For the case $n = 259.0675$,

$$m_{jj}^{(259.0675)} = 0.111251, \quad m_{jk}^{(259.0675)} = 0.046776 \quad j \neq k. \tag{6.33}$$

The definition (6.25) of $q(j,k)$ implies that for this case

$$q(j,j) = 0.0055625, \quad q(j,k) = 0.0023388, \quad j \neq k. \tag{6.34}$$

With these values the probability $20q(j,j)$ of a match at any position is 0.111251, and the probability of a mismatch is 0.888749. Taking the value $C = 1$ in (6.32), the mean score in the substitution matrix is then

$$12(0.111251) - 0.888749 = 0.446. \tag{6.35}$$

We discuss this example further in Section 9.3.3.

6.6 Multiple Sequences

We may have a set of more than two related sequences, all descended from a common ancestor, and in such a case it is often desirable to put them into a multiple alignment. The definition of a multiple alignment is a straightforward generalization of a pairwise alignment. Similarly, the dynamic programming algorithms for constructing the alignments generalize in a straightforward manner. For a small number of sequences, usually fewer than 20, this works well. Problems arise, however, in applying these algorithms to many sequences. The running time for using dynamic programming to do a global multiple alignment of n sequences each of length approximately L is $O((2L)^n)$. For local alignments the situation becomes even worse.

Some algorithms have been developed to find high-scoring alignments quickly, without, however, guaranteeing to find one with the highest score. Perhaps one of the algorithms most commonly used for multiple global alignments is called CLUSTAL W (Thompson et al. (1994)).

In this section we will describe a statistical method for finding ungapped local alignments, introduced by Lawrence et al. (1993). A more general algorithm allowing gapped alignments is given by Zhu et al. (1998). In Section 11.3.2 we give a different method, which constructs gapped multiple alignments.

We describe the process in terms of protein sequences. Label the amino acids in some agreed order as amino acids 1, 2, ..., 20, and suppose that these have respective "background" frequencies $p_1, p_2, \ldots, p_{20}$. Given N protein sequences, of respective lengths $L_1, L_2, \ldots, L_N$, the aim is to find N segments of length W, one in each sequence, that in some sense are most similar to each other. Here the value of W is some chosen fixed number; the choice of W is discussed below. There are $S = \prod_{j=1}^{N}(L_j - W + 1)$ possible

choices for the respective locations of these N segments in the N respective sequences, and it is assumed that N and the L_j are so large that a purely algorithmic approach to finding the most similar segments by an extension of the methods discussed in Section 6.4.2 is not computationally feasible.

We describe the Lawrence et al. approach in Markov chain terms. Consider a Markov chain with S "states," each state corresponding to a choice of the locations of the N segments in the N sequences. Each state of the Markov chain is an aligned array of amino acids. This array has N rows and W columns. The aim is to find that array in which the various rows in some sense best align with each other.

The procedure consists of repeated iteration of a basic step, each step consisting of a move from one state of the Markov chain to another, that is, from one array to another. The initial array can be chosen arbitrarily or on the basis of some biological knowledge as an initial guess of the best alignment. In each step one of the various protein sequences is chosen, and the row in the array corresponding to that sequence is allowed to change in a way described below. It is convenient here to assume that the choice of this sequence is made randomly.

Before describing the way in which changes in the array are made, we illustrate in Figure 6.4 the result of the changes made in two consecutive steps of the procedure. In step 1 the third sequence happened to be chosen, so that the third row in the array was allowed to change, and in step 2 the first sequence happened to be chosen, so that the first row was allowed to change.

position								position								position							
1	2	3	4	$\cdots$	W			1	2	3	4	$\cdots$	W			1	2	3	4	$\cdots$	W		
V	Q	A	L	$\cdots$	N			V	Q	A	L	$\cdots$	N			C	A	A	N	$\cdots$	R		
A	Q	B	N	$\cdots$	R			A	Q	B	N	$\cdots$	R			A	Q	B	N	$\cdots$	R		
L	L	C	R	$\cdots$	N	step 1		C	Q	T	N	$\cdots$	N	step 2		C	Q	T	N	$\cdots$	N		
W	R	A	A	$\cdots$	C	$\longrightarrow$		W	R	A	A	$\cdots$	C	$\longrightarrow$		W	R	A	A	$\cdots$	C		
S	Q	C	C	$\cdots$	T			S	Q	C	C	$\cdots$	T			A	Q	C	C	$\cdots$	T		
S	Q	T	R	$\cdots$	C			S	Q	T	R	$\cdots$	C			S	Q	T	R	$\cdots$	C		
			$\vdots$								$\vdots$								$\vdots$				
G	M	C	R	$\cdots$	T			G	M	C	R	$\cdots$	T			G	M	C	R	$\cdots$	T		

FIGURE 6.4. Full arrays of aligned segments before steps 1, 2, and 3.

