BIOINFORMATICS

Vol. 15 nos 7/8 1999
Pages 563–577

Identifying DNA and protein patterns with statistically significant alignments of multiple sequences

Gerald Z. Hertz and Gary D. Stormo

Department of Molecular, Cellular and Developmental Biology, University of Colorado, Boulder, CO 80309-0347, USA

Received on December 7, 1998; revised and accepted on February 22, 1999

Abstract

Motivation: *Molecular biologists frequently can obtain interesting insight by aligning a set of related DNA, RNA or protein sequences. Such alignments can be used to determine either evolutionary or functional relationships. Our interest is in identifying functional relationships. Unless the sequences are very similar, it is necessary to have a specific strategy for measuring—or scoring—the relatedness of the aligned sequences. If the alignment is not known, one can be determined by finding an alignment that optimizes the scoring scheme.*

Results: *We describe four components to our approach for determining alignments of multiple sequences. First, we review a log-likelihood scoring scheme we call information content. Second, we describe two methods for estimating the P value of an individual information content score: (i) a method that combines a technique from large-deviation statistics with numerical calculations; (ii) a method that is exclusively numerical. Third, we describe how we count the number of possible alignments given the overall amount of sequence data. This count is multiplied by the P value to determine the expected frequency of an information content score and, thus, the statistical significance of the corresponding alignment. Statistical significance can be used to compare alignments having differing widths and containing differing numbers of sequences. Fourth, we describe a greedy algorithm for determining alignments of functionally related sequences. Finally, we test the accuracy of our P value calculations, and give an example of using our algorithm to identify binding sites for the Escherichia coli CRP protein.*

Availability: *Programs were developed under the UNIX operating system and are available by anonymous ftp from ftp://beagle.colorado.edu/pub/consensus.*

Contact: *hertz@colorado.edu*

Introduction

Functionally related DNA or protein sequences are generally expected to share some common sequence elements. For example, a DNA-binding protein is expected to bind related DNA sequences. The pattern shared by a set of functionally related sequences is commonly identified during the process of aligning the sequences to maximize sequence conservation.

Central to any alignment is the method being used to model the alignment. The goal of the model is to summarize the alignment so that the collection of sequences can be described more concisely than simply listing all the sequences. The simplest and oldest method for describing a sequence alignment is the consensus sequence, which contains the most highly conserved letter (i.e. base for DNA or amino acid for protein) at each position of the alignment. However, most alignments are not limited to just a single letter at each position. At some positions of an alignment, any letter may be permissible, although some letters may occur much more frequently than others.

If the sequences are assumed to be conserved because they have not had time to diverge completely since splitting from a common ancestor, then an alignment model incorporating an evolutionary tree is appropriate. Our interest is in sequences that are related because of their common function. Thus, we use a matrix model which does not include phylogenetic information. The simplest matrix model lists some measure of the desirability of each letter at each position of the alignment.

One of the simplest types of matrices is the *alignment matrix*, which lists the number of occurrences of each letter at each position of an alignment (e.g. Figure 1a). Another simple type of matrix is the *weight matrix*, whose elements are the weights used to score a test sequence to measure how close that sequence word matches the pattern described by the matrix (e.g. Figure 1b). A test sequence is aligned along the weight matrix, and its score is the sum of the weights for the letter aligned at each position. Weights can be derived from the alignment matrix (Staden, 1984; Hertz *et al.*, 1990; Tatusov *et al.*, 1994) or determined experimentally (Stormo *et al.*, 1986; Fields *et al.*, 1997).

Matrices can also describe more complex patterns that contain gaps (i.e. sequences contain insertions and deletions relative to each other) or in which different positions are

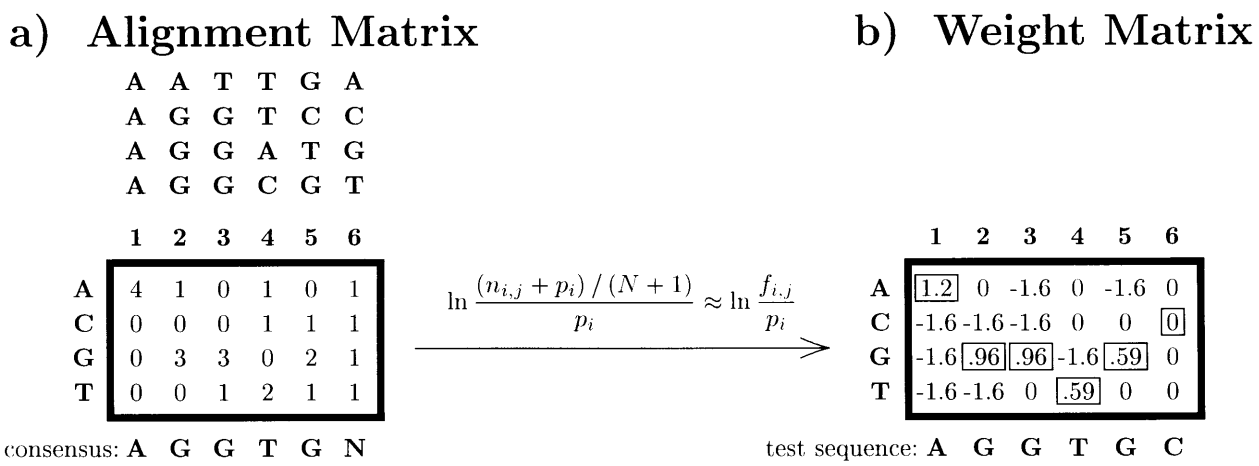

Fig. 1. Examples of the simple matrix model for summarizing a DNA alignment. (**a**) An alignment matrix describing the alignment of the four 6-mers on top. The matrix contains the number of times, $n_{i,j}$, that letter *i* is observed at position *j* of this alignment. Below the matrix is the consensus sequence corresponding to the alignment (N indicates that there is no nucleotide preference). (**b**) A weight matrix derived from the alignment in (a). The formula used for transforming the alignment matrix to a weight matrix is shown above the arrow. In this formula, *N* is the total number of sequences (four in this example), p_i is the *a priori* probability of letter *i* (0.25 for all the bases in this example) and $f_{i,j} = n_{i,j}/N$ is the frequency of letter *i* at position *j*. The numbers enclosed in blocks are summed to give the overall score of the test sequence. The overall score is 4.3, which is also the maximum possible score with this weight matrix.

correlated with each other (Hertz and Stormo, 1995). However, here, we are only concerned with sequences that can be aligned without insertions and deletions. Furthermore, we assume that the positions of an alignment function independently according to whatever biochemical criteria are used to select the underlying, functionally related sequences. Thus, we will only be discussing the simplest matrix model, as illustrated in Figure 1.

A good alignment is assumed to be one whose alignment matrix is rarely expected to occur by chance. A standard statistic for scoring the relative likelihood of an alignment matrix is the log-likelihood ratio. We compare alignments using a variant of the log-likelihood ratio we call *information content* and determine alignments from functionally related, unaligned sequences using a greedy algorithm (Stormo and Hartzell, 1989; Hertz *et al*., 1990).

A limitation in the use of information content has been a lack of good estimates of the statistical significance of observing a specific information content. In this paper, we present an efficient method for calculating the *P* value of an information-content score. In our case, the *P* value is the probability of obtaining an information content greater than or equal to the observed value, given the number of sequences in the alignment and its width. This method combines numerical calculations with a technique from large-deviation statistics. We also present a slower, exclusively numerical method for calculating the *P* value.

Next, we describe how we estimate the number of possible alignments due to the amount of sequence data. This estimate is combined with those of *P* value to arrive at an expectation for observing a particular information content or greater. Finally, we describe the latest version of our greedy algorithm for aligning functionally related sequences. This algorithm has been substantially enhanced since its earliest version (Stormo and Hartzell, 1989; Hertz *et al*., 1990). We also present an example of using our algorithm and statistics to align DNA-binding sites of the *Escherichia coli* CRP protein.

The distinction between the alignment model and the alignment algorithm is important. For example, expectation maximization (Lawrence and Reilly, 1990) and Gibbs sampling (Lawrence *et al*., 1993) are alternative algorithms that have been used to align DNA and protein sequences. However, these alternative algorithms were used with alignment models and log-likelihood statistics intimately related to those used by us. Thus, our calculations of statistical significance are applicable to these other common alignment algorithms.

Information content of an alignment matrix

In our comparison of alignment matrices, we assume that the letters of a sequence are independent and identically distributed. Thus, the *a priori* probability of a sequence of letters is the product of the *a priori* probability of the individual letters. The *a priori* probability of the individual letters might be the overall frequency of the letters within all sequences of an organism (e.g. the genomic frequency of the nucleotide bases) or the frequency within a subset of sequences, such as

the frequency in the data set being aligned. Given the assumption that the distribution of letters is independent and identically distributed, the probability of an alignment matrix is determined by the multinomial distribution:

$$P_{\text{matrix}} = \prod_{j=1}^{L}\left[\frac{N!}{\Pi_{i=1}^{A} n_{i,j}!}\prod_{i=1}^{A} p_i^{n_{i,j}}\right] \quad (1)$$

where i refers to the rows of the matrix (e.g. the bases A, C, G, T for a DNA alignment), j refers to the columns of the matrix (i.e. the positions of the letters within the alignment pattern), A is the total number of letters in the sequence alphabet (four for DNA and 20 for protein), L is the total number of columns in the matrix (six in Figure 1), p_i is the *a priori* probability of letter i, $n_{i,j}$ is the occurrence of letter i at position j, and N is the total number of sequences in the alignment (four in Figure 1).

