priors initially available. Thus, based on the priors and additional information, application of Bayes' rule allows the calculation of posterior estimates of probabilities not initially available. This procedure of predicting probability relationships among variables may be repeated as more data are collected, with the existing model providing the prior information and the new data providing the information to derive a new model. The initial beliefs concerning a parameter of interest are expressed as a prior distribution of the parameter, the new data provide a likelihood for the parameter, and the normalized product of the prior and likelihood (Eq. 41) forms the posterior distribution.

Example: Bayesian Analysis

Another illustrative example of a Bayesian analysis is the game played by Monty Hall in the television game show "Let's Make a Deal." Behind one of three doors a prize is placed by the host. A contestant is then asked to choose a door. The host opens one door (one that he knows the prize is not behind) and reveals that the prize is not behind that door. The contestant is then given the choice of changing to the other door of the three to win. The initial or prior probability for each door is 1/3, but after the new information is provided, these probabilities must be revised. The original door chosen still has a probability of 1/3, but the second door that the prize could be behind now has a probability of 2/3. These new estimates are posterior probabilities based on the new information provided.

In the above example, note that the joint probability of A1 and B1 [P(A1 and B1)] is not equal to the product of P(A1) and P(B1); i.e., 0.24 is not equal to $0.3 \times 0.45 = 0.135$. Such would be the case if the states of A and B were completely independent; i.e., if A and B were statistically independent variables as, for example, in a genetic case of two unlinked genes A and B. In the above example, the state of one variable is influencing the state of the other such that they are not independent of each other, as might be expected for linked genes in the genetic example.

A more general application of Bayes' rule is to consider the influence of several variables on the probability of an outcome. The analysis is essentially the same as that outlined above. To see how the method works with three instead of two values of a variable, think first of an example of three genes, each having three alleles, and of deriving the corresponding conditional probabilities. The resulting joint probabilities will depend on the choice made of the three possible values for each variable. To go even farther, instead of a small number of discrete sets of alternative values of a variable, Bayesian statistical methods may also be used with a large number of values of variables or even with continuous variables.

For sequence analysis by Bayesian methods, a slightly different approach is taken. The variables may include combinations of possible alignments, gap scoring systems, and log odds substitution matrices. The most probable alignments may then be identified. The scoring system used for sequence alignments is quite readily adapted to such an analysis. In an earlier discussion, it was pointed out that a sequence alignment score in bits is the logarithm to the base 2 of the likelihood of obtaining the score in alignments of related sequences divided by the likelihood of obtaining the score in alignments of unrelated sequences. It was also indicated that the highest alignment score should be obtained if the scoring matrix is used that best represents the nucleotide or amino acid substitutions expected between sequences at the same level of evolutionary distance. Bayesian methodology carries this analysis one step farther by examining the probabilities of all possible alignments of the sequences using all possible variations of the input parameters and matrices. These selections are the prior information for the Bayesian statistical analysis and provide various estimates of the alignment that allow us to decide on the most probable alignments. The alignment score for each combination of these variables provides an estimate of the probability of the alignment. By using equations of conditional probability such as Equation 41, posterior information on the probability of alignments, gap scoring system, and substitution matrix can be obtained. For further reading, a Bayesian bioinformatics tutorial by C. Lawrence is available at http://www.wadsworth.org/resnres/bioinfo/.

Application of Bayesian Statistics to Sequence Analysis

To use an example from sequence analysis, a local alignment score (s) between two sequences varies with the choice of scoring matrix and a gap scoring system. In the previous sections, an amino acid scoring matrix was chosen on the basis of its performance in identifying related sequences. Gap penalties were then chosen for a particular scoring matrix on the basis of their performance in identifying known sequence relationships and of their keeping a local alignment behavior by the increase in score between unrelated sequences remaining a logarithmic function of sequence length. The alignment score expressed in bit units was the ratio of the alignment score expected between related sequences to that expected between unrelated sequences, expressed as a logarithm to the base 2. The scores may be converted to an odds ratio (r) using the formula $r = 2^s$. The probability of such a score between unrelated or random

sequences can then be calculated using the parameters for the extreme value distribution for that combination of scoring matrix and gap penalty. Finally, the above analysis may provide several different alignments, without providing any information as to which is the most likely. With the application of Bayesian statistics, the approach is different.

The application of Bayesian statistics to this problem allows one to examine the effect of prior information, such as the chosen amino acid substitution matrix, on the probability that two sequences are homologous. The method provides a posterior probability distribution of all alignments taking into account all possible scoring systems. Thus, the most likely alignments and their probabilities may be determined. This method circumvents the need to choose a particular scoring matrix and gap scoring system because a range of available choices can be tested. The approach also provides conditional posterior distributions on the gap number and substitution matrix. Another application of Bayes statistics for sequence analysis is to find the PAM DNA substitution matrix that provides the maximum probability of a given level of mismatches in a sequence alignment, and thus to predict the evolutionary distance between the sequences.

Bayesian Evolutionary Distance

Agarwal and States (1996) have applied Bayesian methods to provide the best estimate of the evolutionary distance between two DNA sequences. The examples used are sequences of the same length that have a certain level of mismatches. Consequently, there are no gaps in the alignment between the sequences. Sequences of this type originated from gene duplication events in the yeast and *Caenorhabditis elegans* genomes. When there are multiple mismatches between such repeated sequences, it is difficult to determine the most likely length of the repeats. With the application of Bayesian methods, the most probable repeat length and evolutionary time since the repeat was formed may be derived.

The alignment score in bits between sequences of this type may be calculated from the values for matches and mismatches in the DNA PAM scoring matrices described earlier (Table 3.6). Recall that a PAM1 evolutionary distance represents a change of 1 sequence position in 100 and is thought to correspond roughly to an evolutionary distance of 10^7 years. Higher PAMN tables are calculated by multiplying the PAM1 scoring matrix by itself n times. This Markovian model of evolution assumes that any sequence position can change with equal probability, and subsequent changes at a site are not influenced by preceding changes at that site. In addition, a changed position can revert to the original nucleotide at that position. The problem is to discover which scoring matrix (PAM50, 100, etc.) gives the most likely alignment score between the sequences. This corresponding evolutionary distance will then represent the time at which the sequence duplication event could have occurred.