We call the array just before any step is taken the "original" array and the array following this step the "new" array. This new array is also the original array for the next step. We now consider the first step in detail.

The segments in all sequences other than randomly chosen sequence 3 define a reduced array of $N - 1$ segments consisting of the original array

without row 3. This reduced array is the leftmost array in Figure 6.5. The

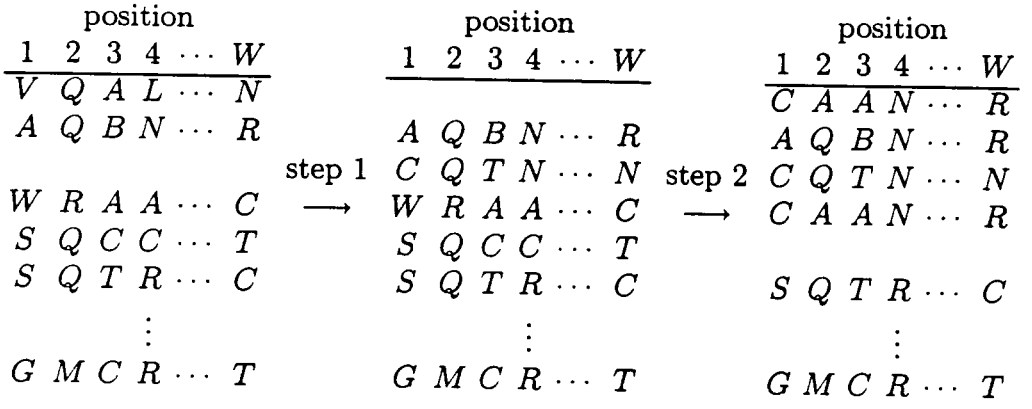

FIGURE 6.5. Partial arrays of aligned segments before steps 1, 2, and 3.

reduced arrays in this figure show that sequence 3 was chosen to change in step 1 and that sequence 1 was chosen to change in step 2, and also shows that sequence 5 was chosen to change in step 3.

Suppose that in the first reduced array, amino acid j ($j = 1, 2, \ldots, 20$) occurs $c_{i,j}$ times in the ith column. From the $c_{i,j}$ values a probability estimate q_{ij} is calculated, defined by

$$q_{ij} = \frac{c_{ij} + b_j}{N - 1 + B}. \tag{6.36}$$

Here the b_j are pseudocounts, as defined in Section 3.7, and $B = \sum_j b_j$. The reason for the introduction of pseudocounts, and the actual choices of the b_j, will be discussed below. For the moment we take the b_j as given.

The aim of the first step is to replace the original segment in the third row by a new segment in a way that tends to increase the overall alignment of the N resulting segments in the new array. There are $L_3 - W$ segments of length W in sequence 3. We call that segment starting in position x in sequence 3 "segment x." Suppose that the amino acids in this segment are $x_1, x_2, \ldots, x_W$. The probability $P_{\boldsymbol{x}}$ of this ordered set of amino acids under the population amino acid frequencies is $P_{\boldsymbol{x}} = p_{x_1} p_{x_2} \cdots p_{x_W}$. The estimated probability $Q_{\boldsymbol{x}}$ of this ordered set of amino acids using the $N-1$ segments in the first reduced array in Figure 6.5 is taken as

$$Q_{\boldsymbol{x}} = q_{1,x_1} q_{2,x_2} \cdots q_{W,x_W}.$$

The likelihood ratio $\mathrm{LR}(x)$ is defined as $Q_{\boldsymbol{x}}/P_{\boldsymbol{x}}$. The numerator may be thought of as the probability of the sequence under the model reflecting only the sequences in the current alignment, while the denominator is the probability of the sequence under background frequencies. The final operation in step 1 is to replace the segment in row 3 of the original array by

segment x with probability

$$\frac{\mathrm{LR}(x)}{\sum_{m=1}^{L_3-W}\mathrm{LR}(m)}. \tag{6.37}$$

The reason for this choice will be discussed in Section 10.5.2. The result of this step is to produce a new array (as illustrated in the second array in Figure 6.4). The replacement procedure tends to replace the original segment in row 3 by a new segment more closely aligned with the remaining segments.