Our assumption is that the most interesting alignments are those whose letter frequencies most differ from the *a priori* probabilities of the letters. The most commonly used measures for scoring the divergence from the *a priori* probabilities of a set of letters are the χ^2 statistic and the log-likelihood ratio. In our work, we use statistics based on the log-likelihood ratio rather than the more *ad hoc* χ^2 statistic. The standard log-likelihood ratio statistic is

$$\text{log-likelihood ratio} = \sum_{j=1}^{L}\sum_{i=1}^{A} n_{i,j}\ln\frac{p_i}{f_{i,j}}$$

where $f_{i,j} = n_{i,j}/N$ is the frequency that letter i occurs at position j such that $\Sigma_{i=1}^{A} f_{i,j} = 1$. When the value of $f_{i,j}$ is close to p_i, –2 times the log-likelihood ratio is approximately equal to the χ^2 statistic. Under these conditions, this product will have a distribution approximated by the χ^2 distribution with $L(A-1)$ degrees of freedom [Wilks (1938) and discussed in many introductory statistics books].

The statistic we use is obtained by dividing the log-likelihood ratio by $-N$. We call this statistic the information content of the sequence alignment and abbreviate it as I_{seq}:

$$I_{\text{seq}} = \sum_{j=1}^{L}\sum_{i=1}^{A} f_{i,j}\ln\frac{f_{i,j}}{p_i} \quad (2)$$

This normalized log-likelihood ratio has gone by various other names according to the perspective of those who have used it. When motivated by information theory, this formula is called the Kullback–Leibler information (Kullback and Leibler, 1951) or relative entropy. When derived from large-deviation principles, it is called the *large-deviation rate function* (Bucklew, 1990).

I_{seq} is also related to thermodynamics. In particular, the information content of DNA sequences that are bound by a common protein has been related to the thermodynamics of the protein–DNA interaction. I_{seq} measures a relationship between the average ΔG of the protein binding a functional DNA site and the ΔG of the protein binding an arbitrary DNA sequence (Berg and von Hippel, 1987; Stormo, 1988; Stormo and Yoshioka, 1991; Fields *et al*., 1997; Stormo and Fields, 1998). Thus, I_{seq} is a measure of the discrimination between the binding of a functional DNA sequence and an arbitrary DNA sequence. The 'seq' subscript indicates that formula (2) is the information content derived from the statistical properties of a sequence alignment. In Fields *et al*. (1997), a closely related information content is discussed, I_{spec}, that is derived through thermodynamics.

Formula (2) has various properties that satisfy intuitive ideas of the information content of an alignment. Equation (2) is a measure of the distance from the center of the distribution where $f_{i,j} = p_i$. When $f_{i,j} = p_i$, the distance is at a minimum and equals zero. The distance is maximized when the least expected letter occurs exclusively, i.e. $f_{m,j} = 1$ and $p_m \leq p_i$ for all values of i. Schneider *et al*. (1986) noticed that $e^{-I_{\text{seq}}}$ is approximately equal to the frequency with which the binding sites for a DNA-binding protein occur within the *E.coli* genome. We (Hertz and Stormo, 1995) have since come up with a more precise description of this relationship: $e^{-I_{\text{seq}}}$ is an upper limit to the expected frequency with which the sequence words within an alignment occur in random sequences.

The *P* value of an information content

A statistic such as information content is not an end in itself. We ultimately wish to calculate the P value of the statistic, i.e. the probability of observing an alignment having the observed information content or greater, given the width of the alignment and the number of sequences in the alignment. As discussed in the previous section, we assume that the probability of a specific letter being observed at any position of a random sequence is equal to that letter's *a priori* probability and is independent of the occurrence of any other letter. Thus, the null model for the alignment matrix is that the distribution of letters in each alignment column is an independent multinomial distribution [formula (1)].

Under the above assumptions, when the information content is small and the number of sequences is large, $2NI_{\text{seq}}$ tends to a χ^2 distribution with $L(A-1)$ degrees of freedom since $-NI_{\text{seq}}$ is a log-likelihood ratio (discussed in many introductory statistics books). Unfortunately, our conditions generally involve very large scores and frequently few sequences; thus, the χ^2 distribution tends to give poor probability estimates.

However, we are able to obtain very accurate estimates of the P value using a technique from large-deviation statistics. Similar techniques have been used to determine the statistical significance of other types of biologically interesting sequence patterns (Karlin and Altschul, 1990; Dembo *et al*.,

1994). In the next section, we first give a general description of the technique adapted from the description in Bucklew (1990, Chapter VII) and then describe how we apply it to our particular problem. In the section following that description, we describe a numerical method for determining the *P* value. In the Results, we test the accuracy of the large-deviation method for determining *P* value against that obtained with the numerical method.

A large-deviation technique for approximating *P* value

The general method

In this subsection, we call our statistic *S* to emphasize that this part of the method is applicable to any statistic and not just information content. Our goal is to determine the *P* value when *S* has a value of S_o. Let $P(S_o)$ be the probability of observing an *S* with a value of S_o. If S_o is close to the average value of *S*, the Central Limit Theorem will frequently lead to a sufficient approximation of the *P* value. For example, $2NI_{seq}$ can be approximated by the χ^2 distribution near the average value of NI_{seq}. The technique described here is applicable both near and far from the average value of *S*. The goal of this technique is to convert the probability distribution into two components. One component can be determined exactly. The other component contains a probability distribution $P_\gamma(S)$ whose average value for *S* equals S_o.

Let $M(\theta)$ be the moment-generating function for the probability distribution $P(S)$. $M(\theta)$, which is defined in most introductory statistics books, is:

$$M(\theta) \equiv \sum_{\text{all S}} e^{\theta S} P(S) \tag{3}$$

We define a new probability distribution $P_\theta(S)$ as:

$$P_\theta(S) \equiv \frac{e^{\theta S} P(S)}{M(\theta)} \tag{4}$$

The $M(\theta)$ in the denominator ensures that:

$$\sum_{\text{all } S} P_\theta(S) = 1$$

By the definitions in equations (3) and (4), the average of *S* for P_θ, μ_θ, is a function of the moment-generating function and its first derivative:

$$\mu_\theta \equiv \sum_{\text{all } S} S P_\theta(S) = M'(\theta)/M(\theta) \tag{5}$$

The variance of *S* for P_θ, σ_θ^2, is a function of the moment-generating function, its second derivative, and μ_θ:

$$\sigma_\theta^2 \equiv \sum_{\text{all } S} (S-\mu_\theta)^2 P_\theta(S) = M''(\theta)/M(\theta) - \mu_\theta^2 \tag{6}$$

When θ equals zero, formulas (5) and (6) give the average and variance of *S* for the original distribution $P(S)$.

Equation (4) can be rearranged so that $P(S)$ is a function of $P_\theta(S)$:

$$P(S) = \left[\frac{M(\theta)}{e^{\theta \mu_\theta}}\right] e^{-\theta(S-\mu_\theta)}, P_\theta(S) \tag{7}$$

Our goal in this technique is to work with a probability distribution $P_\gamma(S)$ whose average value for *S* is S_o. $P_\gamma(S)$ is obtained from equation (4) by setting θ to a value γ such that μ_θ equals S_o. From equation (5), γ is determined from the following formula:

$$S_o = M'(\gamma)/M(\gamma)$$

We numerically solve for γ using an algorithm that combines the Newton–Raphson method with bisection (Press *et al.*, 1988, pp. 273–274). We calculate σ_γ^2 as a by-product of this algorithm because it requires the derivative of μ_θ with respect to θ, which happens to equal σ_θ^2. In practice, this numerical solution requires only a few iterations.

By substituting γ into equation (7), we obtain:

$$P(S) = \left[\frac{M(\gamma)}{e^{\gamma S_o}}\right] e^{-\gamma(S-S_o)} P_\gamma(S)$$

and, thus, the *P* value of S_o equals:

$$P(S \geq S_o) = \left[\frac{M(\gamma)}{e^{\gamma S_o}}\right] \sum_{S \geq S_o} e^{-\gamma(S-S_o)} P_\gamma(S) \tag{8}$$

If $M(\theta)$ and its first two derivatives can be determined efficiently enough, γ, σ_γ^2 and the bracketed component of equation (8) can be determined numerically.