An approach described earlier was to evaluate the alignment scores using a series of matrices and then to identify the matrix giving the highest similarity score. For example, if there are 60 mismatches between sequences that are 100 nucleotides long, the PAM50 matrix score of the alignment in bits ($\log_2$) is $40 \times 1.34 - 60 \times 1.04 = -8.8$, but the PAM125 matrix score is much higher, $40 \times 0.65 - 60 \times 0.30 = 8$. When these log odds scores in bits are converted to odds scores, the difference is 0.002 versus 256. Thus, the PAM125 matrix provides a much better estimate of the evolutionary distance between sequences that have diverged to this degree. The Bayesian approach continues this type of analysis to discover the probability of the alignment as a function of each

evolutionary distance represented by a different PAM matrix. If x is the evolutionary distance represented by the PAMN matrix divided by 100, and k is the number of mismatches in a sequence of length n, then by Bayes' rule and related formulas discussed above

$$P(x|k) = P(k|x)\,P(x) / P(k)$$
$$= P(k|x)\,P(x) / \Sigma_x P(k|x) \qquad (42)$$

$P(x|k)$ is the probability of distance x given the sequence with k mismatches (and $n - k$ matches), $P(k|x)$ is the odds score for the sequence with k mismatches using the log odds scores in the DNA PAM100x matrix, and $P(x)$ is the prior probability of distance x (usually 1 over the number of matrices, thus making each one equally possible). The denominator is the sum of the odds scores over the range of x, which is 0.01 − 4, representing PAM1 to PAM400 (~10 million to 4 billion years). Like the conditional probabilities calculated by Equation 42, this sum represents the area under the probability curve and has the effect of normalizing the probability for each individual scoring matrix used. The shape of the probability curve reveals how $P(x|k)$ varies with x. An example is shown in Figure 3.19.

The probability curves have a single mode or highest score for $k < 3n/4$. Because the curves are not symmetrical about this mode but are skewed toward higher distances, the expected value or mean of the distribution and its standard deviation are the best indication of evolutionary distance. For a sequence 100 nucleotides long with 40 mismatches, the expected value of x is 0.60 with $s = 0.11$, representing a distance of ~600 million years. These estimates are different from the earlier method that was described of finding the matrix that gives the highest alignment score, which would correspond to the mode or highest scoring distance. Other methods of calculating evolutionary distances are described in Chapter 6.

Figure 3.19. $P(x|k)$ for sequence length $n = 100$ and number of mismatches $k = 40$ or 60. (Redrawn from Agarwal and States 1996.)

Working with Odds Scores

Odds scores, and probabilities in general, may be either multiplied or added, depending on the type of analysis. If the purpose is to calculate the probability of one event AND a second event, the odds scores for the events are multiplied. An example is the calculation of the odds of an alignment of two sequences from the alignment scores for each of the matched pairs of bases or amino acids in the alignment. The odds scores for the pairs are multiplied. Usually, the log odds score for the first pair is added to that for the second, etc., until the scores for every pair have been added. An odds score of the alignment in units of logarithm to the base 2 (bits) may then be calculated by the formula odds score = 2 raised to the power of the log odds score. A second type of probability analysis is to calculate the odds score for one event OR a second event, or of a series of events (event 1 OR event 2 OR event 3). In this case, the odds scores are added. An example is the calculation of the odds score for a given sequence alignment using a series of alternative PAM scoring matrices. The alignment scores are calculated in log odds units and then converted into odds scores as described above. The odds scores for the sequences using matrix 1 are added to the odds score using matrix 2, then to the score using matrix 3, and so on, thereby generating the odds score for the set of matrices. From this sum of odds scores, the probability of obtaining one of the odds scores S is S divided by the sum. There are also a number of other uses of this same type of calculation for locating common patterns in a set of sequences by statistical methods that are discussed in Chapter 4.

One difficulty with making such estimations is that the estimate depends on the assumption that the mutation rate in sequences has been constant with time (the molecular clock hypothesis) and that the rate of mutation of all nucleotides is the same. Such problems may be solved by scoring different portions of a sequence with a different scoring matrix, and then using the above Bayesian methods to calculate the best evolutionary distance. Another difficulty is deciding on the length of sequence that was duplicated. In genomes, the presence of repeats may be revealed by long regions of matched sequence positions dispersed among regions of sequence positions that do not match. However, as the frequency of mismatches is increased, it becomes difficult to determine the extent of the repeated region. The application of the above Bayesian analysis allows a determination of the probability distributions as a function of both length of the repeated region and evolutionary distance. A length and distance that gives the highest overall probability may then be determined. Such alignments are initially found using an alignment algorithm and a particular scoring matrix. Analysis of the yeast and *C. elegans* genomes for such repeats has underscored the importance of using a range of DNA scoring matrices such as PAM1 to PAM120 if most repeats are to be found (Agarwal and States 1996). One disadvantage of the Bayesian approach is that a specific mutational model is required, whereas other methods, such as the maximum likelihood approach described in Chapter 6, can be used to estimate the best mutational model as well as the distance. Computationally, however, the Bayesian method is much more practical.

Bayesian Sequence Alignment Algorithms

Zhu et al. (1998) have devised a computer program called the Bayes block aligner which in effect slides two sequences along each other to find the highest scoring ungapped regions

or blocks. These blocks are then joined in various combinations to produce alignments. There is no need for gap penalties because only the aligned sequence positions in blocks are scored. Instead of using a given substitution matrix and gap scoring system to find the highest scoring alignment, a Bayesian statistical approach is used. Given a range of substitution matrices and number of blocks expected in an alignment as the prior information, the method provides posterior probability distributions of alignments. The Bayes aligner is available through a licensing agreement from http://www.wadsworth.org/resnres/bioinfo. A graphical interface for X windows in a UNIX environment and a nongraphical interface for PCs running Windows are available. The method may be used for both protein and DNA sequences. An alignment block between two sequences is defined as a run of one or more identical characters in the sequence alignment that can include intervening mismatches but no gaps, as shown in the following example. Only the aligned blocks are identified and scored; regions of unaligned sequence and gaps between these blocks are not scored. The probability of a given alignment is given by the product of the probabilities of the individual alignment scores in the blocks, as indicated in the following example. The Bayes block aligner scores every possible combination of blocks to find the best scoring alignment.

Example: Block Alignment of Two Sequences and of the Scoring of the Alignment as Used in the Bayes Block Aligner (Zhu et al. 1998)

The score of the alignment is obtained by adding the log odds scores of each aligned pair in each block. Sequence not within these blocks is not scored and there is no penalty for gaps. Regions of both sequences that are not aligned can be present within the gap. The sequence alignment score is therefore determined entirely by the placement of block boundaries.

```
                Block 1                    Block 2

Sequence 1      S  G  T  G  K (gap) K  K  R  L  E
Sequence 2      P  G  S  G  K (gap) K  Q  R  L  T

BLOSUM62
score           -1 6  1  6  5       5  1  5  4  -1

Sum of scores              =    31 half bits
                           =    15.5 bits

Odds of alignment score

                           =    2^15.5 to 1
                           =    4.6 x 10^4 to 1
```

Unlike the commonly used methods for aligning a pair of sequences, the Bayesian method does not depend on using a particular scoring matrix or designated gap penalties. Hence, there is no need to choose a particular scoring system or gap penalty. Instead, a number of different scoring matrices and range of block numbers up to some reasonable maximum are examined, and the most probable alignments are determined. The Bayesian method provides a distribution of alignments weighted according to probability and can also provide an estimate of the evolutionary distance between the sequences that is independent of scoring matrix and gaps.

