Chapter III. Statistical analysis of sequences

Here we are taking the statistical analysis of sequences (that we started when considering the significance of the alignment) one step further. The high point of the discussion is the most popular alignment algorithm: BLAST (Basic Local Alignment Search Tool). The theory of BLAST is quite complex and full account of the theory is beyond the scope of the present discussion. Nevertheless, we will outline the main ideas and the formulas that are used. We start with yet another look at the statistical argument that we used when evaluating the significance of the scores.

The score of an alignment \( T = \sum_{i=1}^{n} s(\tilde{a}_i, \tilde{b}_i) \) is a sum of individual entries to a substitution matrix, substitutions that may include gaps. The BLAST algorithm suggests an approximate statistical estimate for the significance of the above score. Before going into a deep discussion on BLAST it is useful to outline the conceptual analogy to the \( Z \) score analysis and the relative advantages and disadvantages of the two approaches. In the \( Z \) score formulation we test if the score is significantly higher than a typical score of a random sequence. Alternatively, we ask what is the probability to obtain by chance a \( Z \) score higher than a threshold value \( Z_{th} \)? By “high score by chance” we mean a high scoring alignment of a probe sequence against a random sequence sampled from a distribution of individual amino acids, \( p(a_i) \).

Let the probability of observing a \( Z \) score between \( Z \) and \( Z + dZ \) by chance be \( p_Z(Z)dz \) (\( p(Z) \) is the probability density). The answer to the above question is therefore

\[
P_Z(Z > Z_{th}) = \int_{Z_{th}}^{\infty} p_Z(Z)dz
\]

The smaller is \( P_Z(Z > Z_{th}) \) the less likely it is that the observed score \( Z \) was obtained by chance. This is in principle a simple test for significance. However, there is a complication in practice, which is the unknown functional form of \( p(Z) \). A possible solution to the problem is based on numerical experiments. We may compute a large sample of alignments of pairs of sequences that are not related. The sample will be used to estimate the probability density (e.g. by counting the number of times that we observed \( Z \) scores between \( Z \) and \( Z + \Delta Z \), \( n(Z) \), dividing it by the total number of alignments \( N \), and by the interval, \( \Delta Z \)).

The consideration of the dimensionless entity, \( Z \), versus the direct score, \( T \), is especially useful in estimates of the probability density. The reference to a single variable, which does not depend strongly on the sequence length, makes the numerical estimates of the model easier and more straightforward. For example, the score, \( T \), clearly depends on the length and so is the probability density \( p_T(T) \).
Note that in the numerical evaluation described above we use the term “unrelated sequences”. These are not necessarily “random” sequences sampled from distribution of individual amino acids but are true protein sequences that are unrelated to the probe sequence. In other words we changed the reference or the background distribution to reflect the distribution of real proteins. The use of random sequences enters only in the second phase when we evaluate the $Z$ score of one alignment (a pair of sequences). One of the protein sequences undergoes shuffling (randomization of the positions of the amino acids) and an optimal score is computed between the probe and the random sequences. The optimal scores of alignments against random sequences (derived from a single match of the probe sequence into one of the sequences in the database) are used to compute one $Z$ score.

The distribution function, $P(Z > Z_{th})$, can be computed only once prior to any calculation and used ever after. Besides the probability of the prediction being a false positive it is also possible to estimate the probability of being true positive. This can be done only by numerical experiments since there is no analytical theory for true positives. For that purpose we compute the distribution function of $Z$ scores of alignments between related sequences. That is, we ask the double question (and provide numerical estimate): What is the probability that the computed $Z$ score is a false positive and at the same time is a true positive? We hope that the answer to the first question is very low and the answer to the second question is very high.

Ideally the distribution of the false positive and true positive will have zero overlap.

![Figure X: Schematic drawing of overlapping distribution of false and true positives. Ideally we should have a score that is “only” true or “only” false. Typically we accept some probability of false positives to minimize the lost of true positives. The score $Z_{th}$ determines the selection boundary.](image)

In practice, however, this is not the case. The choice of the threshold score $Z_{th}$ that we use to decide if to accept the prediction as true is done to minimize $P(Z > Z_{th})$ and maximize $P^{true}(Z > Z_{th})$. Clearly, the two functions provide complementary information.

The above procedure that is based on numerical calculations of the distribution functions for false and true positives and the careful selection of a threshold value is very reliable. However, (and this is a BIG however), the process is expensive and is difficult to use in
large-scale prediction schemes. The reason is not the calculation of the distribution function that is done only once and used ever after but the calculation of the \( Z \) score itself. Each calculation of a \( Z \) score requires the alignment of tens to a thousand of random sequences. If a single alignment costs about 0.01 to 0.1 second on a 700 MHz PC then a comparison of two sequences (including the \( Z \) score estimate) will take tens of seconds. And a search of one sequence against a small database of (say) ten thousand proteins will take about a day. Of course large-scale comparisons between many genes against larger databases becomes impossible with such timing. One way of reducing the computational efforts is to compute the \( Z \) scores only for high scoring alignments. However, in order not to miss true positive we still need to examine many high score alignments and the relief provided by the above heuristic is limited.