If the overall statistic *S* is the sum of many independent statistics, the Central Limit Theorem justifies estimating $P_\gamma(S)$ with a normal distribution. Under these conditions, the summation in equation (8) can be approximated by an integral. Thus, assuming S_o is greater than or equal to the average *S* so that $\gamma \geq 0$:

$$\sum_{S \geq S_o} e^{-\gamma(S-S_o)} P_\gamma(S)$$

$$\approx \int_{S_o}^{\infty} e^{-\gamma(S-S_o)} \frac{1}{\sigma_\gamma \sqrt{2\pi}} e^{-(S-S_o)^2/(2\sigma_\gamma^2)} \, dS \tag{9}$$

$$= e^{(\gamma\sigma_\gamma)^2/2} \int_{\gamma\sigma_\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx$$

$$= \frac{e^{(\gamma\sigma_\gamma)^2/2}}{2} \int_{(\gamma\sigma_\gamma)^2/2}^{\infty} \frac{1}{\sqrt{\pi}} \left(y^{-1/2}\right) e^{-y} dy \tag{10}$$

$$\approx \frac{e^{(\gamma\sigma_\gamma)^2/2}}{2}\int_{(\gamma\sigma_\gamma)^2/2}^{\infty}\frac{1}{\sqrt{\pi}}\left(\frac{\sqrt{2}}{\gamma\sigma_\gamma}\right)e^{-y}dy$$

$$= \frac{1}{\gamma\sigma_\gamma\sqrt{2\pi}} \qquad (11)$$

γ needs to be non-negative in the above formulas so that the first exponential term in formula (9) does not increase to infinity and magnify the error in using a normal distribution to approximate $P_\gamma(S)$. If S_o is less than the average of S, the large-deviation technique can be used to estimate $P(S \le S_o)$. However, when S_o is less than the average, it is sufficient for our purposes to approximate the distribution $P(S)$ with a gamma distribution for the variable $y = S - S_{min}$, fitted to the numerically determined average and standard deviation.

The integral in formula (10) corresponds to a gamma distribution, which can be efficiently determined numerically (Press *et al.*, 1988, pp. 171–174). Formula (10) can be approximated analytically by formula (11) when S_o is not too close to the average or maximum value of S, i.e. when $\gamma\sigma_\gamma >> 1$. However, we have chosen to use formula (10) rather than formula (11) in our current implementation of this method for approximating P value.

A final set of approximations are useful for scores near the minimal score, S_{min}, but greater than the average, and for scores near the maximal score, S_{max}. The normal approximation of $P_\gamma(S)$ becomes increasingly inaccurate as the difference between S_o and either S_{min} or S_{max} becomes small relative to σ_γ. For example, when S_o approaches S_{max}, $\gamma\sigma_\gamma$ approaches zero and formula (10) equals 0.5 rather than 1, which would be the correct result. To improve our estimate for $P_\gamma(S)$ when $S_o - S_{min} < 3\sigma_\gamma$, we approximate $P_\gamma(S)$ with a gamma distribution for the variable $y = S - S_{min}$, rather than with a normal distribution. To improve our estimate for $P_\gamma(S)$ when $S_{max} - S_o < 3\sigma_\gamma$, we approximate $P_\gamma(S)$ with a gamma distribution for the variable $y = S_{max} - S$.

Applying the large-deviation technique to multiple sequence alignments

To apply the techniques described above, we need to be able to calculate efficiently the moment-generating function $M(\theta)$ and its first two derivatives $M'(\theta)$ and $M''(\theta)$ for the statistic of interest. In this subsection, we describe these calculations for the statistic NI_{seq}, where N is the total number of sequences in the alignment and I_{seq} is the information content statistic defined in formula (2). While analytical approximations would be desirable, a numerical calculation is practical and is what we describe here.

Since we restrict ourselves to simple alignment models in which each column is independent, the moment-generating function only needs to be calculated for a single column since the overall moment-generating function is only dependent on M_c, the moment-generating function for an individual column, and L, the width of the alignment:

$$M(\theta) = M_c(\theta)^L$$

Furthermore, since each column is independent, the statistic NI_{seq} is the sum of an independent component for each column. Thus, as L becomes large, $P_\theta(NI_{seq})$ approaches a normal distribution for values of NI_{seq} close to μ_θ. However, in our experience, the normal approximation works well for $P_\theta(NI_{seq})$ even when the width L equals 1 (see Figure 3c and e).

By the definition in equation (3), the moment-generating function for NI_{seq} for an individual column is:

$$M_c(\theta) = \sum_{\Sigma n_i = N} \exp\left(\theta \sum_{i=1}^{A} n_i \ln\frac{n_i/N}{p_i}\right)\frac{N!}{\Pi_{i=1}^{A} n_i!}\prod_{i=1}^{A} p_i^{n_i} \qquad (12)$$

where n_i is the occurrence of letter i. The outer summation is taken over all combinations of the n_i summing to N. The total number of such combinations is $(N + A - 1)!/N!/(A-1)!$, i.e. $O(N^{A-1})$. A brute force calculation for $M(\theta)$ involving all these combinations is not too bad for a DNA alignment where $A = 4$, but is unacceptable for proteins where there are 20 letters in the amino acid alphabet. Therefore, we use the following dynamic programming algorithm, whose complexity is only $O[(A-2)N^2]$ in time and $O(N)$ in space.

To simplify our description of this algorithm, we define a function $M_i(\theta, n)$ as the component of the moment-generating function that is dependent on letter i, given θ and the occurrence n of letter i:

$$M_i(\theta, n) \equiv \exp\left(\theta n \ln\frac{n/N}{p_i}\right)\frac{p_i^n}{n!}$$

Thus, the definition of $M_c(\theta)$ in equation (12) can be rewritten as:

$$M_c(\theta) = \sum_{\Sigma n_i = N} N! \prod_{i=1}^{A} M_i(\theta, n_i)$$

In this algorithm, the contribution of each letter is incorporated into the calculation for $M_c(\theta)$ one at a time. The algorithm creates a matrix $\mathcal{M}(i,n)$ which contains the intermediate calculations of $M_c(\theta)$ through letter i and containing n sequences. Within $\mathcal{M}(i,n)$, i varies from 1 through A and n varies from 0 through N. $M_c(\theta)$ will equal $\mathcal{M}(A,N)$. For $0 \le n \le N$, $\mathcal{M}(1,n)$ is initialized by the rule:

$$\mathcal{M}(1,n) = N!\, M_1(\theta,n)$$

For $1 < i < A$ and $0 \le n \le N$, $\mathcal{M}(i,n)$ is determined by the rule:

$$\mathcal{M}(i, n) = \sum_{j=0}^{n} \mathcal{M}(i-1, j) M_i(\theta, n-j) \qquad (13)$$

Finally,

$$M_c(\theta) = \mathcal{M}(A,N) = \sum_{j=0}^{N} \mathcal{M}(A-1,j) M_A(\theta, N-j)$$

To calculate $M_c'(\theta)$, the algorithm creates a matrix $\mathcal{M}'(i,n)$ which contains the intermediate calculations of $M_c'(\theta)$ through letter i and containing n sequences. $M_c'(\theta)$ is determined analogously to $M_c(\theta)$ such that $M_c'(\theta)$ will equal $\mathcal{M}'(A,N)$. For $0 \le n \le N$, $\mathcal{M}'(1,n)$ is initialized by the rule:

$$\mathcal{M}'(1,n) = N! \frac{\partial M_1(\theta, n)}{\partial \theta}$$

For $1 < i < A$ and $0 \le n \le N$, $\mathcal{M}'(i,n)$ is determined by the rule:

$$\mathcal{M}'(i,n) = \sum_{j=0}^{n} \left[\mathcal{M}'(i-1,j)\, M_i(\theta, n-j) + \mathcal{M}(i-1,j) \frac{\partial M_i(\theta, n-j)}{\partial \theta} \right] \quad (14)$$

Finally,

$$M_c'(\theta) = \mathcal{M}'(A,N) = \sum_{j=0}^{N} \left[\mathcal{M}'(A-1,j) M_A(\theta, N-j) + \mathcal{M}(A-1,j) \frac{\partial M_A(\theta, N-j)}{\partial \theta} \right]$$

To calculate $M_c''(\theta)$, the algorithm creates a matrix $\mathcal{M}''(i,n)$ which contains the intermediate calculations of $M_c''(\theta)$ through letter i and containing n sequences. $M_c''(\theta)$ is determined analogously to $M_c(\theta)$ such that $M_c''(\theta)$ will equal $\mathcal{M}''(A,N)$. For $0 \le n \le N$, $\mathcal{M}''(1,n)$ is initialized by the rule:

$$\mathcal{M}''(1,n) = N! \frac{\partial^2 M_1(\theta, n)}{\partial \theta^2}$$

For $1 < i < A$ and $0 \le n \le N$, $\mathcal{M}''(i,n)$ is determined by the rule:

$$\mathcal{M}''(i,n) = \sum_{j=0}^{n} \left[\mathcal{M}''(i-1,j) M_i(\theta, n-j) + 2\mathcal{M}'(i-1,j) \frac{\partial M_i(\theta, n-j)}{\partial \theta} + \mathcal{M}(i-1,j) \frac{\partial^2 M_i(\theta, n-j)}{\partial \theta^2} \right] \quad (15)$$

Finally,

$$M_c''(\theta) = \mathcal{M}''(A,N) = \sum_{j=0}^{N} \left[\mathcal{M}''(A-1,j) M_A(\theta, N-j) + 2\mathcal{M}'(A-1,j) \frac{\partial M_A(\theta, N-j)}{\partial \theta} + \mathcal{M}(A-1,j) \frac{\partial^2 M_A(\theta, N-j)}{\partial \theta^2} \right]$$

Since $M_c(\theta)$, $M_c'(\theta)$ and $M_c''(\theta)$ can be efficiently determined for NI_{seq}, we can numerically determine γ, $M(\gamma)$, $M'(\gamma)$, $M''(\gamma)$ and σ_γ. These values can then be used to approximate $P_\gamma(NI_{seq})$ with either a normal or a gamma distribution, which can be substituted into equation (8) to estimate the P value for a specified value of NI_{seq}.