Like dynamic programming methods and the BLAST and FASTA programs, the Bayes block aligner has been used to find similar sequences in a database search. The most extensive comparisons of database searches have shown that the program SSEARCH based on the Smith-Waterman algorithm, with the BLOSUM50 -12,-2 matrix and gap penalty scoring system, can find the most members of protein families previously identified on the basis of sequence similarity (Pearson 1995, 1996, 1998) or structural homology (Brenner et al. 1998). In a similar comprehensive analysis, Zhu et al. have shown that the Bayes block aligner has a slightly better rate than even SSEARCH of finding structurally related sequences at a 1% false-positive level. Hence, this method may be the best one to date for database similarity searching.

The Bayes block aligner defines blocks by an algorithm due to Sankoff (1972). This algorithm is designed to locate blocks by finding the best alignment between two sequences for any reasonable number of blocks. The example shown in Figure 3.20 illustrates the basic block-finding algorithm.

Following the initial finding of block alignments in protein sequences by the Sankoff method, the Bayes block aligner calculates likelihood scores for these alignments for various block numbers and amino acid or DNA substitution matrices. To be biologically more meaningful by avoiding too many blocks, the number of protein sequence blocks k is limited from zero to 20 or the length of the shorter sequence divided by 10, whichever is smaller. For a set of amino acid substitution matrices such as the Dayhoff PAM or BLOSUM matrices, the only requirement is that they should be in the log odds format in order to provide the appropriate likelihood scores by additions of rows and columns in the V and W matrices (Fig. 3.20). A large number of matrices like the V and W matrices in Figure 3.20 are used, each for a different amino acid substitution matrix and block number. In each of these matrices, a number of alignments of the block regions that are found are possible. The score in the lower right-hand corner of each matrix is the sum of the odds scores of all possible alignments in that particular matrix. The odds scores thus calculated in each matrix are summed to produce a grand total of odds scores. The fraction of this total that is shared by a set of alignments under given conditions (e.g., a given number of blocks or an amino acid substitution matrix) provides the information needed to calculate the most probable scoring matrix, block number, etc., by Bayesian formulas. The joint probabilities equivalent to the interior row and column entries in Tables 3.11 and 3.12 are then calculated. In this case, each joint probability is the likelihood of the alignment given a particular block alignment, number of blocks, and substitution matrix, multiplied by the prior probabilities. These prior probabilities of particular alignment, block number, and scoring matrix are treated as having an equally likely prior probability. Once all joint probabilities have been computed for every combination of the alignment variables, the conditional posterior information can be obtained by Bayes' rule, using equations similar to Equation 41. As in Equation 41, the procedure involves

dividing the sum of all alignment likelihoods that apply to a particular value of a particular variable by the sum of all alignment likelihoods found for all variables.

Use of the Bayes Block Aligner for Pair-wise Sequence Alignment

There are several possible uses of the Bayes block aligner for sequence alignment. The overall probability that a given pair of residues should be aligned may be found by several methods. In the first, alignments may be sampled in proportion to their joint posterior probability, as for example, alignments produced by a particular combination of substitution matrix and gap number. A particular substitution matrix and gap number may be chosen based on their posterior probabilities. An alignment may then be obtained from the alignment matrix in much the same manner as the trace-back procedure used to find an alignment by dynamic programming. Once a number of sample alignments has been obtained, these samples may be used to estimate the marginal distribution of all alignments. This distribution then gives the probability that each pair of residues will align. An alternative method of sampling the joint posterior probability distribution is to identify an average alignment for k blocks by sampling the highest peaks in the marginal posterior alignment distribution and by using each successively lower peak as the basis for another alignment block down to a total of k blocks, concatenating any overlaps. These alignments may then be used to obtain the probability of each aligned residue. In the second method, the exact marginal posterior alignment distribution of a specific pair of residues may be obtained by summing over all substitution matrices and possible blocks.

Third, optimal alignment and near-optimal alignments for a given number of blocks can also be obtained. Finally, the Bayes block aligner provides an indication as to whether or not the sequence similarity found is significant. Bayesian statistics examines the posterior probabilities of all alternative models over all possible priors. The Bayesian evidence that two sequences are related is given by the probability that K, the maximum allowed number of blocks, is greater than 0, as calculated in the following example taken from Zhu et al. (1998). The posterior probability of the number of blocks, the substitution matrices, and the aligned residues can all be calculated as described above.

Example: Bayes Block Aligner (Zhu et al. 1998)

The proteins guanylate kinase from yeast (PDB id. 1GKY) and adenylate kinase from beef heart (PDB id. 2AK3, chain A) are known to be structurally related and are from a database of protein sequences that are 26–35% identical. These proteins were aligned with the Bayes block aligner using as prior information an equal chance that the block number k can be any number between 0 and 18, and that the BLOSUM30 to 100 substitution matrices can each equally well predict the aligned positions. The posterior probability distribution of the number of blocks, k, is shown in Figure 3.21A. Values $k > 0$ indicate the possibility of finding one or more blocks. In this example, the probability for values of k is approximately the same for $k > 8$. Below 8, the values decrease

gradually to a low value at $k = 1$ and then increase again abruptly for $k = 0$. The total area under the curve from $k = 0$ to $k = 18$ has been set to 1.

The cumulative posterior probability that the block number K is greater than a given value k is shown in Figure 3.21B. The area under the curve for $k \geq 1$ has the value 0.938. Although at first glance this number appears to represent the probability that the sequences are related, i.e., that $K > 0$, the probability is actually higher by Bayesian standards. Instead, the maximum value for $P(k|\text{sequences})$ in Figure 3.21A, i.e., 0.0731 at $k = 8$, is used. This number times the maximum number of blocks $0.0731 \times 18 = 1.316$, represents the accumulated best evidence that the blocks are related or that $K > 0$. This calculation assumes that all block numbers are equally likely or that $p(k|k>0) = 1/K = 1/18$. The value $P(k = 0|\text{sequences}) = 0.0621$ is the corresponding best evidence that the sequences are not related or that $K = 0$. The probability that the sequences are related is then calculated as $1.316 / (1.316 + 0.0621) = 0.955$. This value is the supremum of $P(k > 0)$ taken over all prior distributions on k, where the supremum is a mathematical term that refers to the least upper bound of a set of numbers. This high a Bayesian probability is strong evidence for the hypothesis that the sequences are homologous. Normally, a Bayesian probability of $p > 0.5$ will suffice (Zhu et al. 1998).