If the task at hand is of genomic scale analysis, namely, the study of ten to hundreds of millions of alignments, then even dynamic programming (computing only the \( T \) scores) can be too expensive.

An intermediate conclusion is therefore that the statistical arguments so far have led to more reliable predictions but not to more efficient calculations.

It is possible to use the general idea of statistical significance to design a more efficient alignment algorithm. The twist here is not to check the statistical significance of an optimal alignment that was obtained by dynamic programming, but to create an approximate alignment using statistical consideration. We design an approximate alignment procedure that will pick alignments with high statistical significance. As explained below the resulting algorithm is considerably more efficient than dynamic programming at the expense of using approximations. On the hand, the incorporation of statistical arguments into the alignment procedure makes the final decision, (true or false positive?), better than a decision that is based only on the \( T \) score. Hence even if the alignment is not optimal the assessment that the two sequences are indeed related by statistical significance is typically pretty good.

We consider first the score \( T \) of an alignment and the probability density of the score \( p(T) \). The \( T \) score is considerably less expensive to compute compared to the \( Z \) score, and in that sense it is more attractive. However, the \( T \) score depends strongly on a number of parameters, for example, the sequence length. It is necessary to develop a theory that will examine the dependence of the score \( T \) on different alignment parameters. This is one achievement of the BLAST algorithm: the development of a statistical theory of the \( T \) scores. We shall discuss the theory later after understanding how the efficiency of match finding is achieved.

Even if we have an exact theory of the statistical significance of a score (and we do not, the BLAST theory is approximate), we still need to select (efficiently) a high scoring alignment in order to assess its significance. A clever idea of BLAST is to perform a search for high scoring short segments using gapless local alignments. The statistical significance test makes it possible to estimate if the short matches are meaningful and worth exploring further.
Efficient scanning of sequences in BLAST

Consider for example the short segment WWWW that is found in both the probe sequence and one of the sequences in the database. Even though the match is found in fragments that are short (and typically shorter segments are less significant), here it is likely to be significant. Tryptophan is a rare residue, which is unlikely to be mutated by another. Therefore, if we have a match for four tryptophans the match is unlikely to be by chance, and is more likely to indicate true relationship between the two sequences. Note that these short segments for which we find matches need not be identical. For example in the above example we may consider WFWW as also a match using scores from the usual substitution matrices. The quantification of “likely” and “unlikely” is at the core of the BLAST statistical estimates. Let us accept for the moment that we can quantify the statistical significance of matching of short segments and consider the problem of efficiency.

Matches for short segments can be search efficiently. Many technical adjustments to the idea we describe below are possible, however for simplicity we focused on the most obvious solution rather than on the most efficient.

One simple idea is to use hash tables and to pre-process the database of annotated proteins (we consider now the problem of seeking a match of an unknown sequence against a large database). Consider a segment of length four. There are $2^4 = 160000$ possible different segments. This number is large but not impossible. We prepare pointers for all possible four characters of the probe sequence. The database is scanned (number of operations of order of $O(N)$ where $N$ is the size of the database), and every fragment of length 4 of the database is immediately tested against the probe sequence using the pointers. We comment that with advance hard disks with rapid access or large memory it is possible to preprocess the entire database and to arrange pointers to the locations of all fragments in the database. In that case the probe sequence analysis will include the calculation of the pointers that will immediately bring us to the matches at the large database. The number of operation is therefore $O(L)$ where $L$ is the length of the probe sequence! Sounds great. Nevertheless, the limiting factor in this case may be the rate of disk access.

The pointer is not limited to identical matches but can also point to all other possible matches that score above a certain threshold $T^r$. Clearly a high threshold will make our life considerably simpler since only a relatively small number of matches will be found that will require further examination in the next step. However, the small number of matches will make the next phase of extending the match, considerably more difficult. The choice of the threshold $T^r$ is a compromise.

Once high scoring segments were identified (hopefully their number is not too large…) the next step is to try to extend them beyond the pre-determined size of a fragment (in our discussion so far it was 4) while maintaining the significance of the (high scoring) alignment. It is important to emphasize that we are left now with considerably smaller number of sequence pairs to probe, which makes the efficient execution of the first step
even more important. The extension of the high scoring fragment can be made (again for example) using dynamic programming and gaps, attempting to link more than one high scoring segment. Hence, it is useful to examine not only individual high scoring segments but also to consider (or put even higher significance) those that are close to each other. The disadvantage of using dynamics programming is the slowing down of the search procedure. In practice direct extension of the matched parts (no gaps) seems to detect sequence relationships quite efficiently, so it is not obvious if the (expensive) implementation of dynamic programming was worth it.