A special case, considered by some of our algorithms, is nucleic acid alignments in which a pattern is assumed to be symmetrical. In this case, when a word is incorporated into an alignment, its reverse complement is also added. If one knows the left half of such an alignment matrix, then one also knows the right half. If the width L of the alignment is even, the moment-generating function is:

$$M(\theta) = M_c(2\theta)^{L/2}$$

where $M_c()$ is defined in equation (12). If L is odd, the moment-generating function is:

$$M(\theta) = M_c(2\theta)^{(L-1)/2} M_{center}(2\theta)$$

where $M_{center}(2\theta)$ is the moment-generating function for the central position of the alignment. $M_{center}()$ differs from $M_c()$ in two ways. First, $M_{center}()$ substitutes $N/2$ for N since only $N/2$ of the letters in the central position are independent. Second, $M_{center}()$ uses an alphabet consisting of only $A/2$ letters since each letter of the original alphabet is indistinguishable from its complement. The *a priori* probability of one of these new letters is the sum of the *a priori* probabilities of the corresponding complementary letters, which should each have the same *a priori* probability.

Approximating the *P* value numerically

In this section, we describe an alternative method for approximating the P values of NI_{seq}. This method creates a table of P values for the statistic after it has been transformed into integer values. The statistic NI_{seq} is transformed into an integer value I' after multiplying NI_{seq} by some factor α:

$$I' \equiv \text{int}(\alpha NI_{seq}) \quad (16)$$

in which the 'int' function rounds a real number to its closest integer. α is chosen so that the maximum (I'_{max}) and minimum (I'_{min}) values of I' differ by some desired amount. The greater the difference between I'_{max} and I'_{min}, the more accurate the estimation of the P value.

In principle, *a probability-generating function*, $G(x)$, can be created for an alignment having a width of L:

$$G(x) = \sum_{I'=I'_{min}}^{I'_{max}} P_L(I') x^{I'}$$

in which $P_L(I')$ is the probability of observing the specified value of I', i.e. the probability of observing $(I' - 0.5)/\alpha \le NI_{seq} < (I' + 0.5)/\alpha$. Thus, the P value for I'_o would equal:

$$\sum_{I'=I'_o}^{I'_{max}} P_L(I')$$

Staden (1989) described an efficient method for numerically estimating the probability-generating function for weight-matrix scores. If the probability-generating function for the information content of an individual column of an alignment matrix is known, Staden's approach can be directly used to approximate the probability-generating function for a multi-column alignment matrix. Let $P(I')$ be the probability of observing I', given a single alignment position, i.e. $P(I') = P_1(I')$. Let I'_m and I'_M be the minimum and maximum values of I', respectively, for a single alignment position. $P_L(I')$ can be approximated from $P_{L-1}()$ and $P()$ using the following relationship:

$$P_L(I') \approx \sum_{j=I'_m}^{I'_M} P(j) P_{L-1}(I'-j)$$

The summation only needs to be taken for values of j for which $P(j) \neq 0$. Let $B \equiv I'_M - I'_m + 1$, and let L be the width of the alignment. For a weight matrix, the distribution analogous to $P(I')$ only has A non-zero values. However, for an alignment matrix, $P(I')$ may have up to B values. Thus, in the worst case, the time complexity of this algorithm is $O(L^2B^2)$.

The probability-generating function for an individual column of a weight matrix requires the determination of only A weights and probabilities; thus, its calculation is trivial to do directly. On the other hand, a brute-force calculation of the probabilities for the information content of an individual column of an alignment matrix is much more complex. This calculation requires a determination for all

$$\frac{(N + A - 1)!}{N!(A - 1)!}$$

combinations of N letters taken from an alphabet containing A letters. This brute force calculation can be practical for nucleic acids where $A = 4$. When A is larger, such as 20 for an amino acid alphabet, the probability-generating function can be approximated using an algorithm similar to that for determining the moment-generating function in the previous section. However, the complexity of the algorithm here is $O[(A-2)BN^2]$ in time and $O(BN)$ in space, which are more complex by a factor of B.

In this algorithm, the probability-generating function is constructed one letter at a time. The algorithm creates a three-dimensional matrix, $\mathcal{P}(i, n, I')$, which contains the intermediate approximation of $P(I')$ through letter i, containing n sequences, and having an intermediate value of I'. Within $\mathcal{P}(i, n, I')$, i varies from 1 through A and n varies from 0 through N. The range of I' is dependent on the values of i and n. $P(I')$ will be approximately equal to $\mathcal{P}(A, N, I')$.

To simplify our description of the algorithm, we define two functions. $I'_i(n)$ is the integer approximation of the component of I' dependent on letter i:

$$I'_i(n) \equiv \text{int}\left(\alpha n \ 1\text{n}\frac{n/N}{p_i}\right)$$

$P_i(n)$ is the component of the multinomial probability [formula (1)] that is dependent on letter i:

$$P_i(n) \equiv \frac{p_i^n}{n!}$$

For $0 \leq n \leq N$, $\mathcal{P}(1,n, I')$ is initialized by the rule:

$$\mathcal{P}(1, n, I'_1(n)) = N! \ P_1(n)$$

For $1 < i < A$ and $0 \leq n \leq N$, $\mathcal{P}(i,n, I')$ is determined by the rule:

$$\mathcal{P}(i, n, I') = \sum_{k=0}^{n} P_i(k) \ \mathcal{P}(i-1, n-k, I'-I'_i(k)) \qquad (17)$$

in which the upper and lower limits of k will frequently need to be lowered and raised, respectively, so that $I'-I'_i(k)$ is within a range consistent with $i - 1$ and $n - k$. Finally,

$$P(I') \approx \mathcal{P}(A, N, I') = \sum_{k=0}^{N} P_A(k) \ \mathcal{P}(A-1, N-k, I'-I'_A(k))$$

The overall time complexity of this method for numerically determining probability-generating functions is $O[(A-2)BN^2 + L^2B^2]$. Relative to the large-deviation method described in the previous section, this method has the disadvantage that its complexity is dependent on the value of B and the width L of the alignment. It has the advantage over the previous method in that its accuracy can be increased by increasing the value of B. Furthermore, it generates a table for the full range of information contents rather than just for an individual information content. The speed of this method can be increased by creating a table of probabilities only for scores greater than a specified value. The greater this minimum score, the greater the speed for the same level of accuracy.

For our alignment algorithms, the large-deviation technique described in the previous section is practical under a wider range of conditions. Table 1 demonstrates the difference in speed between the two algorithms under various conditions. We have recently realized that the Fast Fourier Transform algorithm and the Convolution Theorem (Press *et al*., 1988, Chapter 12) can be applied to formulas (13), (14), (15) and (17) to greatly reduce the time complexity for our algorithms to calculate P values. Thus, the calculation of the moment-generating function in the large-deviation technique can be done in time $O[(A-2)N \log_2 N]$ rather than $O[(A-2)N^2]$, and the numerical calculation of P value can be done in time $O[(A-2)BN \log_2 (BN) + LB \log_2(LB)]$ rather than $O[(A-2)BN^2 + L^2B^2]$. We have not yet incorporated the Fast Fourier Transform into our programs.