The posterior probability distribution for the BLOSUM scoring matrices for alignment of these same two proteins is shown in Table 3.13. Note that the highest probabilities are for BLOSUM tables between BLOSUM50 and BLOSUM 80, and that the highest probability is at BLOSUM62, which is commonly used for protein sequence alignment and database searches. Thus, BLOSUM62 seems best to represent the amino acid substitutions observed in all of the computed alignments between these two proteins. In another alignment of 1GKY and 2AK3-A using the Dayhoff PAM matrices instead of the BLOSUM matrices, the posterior probability distribution of the matrices shown in Figure 3.22 was found. Note that peaks are found at PAM110,

Figure 3.20. The Sankoff algorithm for finding the maximum number of identical residues in two sequences without scoring gaps. The example of two DNA sequences shown is taken from Sankoff (1972). A series of scoring matrices called V and W are made according to the matrix scoring scheme shown in parts *A*—*D*. In *A*, the algorithm first examines the maximum number of bases that can match. The scoring scheme used in this example is that a match between two bases is scored as 1 and a mismatch as 0. This number, 4, is shown in the lower right-hand corner of the matrix. To obtain this number, the method does not consider the number of gapped regions between each group of matched pairs, defined as an unconstrained set of matches by Sankoff. For example, a_1 can pair with b_3, and a_2 with b_4, to comprise a group of two sequential pairs, shown in bold. Then there is an unmatched region followed by a match of a_4 with b_6, unmatched base a_5, and finally a match between a_6 and b_7. Thus, two unmatched (gapped) regions will be included in this alignment. A second such set of matches that gives a maximum number of matches is shown as italicized positions. In this case, there is one unmatched region between the groups of matches. In *B–D*, a slightly different computational method is used to find the maximum possible number of matches given that there are zero gapped regions, one gapped region, two gapped regions, etc. In *B*, a matrix V_0, where subscript 0 indicates the number of gapped regions permitted, is first calculated. The bold and italicized positions indicate the scores found for the two groups of matches. To simplify the calculation of higher-level V matrices (V_1, V_2, etc.), another set of matrices (W_1, W_2, etc.) is also calculated. In *C*, the calculation of W_0 is shown. Using the scores calculated in W_0, matrix position and the algorithm shown in D, V_1 is then produced. V_1 shows the same combinations of matches found in the

A. *W* matrix

i \ j	b 0	C 1	C 2	A 3	G 4	T 5	C 6	T 7
a 0	0	0	0	0	0	0	0	0
A 1	0	0	0	**1**	1	1	1	1
G 2	0	0	0	0	**2**	2	2	2
C 3	0	*1*	1	1	2	2	3	3
C 4	0	0	*2*	2	2	2	**3**	3
A 5	0	1	2	*3*	3	3	3	3
T 6	0	0	1	2	3	*4*	4	**4**

$$W(i,j) = \max \begin{cases} W(i-1,j), \\ W(i,j-1), \\ W(i-1,j-1) + s(a_i, b_j) \end{cases}$$

where $s(a_i, b_j)$ is score of match of a_i with b_j.

B. V_0 matrix

i \ j	b 0	C 1	C 2	A 3	G 4	T 5	C 6	T 7
a 0	0	0	0	0	0	0	0	0
A 1	0	0	0	**1**	0	0	0	0
G 2	0	0	0	0	**2**	0	0	0
C 3	0	*1*	1	0	0	2	1	0
C 4	0	1	*2*	1	0	1	**3**	3
A 5	0	1	1	*3*	1	0	0	3
T 6	0	0	0	1	3	2	0	1

$$V_0(i,j) = V_0(i-1,j-1) + s(a_i, b_j)$$

C. W_0 matrix

i \ j	b 0	C 1	C 2	A 3	G 4	T 5	C 6	T 7
a 0	0	0	0	0	0	0	0	0
A 1	0	0	0	1	1	1	1	1
G 2	0	0	0	1	2	2	2	2
C 3	0	1	1	1	2	2	2	2
C 4	0	1	2	2	2	2	3	3
A 5	0	1	2	3	3	3	3	3
T 6	0	1	2	3	3	3	3	3

$$W_0(i,j) = \max \begin{cases} W_0(i-1,j), \\ V_0(i,j), \\ W_0(i,j-1) \end{cases}$$

where $V_0(i, j)$ is from the V_0 matrix in part B.

D. V_1 matrix

i \ j	b 0	C 1	C 2	A 3	G 4	T 5	C 6	T 7
a 0	0	0	0	0	0	0	0	0
A 1	0	0	0	1	0	0	0	0
G 2	0	0	0	0	**2**	1	1	1
C 3	0	*1*	1	0	1	2	3	2
C 4	0	1	*2*	1	1	2	**3**	3
A 5	0	0	1	*3*	2	2	2	3
T 6	0	0	1	2	3	*4*	3	**4**

$$V_1(i,j) = \max \begin{cases} V_1(i-1,j-1), \\ W_0(i-1,j-1) \\ \quad + s(a_i, b_j) \end{cases}$$

where $W_0(i, j)$ are obtained from the W_0 matrix in part C.

---

unconstrained case in A, and, therefore, no further calculation of matrices is necessary. In other cases, q V and W matrices will be calculated so that alignments with an increased number of unmatched or gapped regions may be found according to the formulas:

$$W_q(i,j) = \max \begin{cases} W_q(i-1,j), \\ V_q(i,j), \\ W_q(i,j-1) \end{cases}$$

$$V_q(i,j) = \max \begin{cases} V_q(i-1,j-1), \\ W_{q-1}(i-1,j-1) \\ \quad + s(a_i, b_j) \end{cases}$$

The number of computational steps required is equal to the product of the sequence lengths times the number of cycles needed to reach the unconstrained alignment, as shown in the right-hand corner of the matrix (A). The method may also be used for aligning protein sequences (Zhu et al. 1998) that are distantly related, as described below.

Table 3.13. *Posterior probability distribution of BLOSUM scoring matrices for alignment of 1GKY and 2AK3-A*

Matrix	Posterior probability
BLOSUM30	0.0257
BLOSUM35	0.0449
BLOSUM40	0.0825
BLOSUM45	0.1115
BLOSUM50	0.1755
BLOSUM62	0.2867
BLOSUM80	0.2350
BLOSUM100	0.0382