In the following step this procedure is repeated, with (in Figure 6.4) the segment from sequence 1 being randomly chosen to change. In the next step the segment from sequence 5 was randomly chosen to change, and so on. As one step follows another the N segments in the array tend to become more similar to each other, or in other words, to align better. This iterative procedure visits various possible alignments according to a random process, and thus does not systematically approach the "best" alignment. However, after many steps the best alignment should tend to arise more and more frequently and hence be recognized.

This procedure is essentially one of Gibbs sampling, and we further consider its properties in the discussion of the Gibbs sampling procedure in Section 10.5.2. In order to introduce the analysis of Section 10.5.2, we adopt here a general notation suitable for the discussion of that section. Suppose that before some specific step is taken, the current array is array s in the collection of S possible arrays, and that after this step is taken the new array is array u. Arrays s and u are identical in all rows other than the one that is changed during this step. Thus the reduced arrays obtained by eliminating the row in which arrays s and u differ lead to identical values of the q_{ij}.

Now consider the amino acids in arrays s and u in the row in which they differ. Suppose that in array s these are denoted by $s(1)$, $s(2), \ldots,$ $s(W)$ and in array u are denoted by $u(1)$, $u(2), \ldots,$ $u(W)$. If the transition probability from array s to array u is denoted by p_{su}, then expression (6.37) shows that

$$\frac{p_{su}}{p_{us}} = \frac{q_{1,u(1)}q_{2,u(2)}\cdots q_{W,u(W)}}{q_{1,s(1)}q_{2,s(2)}\cdots q_{W,s(W)}} \cdot \frac{p_{s(1)}p_{s(2)}\cdots p_{s(W)}}{p_{u(1)}p_{u(2)}\cdots p_{u(W)}}. \tag{6.38}$$

We return to this ratio in Section 10.5.2.

The reason for using pseudocounts in the probability estimate (6.36) is the following. In step 1 above, there might be an excellent alignment of segment x of protein sequence 3 with the $N-1$ segments in the reduced array, except that the amino acid at position i in sequence x is not represented at position i in the reduced array. If pseudocounts were not used and the probability estimate q_{ij} replaced by $c_{ij}/(N-1)$, segment x could not be chosen at this step, since in this case the probability $c_{ij}/(N-1)$

would be 0 for this segment. We therefore assume that each b_j is greater than 0, and then introduction of pseudocounts allows the choice of segment x in the above case.

The choice of the pseudocounts b_j must be made subjectively. Lawrence et al. (1993) make the reasonable suggestion of choosing b_j proportional to the background frequency p_j of amino acid j. The choice of the proportionality constant is less obvious, and Lawrence et al. claim that in practice the constant $\sqrt{N}$ works well.

So far the length W of the segments considered has been assumed to be given. The choice of this length is discussed in detail by Lawrence et al. together with further tactical questions.

It might be asked why the choice of each new segment is random. An alternative procedure in the step described above would be to replace the current segment in sequence one by that segment maximizing the ratio in (6.37). The entire operation is then fully deterministic, and runs the risk of settling on a locally rather than a globally best alignment. The stochastic procedure allows movement from a locally best alignment to a globally best alignment. However, it is not guaranteed that the procedure will escape from the neighborhood of a locally best alignment in reasonable time, and thus may be sensitive to the choice of the initial alignment.

Problems

6.1. Prove that

$$\sum_{k=0}^{\min\{m,n\}} \binom{m}{k}\binom{n}{k} = \binom{m+n}{n} = \binom{m+n}{m}.$$

Hint: Think of having to select n objects from a set with $s = m+n$ objects, of which m are of one kind and n are of another kind.

6.2. The BLOSUM62 substitution matrix given in Table 6.7 is often used to assign a score to each pair of aligned amino acids. Use this matrix and a linear gap penalty of $d = 5$ to find all highest-scoring alignments between $\boldsymbol{x} = EATGHAG$ and $\boldsymbol{y} = EEAWHEAE$.

6.3. Suppose there are three symbols A, B, and C, and two blocks of data as given below. You are to construct the BLOSUM68 substitution matrix derived from these data, however in the step involving taking the log-ratios, multiply the logarithms by 8 instead of by 2 (before rounding). *Hint:* Of the six scores, one should be zero, two negative, and three positive.