Table 1. The time required to determine the *P* value for DNA sequence alignments containing various numbers of sequences and having various widths. Shown are the time to determine a table of *P* values numerically (NUM) and the time to determine an individual *P* value using the large-deviation (LD) method. Equiprobable DNA alphabets were used in all examples. NI_{seq} was multiplied by 10 before converting to an integer for determining *P* values numerically [formula (16)]. LD times are the average of 10 determinations each at a different value of NI_{seq}. Programs were run on a SUN Ultra 30 workstation with a 296 MHz processor

Number of sequences	Width	Time NUM	Time LD	NUM/LD
10	1	0.01 s	0.005 s	2
10	10	0.1 s	0.005 s	20
10	100	12 s	0.005 s	2400
100	1	0.2 s	0.3 s	0.6
100	10	51 s	0.3 s	170
100	100	1.6 h	0.3 s	19 000
1000	1	3 min	32 s	6
1000	10	2 h	32 s	230

Counting the number of possible alignments

The *P* value of an individual alignment is not usually sufficient to describe the statistical significance of that alignment. We are generally interested in the overall best alignment given a typically huge number of possible alignments that can be constructed from a set of sequence data. For example, an alignment of width *L*, having a contribution of exactly one word from each of *N* sequences of length *Q*, can be chosen in $(Q - L + 1)^N$ ways. More generally, an alignment may contain *at most* one word from each of *N* sequences. Let *n* be the number of words in the alignment, let $Q' = (Q - L + 1)$ be the number of possible starting positions in each sequence, and let $\mathcal{A}(n)$ be the number of possible alignments. Given that $n \le N$, an alignment can be chosen in the following number of ways:

$$\mathcal{A}(n) = \frac{N!}{n!(N-n)!}(Q')^n \tag{18}$$

Another generalization allows each sequence to contribute one or more words to the alignment. We derive a formula for $\mathcal{A}(n)$, in which $n \ge N$, with the aid of a generating function—a polynomial in which the coefficient of x^n is $\mathcal{A}(n)$. In the following derivation, we have permitted alignments to contain overlapping words so that the alignment width *L* only enters the results through Q'. This assumption simplifies the problem, but gives an overestimate of $\mathcal{A}(n)$ if overlaps are not permitted.

The coefficients of the following polynomial are the number of ways *n* words can be chosen from a single sequence for $1 \le n \le Q'$:

$$(x+1)^{Q'} - 1 = \sum_{n=1}^{Q'} \frac{Q'!}{n!(Q'-n)!} x^n$$

Therefore, the coefficients of the following polynomial are the number of ways *n* words can be chosen from a set of *N* sequences when each sequence must contribute at least one word to the total of *n* words:

$$\begin{aligned}\sum_{n=N}^{NQ'} \mathcal{A}(n)x^n &= \left[(x+1)^{Q'} - 1\right]^N \\ &= \sum_{i=0}^{N} (-1)^{N-i} \frac{N!}{i!(N-i)!} (x+1)^{iQ'} \\ &= \sum_{n=N}^{NQ'} \sum_{i \ge \frac{n}{Q'}}^{N} (-1^{N-i}) \left[\frac{N!}{i!(N-i)!}\right]\left[\frac{(iQ')!}{n!(iQ'-n)!}\right] x^n\end{aligned}$$

Thus, given that $n \ge N$, an alignment can be chosen in the following number of ways:

$$\mathcal{A}(n) = \sum_{i \ge \frac{n}{Q'}}^{N} (-1^{N-i}) \left[\frac{N!}{i!(N-i)!}\right]\left[\frac{(iQ')!}{n!(iQ'-n)!}\right] \tag{19}$$

In formula (19), the dummy variable *i* is initialized to the smallest integer $\ge n/Q'$.

Formulas (18) and (19) give the same result when $n = N$. If the lengths of each of the *N* sequences are not identical, we approximate $(Q - L + 1)$ by its geometric mean so that formulas (18) and (19) are exact only when each sequence contributes exactly one word to the alignment (i.e. $n = N$).

An even less restrictive alignment constraint allows each of the *N* sequences to contribute zero or more words to the alignment. If the *i*th sequence has a length of Q_i, then there are a total of $N_T = \Sigma_{i=1}^{N}(Q_i - L + 1)$ possible starting positions for each of the *n* words in the alignment. Therefore, the alignment can be chosen in

$$\mathcal{A}(n) = \frac{N_T!}{n!(N_T - n)!} \tag{20}$$

ways if the words contained in the alignment are allowed to overlap.

If each alignment was independent of each other alignment, the number of possible alignments ($\mathcal{A}$) could be combined with the *P* value (P_{mat}) for an individual alignment matrix to determine an overall *P* value ($P_{overall}$):

$$P_{overall} = 1 - (1 - P_{mat})^{\mathcal{A}} \tag{21}$$

$$\approx 1 - \exp(-\mathcal{A}\, P_{mat}) \tag{22}$$

$$\approx \mathcal{A}\, P_{mat} \tag{23}$$

The approximation in formula (22) assumes that $P_{mat} << 1$ and corresponds to an extreme value distribution. If the lower

limit of integration in formula (9) is set to a value $S_o + \varepsilon$ close to S_o, the approximation for P value corresponding to formula (11) generalizes to:

$$P(S \geq S_o + \varepsilon) \approx \left[\frac{M(\gamma)}{e^{\gamma S_o}}\right] \frac{e^{-\gamma\varepsilon}}{\gamma\sigma_\gamma\sqrt{2\pi}}$$

When this approximation for P value is substituted into formula (22), we obtain the P value for the type of extreme value distribution that Claverie (1994) observed for the scores of weight matrices. However, this approximation and the precise extreme value distribution should only be accurate within a localized region around the score S_o.

The approximation in formula (23) further assumes that $\mathcal{A}P_{mat} << 1$. Although $\mathcal{A}P_{mat}$ is only approximately equal to the overall P value, it is exactly equal to the expected number of alignments having an information content greater than or equal to the observed value of I_{seq}, even if the alignments are not independent. However, a lack of independence will increase the standard deviation of this expectation. We call this expectation the *expected frequency* of the information content, given the width and the number of sequences in the alignment.

Since alignments are generally not independent, we use the expected frequency [formula (23)] to compare alignments rather than the overall P value [formula (21) or (22)]. The expectation will be larger than the overall P value and, therefore, is a more conservative measure of statistical significance than is the overall P value. Both the expectation and the overall P value allow the comparison of alignments having differing widths and containing differing numbers of sequences. Multiplying the expectation or overall P value by the number of different widths and the maximum number of sequence words being considered would conservatively account for these two additional degrees of freedom.

In principle, the P value of an individual alignment and the overall P value can be determined by repeatedly aligning randomized sequences. However, in general, such procedures cannot be repeated enough to observe the very small P values and expectations seen in practice (e.g. Table 2). In the case of the overall P value, the randomized-sequence approach is further limited because practical alignment algorithms are not guaranteed to find the highest scoring alignment, unless the amount of sequence data is relatively small. With randomly generated sequences, multiple alignment algorithms will probably fail to find the highest scoring alignment because too many alignments will have scores close to this mathematically optimum alignment. Thus, repeatedly aligning randomized sequences is not expected to work well to determine accurate overall P values, although this procedure can still give insight into whether a particular value of I_{seq} is expected by chance.

Algorithms for determining the alignment having the optimum information content

Our ultimate goal is to apply our models and statistics for sequence alignments to identify optimal alignments and determine consensus patterns describing functional relationships. Our models and statistics are applicable to various algorithms for determining multiple-sequence alignments, e.g. expectation maximization (Lawrence and Reilly, 1990), Gibbs sampling (Lawrence *et al*., 1993) and our greedy algorithm (Stormo and Hartzell, 1989; Hertz *et al*., 1990). The goal of all these algorithms is to determine a sequence alignment that maximizes a log-likelihood statistic. In this section, we describe the current version of our greedy algorithm.

Table 2. The most statistically significant alignments found by the CONSENSUS program when aligning CRP-binding sequences using *a priori* probabilities of 30% for A and T, and 20% for G and C

Width	Number of sequences	P value	Expected frequency
16	17	8×10^{-50}	8×10^{-10}
17	4	5×10^{-13}	1×10^{-1}
18	17	3×10^{-48}	2×10^{-8}
19	6	1×10^{-18}	3×10^{-2}
20	19	2×10^{-54}	5×10^{-11}
21	6	5×10^{-19}	8×10^{-3}
22	22	3×10^{-60}	2×10^{-11}
23	9	2×10^{-26}	2×10^{-3}
24	21	4×10^{-57}	2×10^{-10}
25	9	7×10^{-28}	6×10^{-5}
26	21	2×10^{-56}	9×10^{-10}
27	6	2×10^{-18}	2×10^{-2}
28	21	1×10^{-55}	3×10^{-9}
29	12	7×10^{-32}	7×10^{-3}
30	21	6×10^{-55}	8×10^{-9}

We previously described an algorithm which sought an alignment that maximized the information content [formula (2)], but was dependent on the order with which the sequences were presented to the program (Stormo and Hartzell, 1989; Hertz *et al*., 1990). Here, we describe two related alignment algorithms that are order independent. The first algorithm, like our previously published algorithm, requires the user to specify the width of the pattern being sought based on previous biochemical knowledge. The second algorithm determines the width of the alignment, but requires that the user adjust a bias that is subtracted from the information content so that the average score is negative. To use these algorithms effectively, the width of the pattern needs to be varied either directly, as in the first algorithm, or indirectly by varying the bias, as in the second algorithm. The expected fre-

quency statistic described in the previous section can then be used to compare alignments having differing widths and containing differing numbers of sequences.