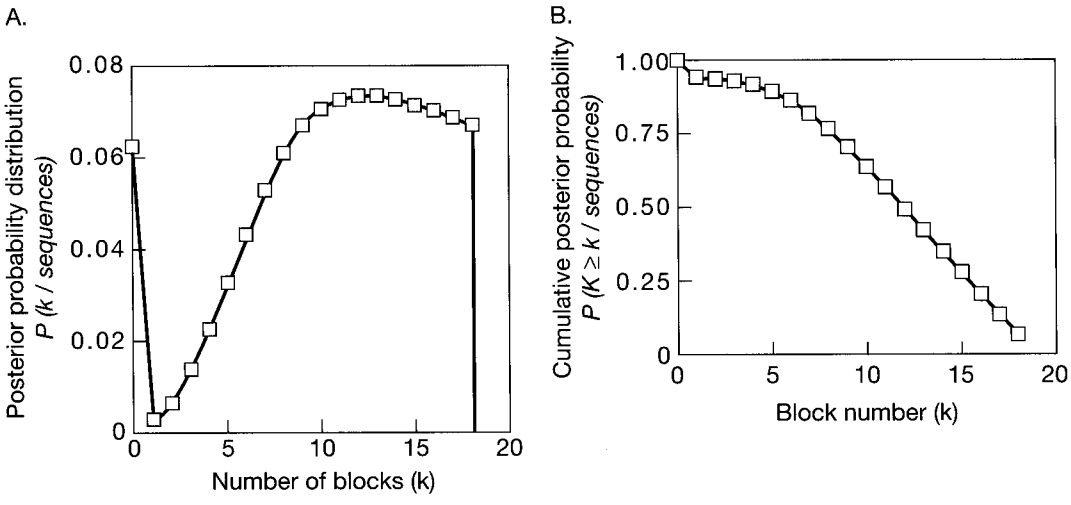


Figure 3.21. Posterior probability distribution of number of blocks from alignment of 1GKY and 2AK3-chain A by the Bayes block aligner (analysis of Zhu et al. 1998). *(A)* Posterior probability distribution of the block number, *k*. *(B)* Cumulative posterior probability distribution. This distribution shows the probability of a block number *K* greater than or equal to the value *k*. Values are derived from the probability distribution of *k* given in *A*. For example, $P(k \geq 1) = P(k \geq 0) - P(k=0) = 1 - 0.062 = 0.938$.

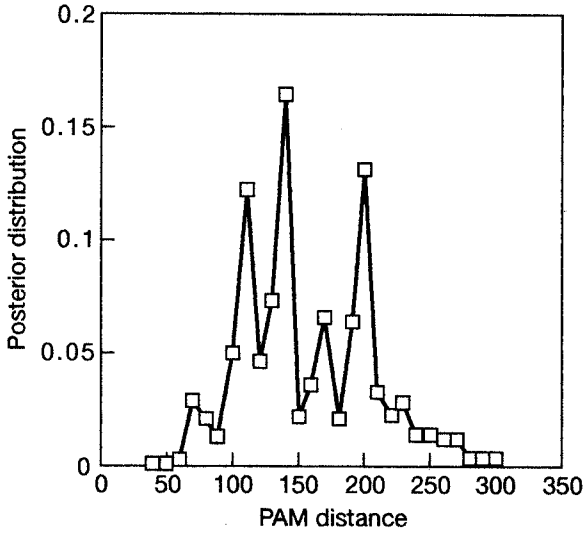


Figure 3.22. Posterior probability distribution of Dayhoff PAM scoring matrices for alignment of 1GKY and 2AK3-A.

140, and 200, thereby suggesting that substitution matrices for different evolutionary distances reflect the observed substitutions in different block alignments. The lower PAM matrix may be recognizing a more conserved domain, for example. This interesting observation implies that the alignment blocks found may be separated by different evolutionary distances, or at least may have undergone increased mutational variation. Thus, this type of analysis can provide information as to the evolutionary history of genes, including the possible involvement of duplications, rearrangements, and genetic events producing chimeras.

Another type of analysis that can be performed with the Bayes block aligner is to examine the probability of the alignments. The procedure is entirely different from other methods of sequence alignment such as dynamic programming. On the one hand, with dynamic programming methodology, a single best alignment is found for a given scoring matrix and gap penalty, and the odds for finding as good a score between random sequences of the same length and complexity is determined. On the other hand, with Bayesian alignment methods, all possible alignments are considered for a reasonable number of blocks and a set of substitution matrices. Rather than a probability of a single alignment, the probabilities of many alignments are provided. Many possible alignments may be examined and compared, and the frequency of certain residues in the sequences in these alignments may be determined.

For 1GKY and 2AK3-A, no highly probable single optimal or near-optimal alignment is found, suggesting these alignments are not representative of the best possible alignment of these sequences. Experience with the method has suggested that a minimum number of blocks that best represents the expected domain structure is the best approach. An average

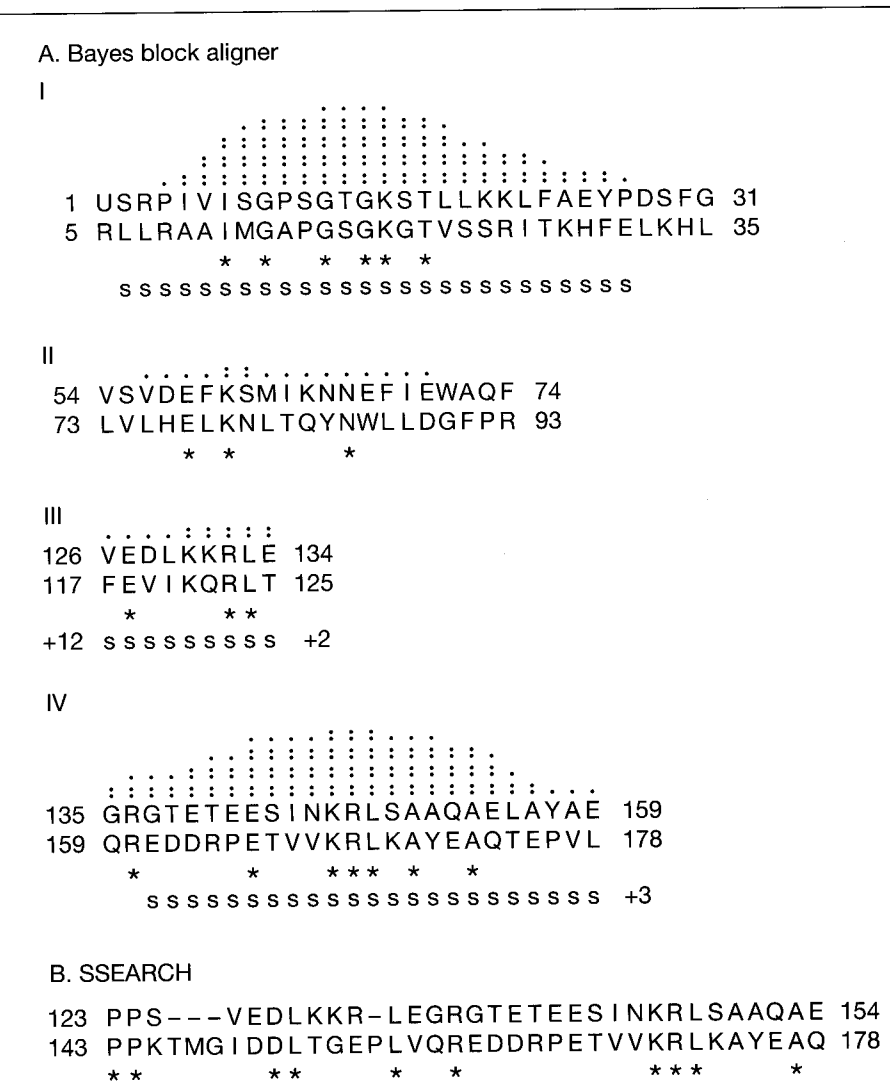


Figure 3.23. The alignment of 1GKY and 2AK3-A obtained with the Bayes aligner (*A*) and by SSEARCH (*B*), a dynamic programming method that provides local alignments (from Zhu et al. 1998). The highest-scoring sequence positions in the marginal posterior alignment distribution for the sequences for a block number of probability greater than 0.9 and the BLOSUM substitution matrices were successively sampled, and are shown in *A*. Neighboring aligned positions with scores greater than 0.25 of the peak value were included. Dots above the sequences indicate the relative probability of the aligned sequence positions. Asterisks are placed to highlight sequence identities. There is a clear correlation between the number of identities and the posterior probabilities. Alignment positions marked with an '*s*' were also identified by structural alignment using the program VAST (see Chapter 9). In regions III and IV, longer aligned regions were found by VAST than by the Bayes aligner. Three other regions identified by VAST of lengths 7, 7, and 8, two of which include 1–2 identities, were not reported by the Bayes aligner. In *B*, a local alignment of the sequences with SSEARCH is shown. The alignment parameters (BLOSUM50 substitution table and scoring penalties of −12,−2) are optimized for superfamily and family alignments. The center and right end of the alignment shown are approximately the same as that of alignment IV, but gaps are incorrectly predicted in the left end.

alignment for a number of blocks of probability greater than 0.9 has been found to give good agreement with predicted structural alignments. Values of k are obtained from the probability distribution for k such as in Figure 3.21. Using this approach with the Bayes aligner, the alignments between 1GKY and 2AK3-A shown in Figure 3.23 have been predicted. Although most of the predicted alignments correspond to expected structural alignments with the active site of the enzyme, alignment II does not so correspond (Fig. 3.24). Such false-negative predictions of structural alignments are the commonest error of Bayesian methods, probably because of relaxed conditions for scoring alignments in the

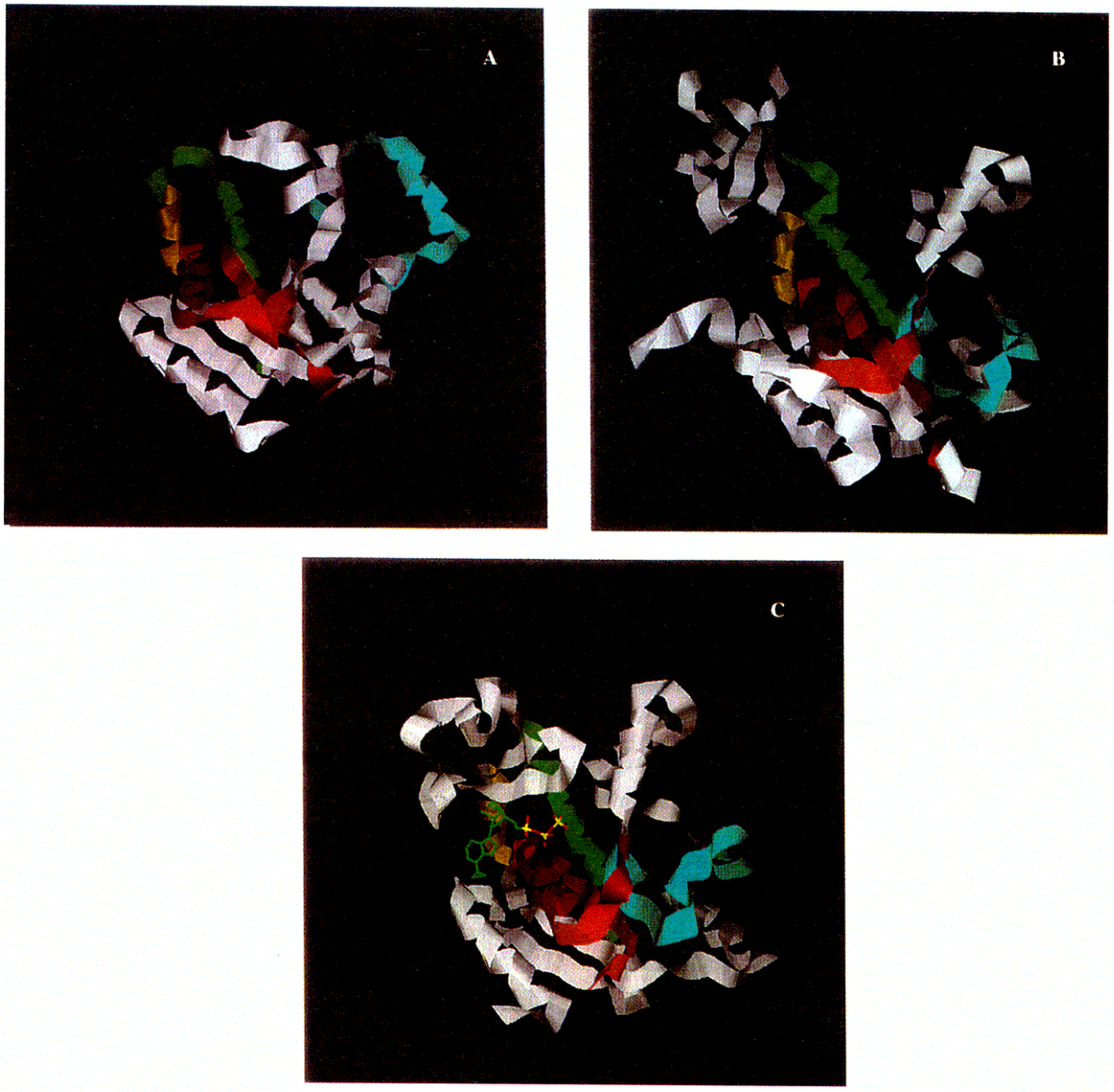


Figure 3.24. The positions of the alignments predicted by the Bayes block aligner. Predicted alignment I is shown in *red*, II in *cyan*, III in *orange*, and IV in *green*. (*A*) 1GKY, (B) 2AK3-A, and (*C*) 2AKY, which is similar to 2AK3-A. 2AKY is cocrystallized with an ATP analog. I, III, and IV may be structurally superimposed, but not II. (Reprinted, with permission, from Zhu et al. 1998 [copyright Oxford University Press].)

use of unconstrained prior information (Zhu et al. 1998). For these proteins, which share little sequence identity, the Bayes aligner correctly predicts many, but not all, features of the structural alignment, and does so better than a dynamic programming method that provides local alignments. In other cases, the Bayes aligner may not perform as well as dynamic programming. The prudent choice is to use the Bayes aligner as one of several computer tools for aligning sequences.

REFERENCES

Abagyan R.A. and Batalov S. 1997. Do aligned sequences share the same fold? *J. Mol. Biol.* **273:** 355–368.