As with all common multiple alignment algorithms, our approach is a compromise between the need to keep the algorithm computationally practical and the desire to obtain the mathematically optimum alignment. Thus, our algorithms are not guaranteed to give the mathematically optimum alignment for alignments containing many sequences.

The user specifies the width of the alignment

We first describe our algorithm in which the user sets the width directly. This algorithm is implemented in a program called CONSENSUS. The user first designates the maximum number of alignments that can be saved (e.g. 100 or 1000). Typically, less alignments are ultimately saved because some will be identical. Besides the width, there are various constraints the user can impose. The following three alternatives correspond to the constraints, described in the previous section, that control the number of times each sequence can contribute to an alignment:

1. Each sequence can contribute at most one word to the alignment [corresponds to formula (18)].
2. After each sequence has contributed exactly once to the alignment, sequences can contribute additional words to the alignment [corresponds to formula (19)].
3. Each sequences can contribute zero or more words to the alignment [corresponds to formula (20)].

For alternatives (2) and (3), the user also needs to decide by how much words from the same sequence should be allowed to overlap.

When aligning nucleotide sequences, the user also needs to decide whether to include the complementary sequence. If the complementary sequence is included, the user needs to decide whether complementary words will be allowed in the same alignment. If complementary words are allowed in the same alignment, the user needs to decide whether to require the pattern to be symmetrical, thus requiring the inclusion of complementary words in each alignment.

Figure 2 gives a simplified example of the alignment algorithm when aligning exactly one word from each sequence. The following is a more general description of how the alignment algorithm proceeds.

CYCLE 1 Create a one-sequence alignment matrix for each sequence word. If the user has sufficient prior information, the initial set of one-sequence alignment matrices can be created from a subset of the total sequence data. Such user intervention can be essential if there is a large amount of sequence data.

CYCLE 2 Subject to the constraints determined by the user, determine all possible pairwise alignments of the one-sequence alignment matrices and the remaining sequence words to create alignment matrices representing two-sequence alignments. Score the new alignment matrices according to their information content. The highest scoring two-sequence alignment matrices, derived from each one-sequence alignment matrix, are saved up to the user-specified number of alignments.

CYCLE 3 Each alignment matrix saved from CYCLE 2 is paired with each word not already contained in the alignment matrix, and the new three-sequence alignment matrices are scored according to their information content. The highest scoring three-sequence alignment matrices, derived from each two-sequence alignment matrix, are saved up to the user-specified number of alignments.

CYCLE N Each alignment matrix saved from CYCLE ($N-1$) is paired with each word not already contained in the alignment matrix, and the new N-sequence alignment matrices are scored according to their information content. The highest scoring N-sequence alignment matrices, derived from each ($N-1$)-sequence alignment matrix, are saved up to the user-specified number of alignments.

The algorithm continues until each sequence has contributed exactly once to each saved alignment or until some user-determined number of words contributes to each alignment. The user can also direct the algorithm to quit when it appears to have already identified the alignment having the minimum expected frequency. This is decided if a user-specified number of cycles passes after creating the alignment that currently has the minimum expected frequency. The program can print the highest scoring alignment matrix from each cycle; thus, a collection of alignments having differing numbers of words can be printed. These alignment matrices are ordered according to their expected frequency. The program can also print the alignment matrices saved after the last cycle.

Whenever two alignments are identical, only one of the two alignments is saved. Each saved alignment is summarized in a matrix whose elements are the number of occurrences of letters. Therefore, each pairwise alignment involves the aligning of a sequence against a matrix and, thus, is similar in concept to aligning a protein sequence against the profile of a protein family (Gribskov *et al*., 1990).

The user specifies a standard deviation bias

We have also developed a similar algorithm in which the user does not explicitly specify the width of the alignment. This algorithm is implemented in a program called WCONSENSUS. A property of the information content is that it is always

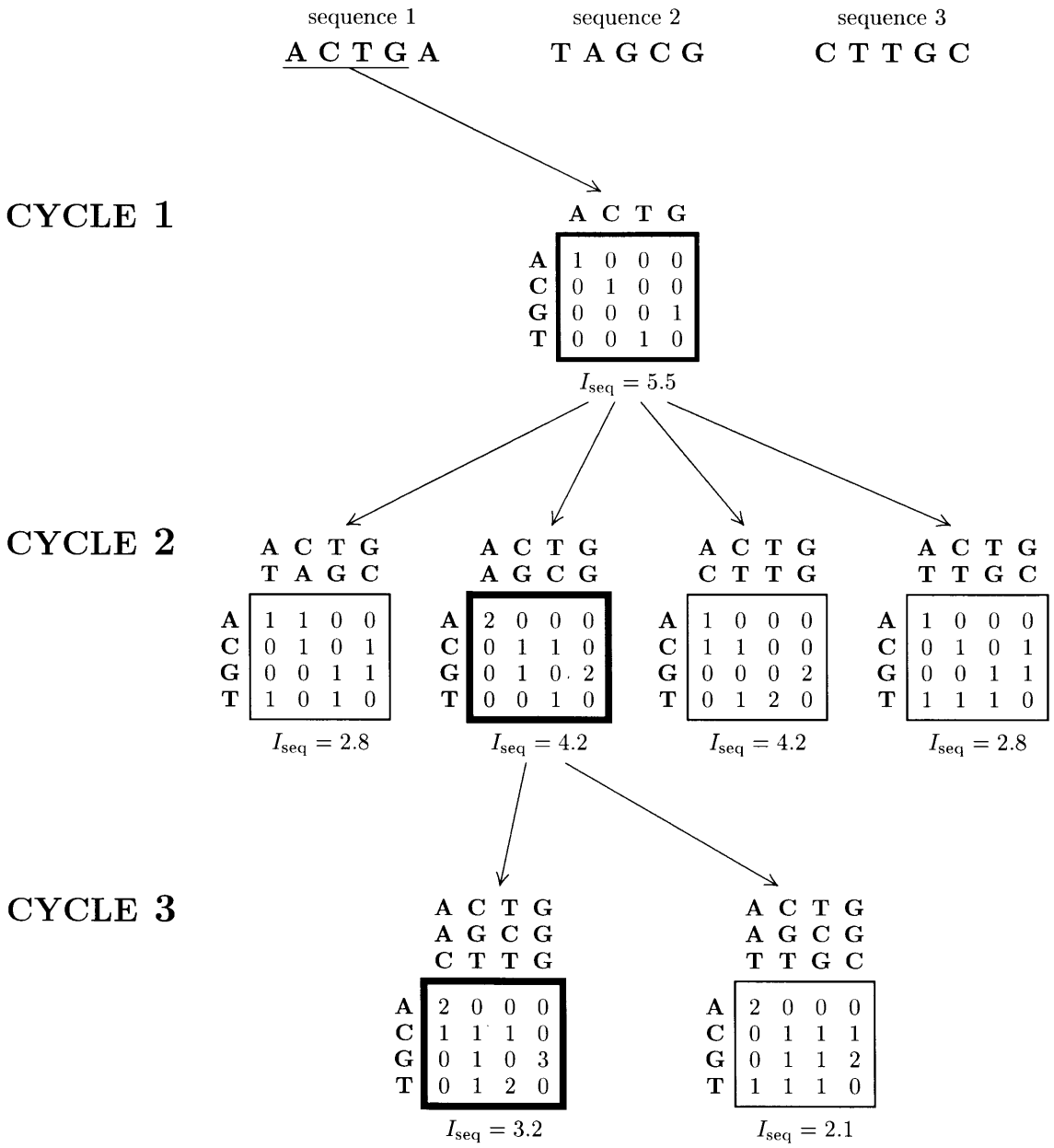

Fig. 2. An example of the algorithm for finding sequence alignments of a fixed width, assuming each sequence contributes exactly once to the final alignment. Alignments of width 4 are being sought from the three single-stranded DNA sequences listed at the top. Each base has an *a priori* probability of 25%. **Cycle 1:** For simplicity, only the alignment matrix originating from the first 4-mer of the first sequence is shown. In the actual program, one alignment matrix would be created for each of the six 4-mers that exist in the three sequences of length five. **Cycle 2:** The four matrices that can be created by adding an additional sequence to the matrix shown in cycle 1. We designated that a maximum of six alignments be saved after the first cycle; therefore, only one descendant of the parental matrix shown in cycle 1 can be saved. The saved matrix is surrounded by the heavy line. If all six matrices created in cycle 1 were followed, four matrix alignments would be saved after cycle 2 because two redundant matrices will be eliminated. **Cycle 3:** The two matrices that can be created by adding a third sequence to the saved matrix shown in cycle 2. The saved matrix is surrounded by the heavy line. If all the matrices created in cycle 1 were followed, five matrix alignments would be saved after cycle 3 because one redundant matrix will be eliminated. Thus, the alignment shown is not the highest scoring.

non-negative. However, for this local alignment algorithm to work, we need the score to be negative on average so that an interesting alignment can appear as a region of positive information. This is similar to the constraint on the scores used