Agarwal P. and States D.J. 1996. A Bayesian evolutionary distance for parametrically aligned sequences. *J. Comput. Biol.* **3:** 1–17.

———. 1998. Comparative accuracy of methods for protein sequence similarity search. *Bioinformatics* **14:** 40–47.

Altschul S.F. 1991. Amino acid substitution matrices from an information theoretic perspective. *J. Mol. Biol.* **219:** 555–565.

———. 1993. A protein alignment scoring system sensitive to all evolutionary distances. *J. Mol. Evol.* **36:** 290–300.

Altschul S.F. and Erickson B.W. 1986. A nonlinear measure of subalignment similarity and its significance levels. *Bull. Math. Biol.* **48:** 617–632.

Altschul S.F. and Gish G. 1996. Local alignment statistics. *Methods Enzymol.* **266:** 460–480.

Altschul S.F., Boguski M.S., Gish W., and Wootton J.C. 1994. Issues in searching molecular databases. *Nat. Genet.* **6:** 119–129.

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.

Argos P. 1987. A sensitive procedure to compare amino acid sequences. *J. Mol. Biol.* **193:** 385–396.

Arratia R. and Waterman M.S. 1989. The Erdös-Rényi strong law for pattern matching with a given proportion of mismatches. *Ann. Probab.* **17:** 1152–1169.

Arratia R., Gordon L., and Waterman M. 1986. An extreme value theory for sequence matching. *Ann. Stat.* **14:** 971–993.

———. 1990. The Erdös-Rényi law in distribution, for coin tossing and sequence matching. *Ann. Stat.* **18:** 539–570.

Bairoch A. 1991. PROSITE: A dictionary of sites and patterns in proteins. *Nucleic Acids Res.* **19:** 2241–2245.

Benner S.A., Cohen M.A., and Gonnet G.H. 1994. Amino acid substitution during functionally constrained divergent evolution of protein sequences. *Protein Eng.* **7:** 1323–1332.

Branden C. and Tooze J. 1991. *Introduction to protein structure.* Garland Publishing, New York.

Brenner S.E., Chothia C., and Hubbard T. 1998. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci.* **95:** 6073–6078.

Chao K.-M., Hardison R.C., and Miller W. 1994. Recent developments in linear-space alignment methods: A survey. *J. Comput. Biol.* **1:** 271–291.

Chvátal V. and Sankoff D. 1975. Longest common subsequences of two random sequences. *J. Appl. Probab.* **12:** 306–315.

Collins J.F., Coulson A.F., and Lyall A. 1988. The significance of protein sequence similarities. *Comput. Appl. Biosci.* **4:** 67–71.

Dayhoff M.O. 1978. Survey of new data and computer methods of analysis. In *Atlas of protein sequence and structure,* vol. 5, suppl. 3. National Biomedical Research Foundation, Georgetown University, Washington, D.C.

Dayhoff M.O., Barker W.C., and Hunt L.T. 1983. Establishing homologies in protein sequences. *Methods Enzymol.* **91:** 524–545.

Doolittle R.F. 1981. Similar amino acid sequences: Chance or common ancestry. *Science* **214:** 149–159.

———. 1986. *Of URFs and ORFs: A primer on how to analyze derived amino acid sequences.* University Science Books, Mill Valley, California.

Durbin R., Eddy S., Krogh A., and Mitchison G. 1998. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids.* Cambridge University Press, United Kingdom.

Feng D.F., Johnson M.S., and Doolittle R.F. 1985. Aligning amino acid sequences: Comparison of commonly used methods. *J. Mol. Evol.* **21:** 112–125.

Fitch W.M. 1966. An improved method of testing for evolutionary homology. *J. Mol. Biol.* **16:** 9–16.

———. 1970. Distinguishing homologous from analogous proteins. *Syst. Zool.* **19:** 99–113.

Fitch W.M. and Markowitz E. 1970. An improved method for determining codon variability in a gene and its application to the rate of fixation of mutations in evolution. *Biochem. Genet.* **4:** 579–593.

Fitch W.M. and Smith T.F. 1983. Optimal sequences alignments. *Proc. Natl. Acad. Sci.* **80:** 1382–1386.

George D.G., Barker W.C., and Hunt L.T. 1990. Mutation data matrix and its uses. *Methods Enzymol.* **183:** 333–351.

Gibbs A.J. and McIntyre G.A. 1970. The diagram, a method for comparing sequences. Its use with amino acid and nucleotide sequences. *Eur. J. Biochem.* **16:** 1–11.

Gonnet G.H., Cohen M.A., and Benner S.A. 1992. Exhaustive matching of the entire protein sequence database. *Science* **256:** 1443–1445.

———. 1994. Analysis of amino acid substitution during divergent evolution: The 400 by 400 dipeptide substitution matrix. *Biochem. Biophys. Res. Commun.* **199:** 489–496.

Gotoh O. 1982. An improved algorithm for matching biological sequences. *J. Mol. Biol.* **162:** 705–708.

Gray G.S. and Fitch W.M. 1983. Evolution of antibiotic resistance genes: The DNA sequence of a kanamycin resistance gene from *Staphylococcus aureus. Mol. Biol. Evol.* **1:** 57–66.

Gribskov M. and Burgess R.R. 1986. Sigma factors from *E. coli, B. subtilis,* phage SP01, and phage T4 are homologous proteins. *Nucleic Acids Res.* **14:** 6745–6763.

Gumbel E.J. 1962. Statistical theory of extreme values (main results). In *Contributions to order statistics* (ed A.E. Sarhan and B.G. Greenberg), chap. 6, p. 71. Wiley, New York.

Gusfield D. and Stelling P. 1996. Parametric and inverse-parametric sequence alignment with XPARAL. *Methods Enzymol.* **266:** 481–494.

Henikoff S. and Henikoff J.G. 1991. Automated assembly of protein blocks for database searching. *Nucleic Acids Res.* **19:** 6565–6572.

———. 1992. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci.* **89:** 10915–10919.

———. 1993. Performance evaluation of amino acid substitution matrices. *Proteins Struct. Funct. Genet.* **17:** 49–61.

Henikoff S., Greene E.A., Pietrokovski S., Bork P., Attwood T.K., and Hood L. 1997. Gene families: The taxonomy of protein paralogs and chimeras. *Science* **278:** 609–614.

Huang X. 1994. On global sequence alignment. *Comput. Appl. Biosci.* **10:** 227–235.

Huang X. and Miller W. 1991. A time-efficient, linear-space local similarity algorithm. *Adv. Appl. Math.* **12:** 337–357.

Huang X., Hardison R.C., and Miller W. 1990. A space-efficient algorithm for local similarities. *Comput. Appl. Biosci.* **6:** 373–381.

Johnson M.S. and Overington J.P. 1993. A structural basis for sequence comparisons: An evaluation of scoring methodologies. *J. Mol. Biol.* **233:** 716–738.

Jones D.T., Taylor W.R., and Thornton J.M. 1992. The rapid generation of mutation data matrices from protein sequences. *Comput. Appl. Biosci.* **8:** 275–282.

———. 1994. A mutation data matrix for transmembrane proteins. *FEBS Lett.* **339:** 269–275.

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.* **87:** 2264–2268.

———. 1993. Applications and statistics for multiple high-scoring segments in molecular sequences. *Proc. Natl. Acad. Sci.* **90:** 5873–5877.

Karlin S., Bucher P., and Brendel P. 1991. Statistical methods and insights for protein and DNA sequences. *Annu. Rev. Biophys. Biophys. Chem.* **20:** 175–203.

Kidwell M.G. 1983. Evolution of hybrid dysgenesis determinants in *Drosophila melanogaster. Proc. Natl. Acad. Sci.* **80:** 1655–1659.

Lawrence J.G. and Ochman H. 1997. Amelioration of bacterial genomes: Rates of change and exchange. *J. Mol. Biol.* **44:** 383–397.

Li W. and Graur D. 1991. *Fundamentals of molecular evolution.* Sinauer Associates, Sunderland, Massachusetts.

Lipman D.J., Wilbur W.J., Smith T.F., and Waterman M.S. 1984. On the statistical significance of nucleic acid similarities. *Nucleic Acids Res.* **12:** 215–226.

Maizel J.V., Jr. and Lenk R.P. 1981. Enhanced graphic matrix analysis of nucleic acid and protein sequences. *Proc. Natl. Acad. Sci.* **78:** 7665–7669.

Miller W. and Myers E.W. 1988. Sequence comparison with concave weighting functions. *Bull. Math. Biol.* **50:** 97-120.

Miyamoto M.M. and Fitch W.M. 1995. Testing the covarion hypothesis of evolution. *Mol. Biol. Evol.* **12:** 503–513.

Mott R. 1992. Maximum-likelihood estimation of the statistical distribution of Smith-Waterman local sequence similarity scores. *Bull. Math. Biol.* **54:** 59–75.

Myers E.W. and Miller W. 1988. Optimal alignments in linear space. *Comput. Appl. Biosci.* **4:** 11–17.

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

Pearson W.R. 1990. Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol.* **183:** 63–98.

———. 1995. Comparison of methods for searching protein sequence databases. *Protein Sci.* **4:** 1150–1160.

———. 1996. Effective protein sequence comparison. *Methods Enzymol.* **266:** 227–258.

———. 1998. Empirical statistical estimates for sequence similarity searches. *J. Mol. Biol.* **276:** 71–84.

Pearson W.R. and Miller W. 1992. Dynamic programming algorithm for biological sequence comparison. *Methods Enzymol.* **210:** 575–601.

Rechid R., Vingron M., and Argos P. 1989. A new interactive protein sequence alignment program and comparison of its results with widely used programs. *Comput. Appl. Biosci.* **5:** 107–113.

Risler J.L., Delorme M.O., Delacroix H., and Henaut A. 1988. Amino acid substitutions in structurally related proteins: A pattern recognition approach. *J. Mol. Biol.* **204:** 1019–1029.

Sander C. and Schneider R. 1991. Database of homology derived protein structures and the structural meaning of sequence alignment. *Proteins* **9:** 56–68.

Sankoff D. 1972. Matching sequences under deletion/insertion constraints. *Proc. Natl. Acad. Sci.* **69:** 4–6.

Schwartz S., Miller W., Yang C.-M., and Hardison R.C. 1991. Software tools for analyzing pairwise alignments of long sequences. *Nucleic Acids Res.* **19:** 4663–4667.

Sellers P.H. 1974. On the theory and computation of evolutionary distances. *SIAM J. Appl. Math.* **26:** 787–793.

———. 1980. The theory and computation of evolutionary distances: Pattern recognition. *J. Algorithms* **1:** 359–373.

Smith H.O., Annau T.M., and Chandrasegaran S. 1990. Finding sequence motifs in groups of functionally related proteins. *Proc. Natl. Acad. Sci.* **87:** 826–830.

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

———. 1981b. Comparison of biosequences. *Adv. Appl. Math.* **2:** 482–489.

Smith T.F., Waterman M.S., and Burks C. 1985. The statistical distribution of nucleic acid similarities. *Nucleic Acids Res.* **13:** 645–656.

Smith T.F., Waterman M.S., and Fitch W.M. 1981. Comparative biosequence metrics. *J. Mol. Evol.* **18:** 38–46.

Sonnhammer E.L. and Durbin R. 1995. A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis. *Gene* **167:** GC1–10.

States D.J. and Boguski M.S. 1991. Similarity and homology. In *Sequence analysis primer* (ed. M. Gribskov and J. Devereux), pp. 92–124. Stockton Press, New York.

States D.J., Gish W., and Altschul S.F. 1991. Improved sensitivity of nucleic acid database searches using application-specific scoring matrices. *Methods* **3:** 66–70.

Tatusov R.L., Koonin E.V., and Lipman D.J. 1997. A genomic perspective on protein families. *Science* **278:** 631–637.

Vingron M. and Waterman M.S. 1994. Sequence alignment and penalty choice: Review of concepts, case studies and implications. *J. Mol. Biol.* **235:** 1–12.

Vogt G., Etzold T., and Argos P. 1995. An assessment of amino acid exchange matrices: The twilight zone re-visited. *J. Mol. Biol.* **249:** 816–831.

Waterman M.S., Ed. 1989. Sequence alignments. In *Mathematical methods for DNA sequences.* CRC Press, Boca Raton, Florida.

———. 1994. Parametric and ensemble sequence alignment algorithms. *Bull. Math. Biol.* **56:** 743–767.

Waterman M.S. and Eggert M. 1987. A new algorithm for best subsequence alignments with application to tRNA-tRNA comparisons. *J. Mol. Biol.* **197:** 723–728.

Waterman M.S. and Vingron M. 1994a. Rapid and accurate estimates of statistical significance for sequence database searches. *Proc. Natl. Acad. Sci.* **91:** 4625–4628.

———. 1994b. Sequence comparison significance and Poisson distribution. *Stat. Sci.* **9:** 367–381.

Waterman M.S., Eggert M., and Lander E. 1992. Parametric sequence comparisons. *Proc. Natl. Acad. Sci.* **89:** 6090–6093.

Waterman M.S., Gordon L., and Arratia R. 1987. Phase transitions in sequence matches and nucleic acid structure. *Proc. Natl. Acad. Sci.* **84:** 1239–1243.

Waterman M.S., Smith T.F., and Beyer W.A. 1976. Some biological sequence metrics. *Adv. Math.* **20:** 367–387.

Wilbur W.J. 1985. On the PAM model of protein evolution. *Mol. Biol. Evol.* **2:** 434–447.

Zhu J., Liu J.S., and Lawrence C.E. 1998. Bayesian adaptive sequence alignment algorithms. *Bioinformatics* **14:** 25–39.