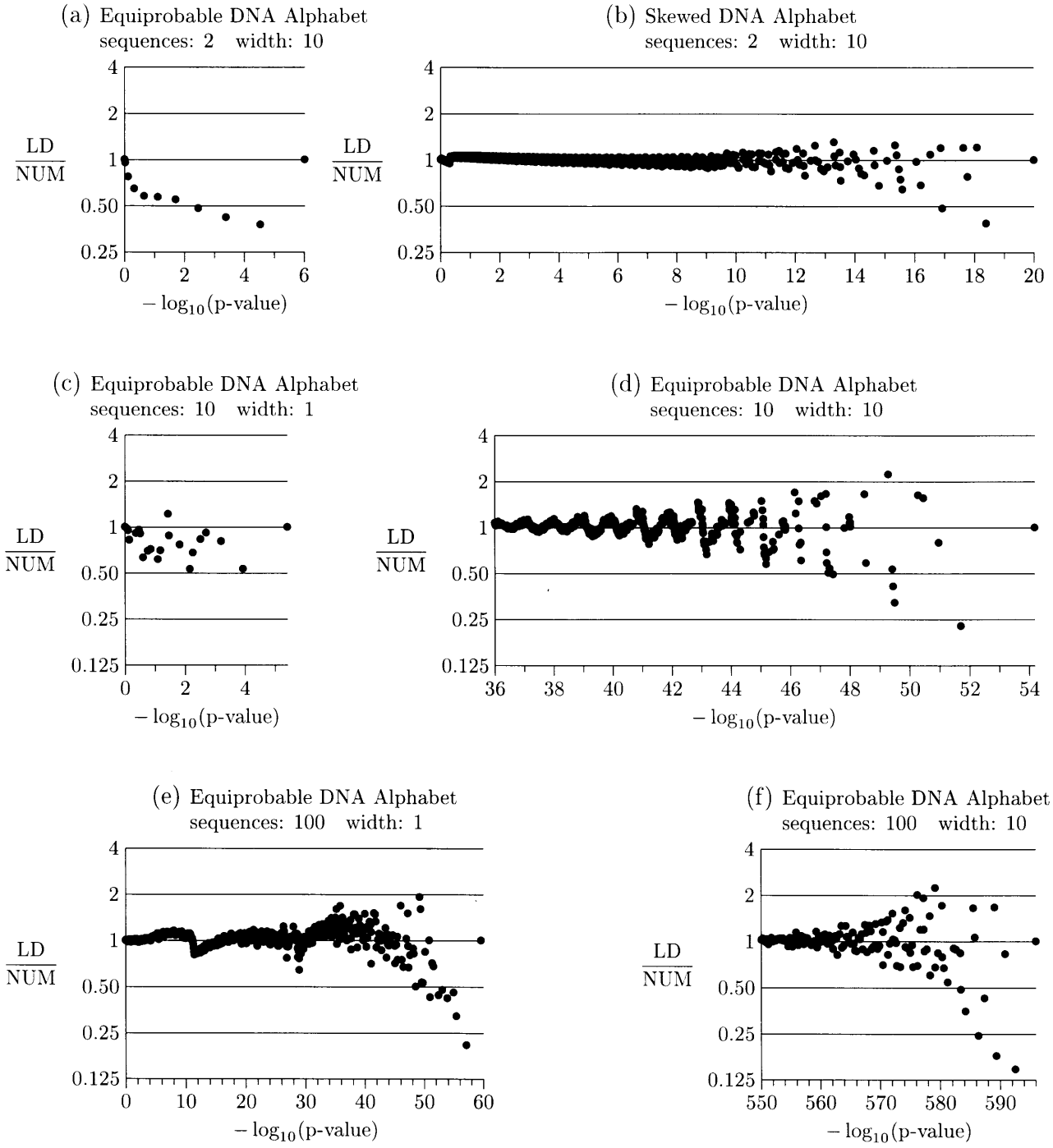

Fig. 3. The ratio of the *P* value calculated by the large-deviation (LD) method to the *P* value determined numerically (NUM) for various values of NI_{seq}. The horizontal axis is the numerically determined *P* value, which is assumed to be accurate. The vertical axis is the ratio. For simplicity, (**d**) and (**f**) exclude the left-hand portion of the graph where the ratio is always between 0.9 and 1.1. The letters of the equiprobable DNA alphabet each have an *a priori* probability of 0.25. The letters of the skewed DNA alphabet have *a priori* probabilities of 0.1, 0.2, 0.3 and 0.4.

in the algorithm of Smith and Waterman (1981). We force the scores to be negative on average by subtracting a positive bias from the information score for each column of the alignment. Higher biases decrease the width of the highest scoring alignment.

We subtract two biases from the information content for each position of an alignment. The first bias is the average information content expected from a collection of N letters occurring with the designated *a priori* probabilities, where N is the number of sequences in the alignment. This correction causes the score expected of an arbitrary alignment to equal zero. The score determined by subtracting this first bias approximates the information content of the corresponding pattern as the number of sequences in the alignment goes to infinity (Schneider *et al*., 1986). Therefore, we call this score the *adjusted information content.*

The second bias subtracted from each position is some multiple of the standard deviation of the information content expected from a collection of N letters occurring with the designated *a priori* probabilities. Subtraction of this second bias, in addition to the first, is what causes the expected alignment score to be less than zero. During the alignment algorithm, we call the score determined by subtracting these two biases from the information content the *crude information content.*

The number of standard deviations to subtract is not expected to be the same for all alignments. We try a range of values, such as 0.5, 1, 1.5 and 2, and then compare the various alignments identified according to their expected frequency or according to empirical constraints. Our standard-deviation bias, which is a simple multiple of the alignment's width, should not be confused with the standard deviation of the information content, which is a multiple of the square root of the alignment's width.

The algorithm, in which the user specifies a standard-deviation bias, is very similar to the algorithm in which the alignment width is explicitly set. However, it differs in the following two ways.

1. The algorithm seeks to maximize the crude information content rather than the true information content.
2. The saved alignment matrices include the alignment information from the sequence ends not included in the local region of high crude information content. Therefore, these end regions can become incorporated into the local alignment as additional sequences are added. Thus, a single alignment matrix can contain information on multiple motifs that are spaced the same in each of the aligned sequences. However, each alignment printed out by the program only contains a single peak of high crude information content.

Results

The accuracy of the P value approximations

The large-deviation method for determining the P value was compared to the numerical method. We used four different distributions of letters: an equiprobable four-letter DNA-like alphabet in which each letter had an *a priori* probability of 0.25; a highly skewed four-letter DNA-like alphabet in which letters had *a priori* probabilities of 1/10, 2/10, 3/10, 4/10; an equiprobable 20-letter protein-like alphabet in which each letter had an *a priori* probability of 0.05; and a highly skewed 20-letter protein-like alphabet in which letters had *a priori* probabilities of 1/210, 2/210, …, 20/210.

Since the results for the DNA-like and protein-like alphabets are similar, we only show results for the DNA-like alphabet (Figure 3). For almost all examples, we multiplied NI_{seq} by 1000 before converting the statistic to an integer for the numerical method; thus, we consider the P values calculated by the numerical method to be very accurate. The exception was for the alignments containing 100 sequences and having a width of 10, in which we only multiply by 100 to reduce the complexity of the calculation (e.g. Figure 3f).

As expected, because the estimate of $P_\gamma()$ is based on the Central Limit Theorem, the accuracy of the large-deviation method increases as the width of the alignment increases. However, the method is fairly accurate even with an alignment width of one (Figure 3c and e). As the number of sequences and the width increase, errors >10% only appear at the very largest information contents where the P value is the smallest (Figure 3d and f). For alignments containing very few sequences, the results are more accurate for the skewed *a priori* distribution, presumably because of the greater number of different values of NI_{seq} (Figure 3a versus 3b).

The decrease in accuracy as the information content approaches its maximum value is not surprising. The normal approximation to $P_\gamma()$ will become increasingly bad as the distance between its average and the minimum or maximum values of NI_{seq} becomes small relative to its standard deviation. To reduce this problem, our algorithm uses a gamma distribution to estimate $P_\gamma()$ at the lower and higher values of NI_{seq} where σ_γ is greater than one-third the distance to either the minimum or maximum value of NI_{seq}. While the normal distribution ranges from minus infinity to plus infinity, the gamma distribution has one finite end; thus, we suspected it would be a better estimate of $P_\gamma()$ when its average is close to a finite maximum or minimum.

The gamma distribution is somewhat superior to the normal distribution at the lower values of NI. This result is expected since a type of gamma distribution (i.e. the χ^2 distribution) is predicted by the Central Limit Theorem. Unfortunately, the gamma distribution does not seem to improve the estimate of $P_\gamma()$ at the higher values of NI_{seq}. Perhaps the discrete nature of the true distribution becomes increasingly important for $P_\gamma()$ as its average approaches the maximum value of NI_{seq}. Thus, neither the continuous normal nor the continuous gamma distribution are able to capture the properties of the true, discrete distribution.

Aligning DNA-binding sites of the E.coli CRP protein

The *E.coli* cyclic AMP receptor protein (CRP) is responsible for activating the transcription of many genes and repressing the transcription of a few (Collado-Vides *et al*., 1991). CRP works by directly binding to the DNA in the promoter region. This protein also goes by the name catabolite gene-activator protein (CAP).

A collection of 18 CRP-regulated genes, each 105 base pairs (bp) long, was aligned with the current version of our CONSENSUS program. Based on experimental evidence, these 18 sequences contain 24 putative binding sites. This set of sequences has proven useful for testing our original greedy alignment algorithm (Stormo and Hartzell, 1989) and the original expectation-maximization alignment algorithm (Lawrence and Reilly, 1990).

Since CRP binds as a dimer, the program was directed to assume that the binding site was symmetrical. Thus, if a sequence word was incorporated into an alignment, its reverse complement was also incorporated. Since some of the sequences had more than one CRP binding site and some of the experimentally determined sites may be incorrect, the program was directed to allow each sequence to contribute zero or more words to the alignment. However, each word had to be separated by at least 10 bp (i.e. one helical twist of the DNA). The width of the alignment was varied from 16 through 30 bp. Alignments were allowed to have up to 40 complementary word pairs.

Although the frequency of each base in the *E.coli* genome is essentially the same, promoter regions are AT rich. In our analysis, we tried both genome-like *a priori* probabilities of 25% and the frequencies in the dataset being aligned, i.e. 30% for A and T, and 20% for G and C. When the genomic frequency of 25% was used as the *a priori* probability for each base, the optimum alignment contained 14 out of the 24 putative sites, but also included 16 other AT-rich sequences in the alignment. When the observed frequencies were used as the *a priori* probabilities, the optimum alignment contained 19 out of the 24 putative binding sites. This alignment also contained three sites not believed to be CRP binding sites; however, each of these three sites overlapped a putative site.

The optimum alignment was 22 bp wide and had an expected frequency of 2×10^{-11} (Table 2). Because the alignments were required to be symmetrical, the alignments with odd widths are substantially different from the alignments with even widths. As a result, the alignments having odd widths have substantially higher expected frequencies than those having even widths. The pattern identified is similar to the one we (Stormo and Hartzell, 1989) and others (Lawrence and Reilly, 1990) have previously identified for the CRP binding site. The important difference is the statistics that allow the user to judge whether a pattern is statistically significant and to compare alternative alignments. By using the expected frequency to compare alignments, the user can impose less assumptions on the width of the pattern and on the number of sites contained in the alignment.

Discussion

Information content and related statistics have proven their usefulness for identifying and analyzing sequence alignments (Schneider *et al*., 1986; Berg and von Hippel, 1987; Stormo and Hartzell, 1989; Hertz *et al*., 1990; Lawrence and Reilly, 1990; Lawrence *et al*., 1993). However, a major deficiency has been the lack of an accurate measure of the *P* value of a particular information content. In this paper, we use large-deviation statistics and an efficient algorithm for determining the moment-generating function to estimate the *P* value of an information content accurately. The large-deviation approach is also applicable to the scores of weight matrices and, in this regard, can be used to generalize some of the statistical results presented in Berg and von Hippel (1987).

In the results presented here, we have assumed that the *a priori* probability of each letter of a sequence is independent and identically distributed. Although this assumption is not too bad an approximation for the *E.coli* genome, it is clearly deficient for eukaryotic genomes. We are currently working on extending our approach to eliminate this assumption. The simple alignment matrix (Figure 1) explicitly assumes that each position of an alignment pattern contributes independently to the activity that has been selected for. In principle, we can construct more complex alignment matrices that incorporate information on the correlations between the positions of an alignment pattern (Hertz and Stormo, 1995).

Biologically related sequences can also contain insertions and deletions relative to each other. We have developed alignment matrices and information content formulas that account for gaps (i.e. deletions) in alignment matrices (Hertz and Stormo, 1995). When the gaps at each position of an alignment are assumed to be independent, our large-deviation method can be directly applied to calculate statistical significance. However, gaps are generally expected to be correlated with any gaps in adjacent positions. We are currently extending our method to calculate the statistical significance of alignment matrices containing information on the correlations between adjacent gaps.

Our procedure for estimating the *P* value is not limited to the information content of sequence alignments. It is directly applicable to other hypothesis testing problems in which variables are independent and sampled from a finite alphabet. The statistic can be anything in which the overall statistic is the sum of components that are dependent on only an individual letter (e.g. the χ^2 statistic).

Acknowledgements

We wish to thank Christian Burks for suggesting that we model alignments containing one or more words from each sequence. We wish to thank Jean-Michel Claverie, Timothy Bailey and Josh Stuart for helpful comments on this manuscript. We especially wish to thank Josh Stuart for his careful checking of the formulas. This work was supported by Public Health Service grant HG-00249 from the National Institutes of Health.

References

Berg,O.G. and von Hippel,P.H. (1987) Selection of DNA binding sites by regulatory proteins: Statistical-mechanical theory and application to operators and promoters. *J. Mol. Biol.*, **193**, 723–750.

Bucklew,J.A. (1990) *Large Deviation Techniques in Decision, Simulation, and Estimation*. John Wiley and Sons, New York.

Claverie,J.M. (1994) Some useful statistical properties of position-weight matrices. *Comput. Chem.*, **18**, 287–294.

Collado-Vides,J., Magasanik,B. and Gralla,J.D. (1991) Control site location and transcriptional regulation in *Escherichia coli*. *Microbiol. Rev.*, **55**, 371–394.

Dembo,A., Karlin,S. and Zeitouni,O. (1994) Limit distribution of maximal non-aligned two-sequence segmental score. *Ann. Prob.*, **22**, 2022–2039.

Fields,D.S., He,Y., Al-Uzri,A.Y. and Stormo,G.D. (1997) Quantitative specificity of the Mnt repressor. *J. Mol. Biol.*, **271**, 178–194.

Gribskov,M., Lüthy,R. and Eisenberg,D. (1990) Profile analysis. *Methods Enzymol.*, **183**, 146–159.

Hertz,G.Z. and Stormo,G.D. (1995) Identification of consensus patterns in unaligned DNA and protein sequences: A large-deviation statistical basis for penalizing gaps. In Lim,H.A. and Cantor,C.R. (eds), *Proceedings of the Third International Conference on Bioinformatics and Genome Research.*. World Scientific Publishing, Singapore, pp. 201–216.

Hertz,G.Z., Hartzell,G.W. III and Stormo,G.D. (1990) Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Comput. Appl. Biosci.*, **6**, 81–92.

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. USA*, **87**, 2264–2268.

Kullback,S. and Leibler,R.A. (1951) On information and sufficiency. *Ann. Math. Stat.*, **22**, 79–86.

Lawrence,C.E. and Reilly,A.A. (1990) An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins*, **7**, 41–51.

Lawrence,C.E., Altschul,S.F., Boguski,M.S., Liu,J.S., Neuwald,A.F. and Wootton,J.C. (1993) Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, **262**, 208–214.

Press,W.H., Flannery,B.P., Teukolsky,S.A. and Vetterling,W.T. (1988) *Numerical Recipes in C*. Cambridge University Press, Cambridge.

Schneider,T.D., Stormo,G.D., Gold,L. and Ehrenfeucht,A. (1986) Information content of binding sites on nucleotide sequences. *J. Mol. Biol.*, **188**, 415–431.

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

Staden,R. (1984) Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res.*, **12**, 505–519.

Staden,R. (1989) Methods for calculating the probabilities of finding patterns in sequences. *Comput. Appl. Biosci.*, **5**, 89–96.

Stormo,G.D. (1988) Computer methods for analyzing sequence recognition of nucleic acids. *Annu. Rev. Biophys. Biophys. Chem.*, **17**, 241–263.

Stormo,G.D. and Fields,D.S. (1998) Specificity, free energy and information content in protein–DNA interactions. *Trends Biochem. Sci.*, **23**, 109–113.

Stormo,G.D. and Hartzell,G.W. III (1989) Identifying protein-binding sites from unaligned DNA fragments. *Proc. Natl Acad. Sci. USA*, **86**, 1183–1187.

Stormo,G.D. and Yoshioka,M. (1991) Specificity of the Mnt protein determined by binding to randomized operators. *Proc. Natl Acad. Sci. USA*, **88**, 5699–5703.

Stormo,G.D., Schneider,T.D. and Gold.L. (1986) Quantitative analysis of the relationship between nucleotide sequence and functional activity. *Nucleic Acids Res.*, **14**, 6661–6679.

Tatusov,R.L., Altschul,S.F. and Koonin,E.V. (1994) Detection of conserved segments in proteins: Iterative scanning of sequence databases with alignment blocks. *Proc. Natl Acad. Sci. USA*, **91**, 12091–12095.

Wilks,S.S. (1938) The large-sample distribution of the likelihood ratio for testing composite hypotheses. *Ann. Math. Stat.*, **9**, 60–62.