It is clear that the cost of the second step should depend only weakly on the database size (the number of potential matches that we find will depend on the database size). As a result BLAST searches are efficient.

**Brief statement of BLAST statistical framework**

The theory behind BLAST that we shall considers next provide us with an estimate of the probability that two random sequences of length \( n \) and \( m \) will score more than \( T_r \). To make our match unusual and more likely to be biologically significance this probability better be small.

In the present approach we restrict ourselves to local alignments only

We consider the score, \( T \), aligning two sequences \( A = a_1...a_n \) and \( B = b_1...b_n \),

\[
T = \sum_i s(a_i, b_i)
\]

where the matrix elements, \( s(a_i, b_i) \), are the appropriate entries to the BLOSUM matrix. We assume that the entries are uncorrelated and therefore the score \( T \) is a sum of uncorrelated random numbers that are sampled from the same probability function. We wish to determine the probability that an observed score \( T_{obs} \) was obtained by chance.

It is useful to think on the score as a random walk in which the change from \( T_i \) to \( T_{i+1} \) are the changes induced by one step of the walker. Since the alignment we consider is local the length of the walk is not predetermined to begin with, and nor is the score. We terminate the alignment when further build-up of the alignment does not seem to be helpful. In our case it is when \( T_i \) reaches a negative value (-1). Previously we terminate at the value of zero. The choice of different (low) termination values depends on the choice of the substitution matrix. At the least we require that the average value of the substitution matrix (over all elements), \( \langle s(a, b) \rangle = \sum p_a p_b s(a, b) \), is negative. The probabilities \( p_a \) or \( p_b \) are the “background” probabilities for individual amino acids. The average should be 0 since for sufficiently large \( n \) the score of the alignment is roughly \( T \approx n \cdot \langle s(a, b) \rangle \). To ensure that the length of the alignment is finite the average of the substitution matrix, \( \langle s(a, b) \rangle \) must be negative, otherwise the score and the length may grow to infinite.
A schematic presentation of a random walk that represent a (random) alignment. The alignment starts at zero and then terminates when it reaches the value of \(-1\). No termination for an upper bound is assumed, however, since the substitution matrix is negative on the average, the alignment should terminate at finite length \(n\).

In BLAST we address the following questions:

(i) What is the probability of obtaining a maximum score of an alignment, \(T_T\), by chance before the alignment reaches \(-1\) (i.e. what is the probability that the alignment is not significant).

(ii) What is the distribution of alignment lengths before they are terminated (by hitting the absorbing boundary at \(-1\))

We will not follow the theory in all its glory, since some of the arguments are too complex to be included in the present discussion. However, we will outline a few simple examples demonstrating the main idea behind the BLAST approach. Before continuing we provide first the main result of the statistical theory.

The probability to obtain by chance a score \(T\) larger or equal to \(T_T\) is

\[
P (T > T_T) = 1 - e^{-y}
\]

where \(y = K \cdot m \cdot n \cdot \exp[\lambda T_T]\)

It is expected that the maximal score by chance will depend on the length of the sequence, and indeed \(m\) and \(n\) are the lengths of the two sequences. There are two parameters in the theory, \(K\) and \(\lambda\), that require further discussion. For example, \(\lambda\), which is a simpler parameter, is determined by the expression

\[
\sum_i p_i p_j \cdot \exp[\lambda s_{ij}] = 1
\]

Hence, the parameter, \(\lambda\), determines the scale of the substitution matrix.
It is useful to think on the score as a result of a random walk. In that case we may ask what is the probability that the walk will be terminated at a given upper bound instead of a lower bound. Hence we consider a random walk between two absorbing boundaries. The lower bound is $a$ and the upper boundary is $b$ (note that in BLAST we consider only the lower boundary $a$ which is set to $-1$. The upper boundary here is added for convenience). We start our walk in the space of scores at zero. When no amino acids are aligned against each other (the beginning) then the total score is zero by definition.

A schematic presentation of a walk in score space. We always start at zero and terminate either at the lower boundary $a$, or the upper boundary $b$. We ask what is the probability $f_0$ that the walk will terminate at $b$ and not on $a$. Clearly the higher is $b$ (keeping $a$ fixed) the lower is the probability of hitting $b$ before $a$.

The probability that an alignment will reach a maximum value
We consider a walk (extension of the length of the alignment) that starts at score zero and at length zero. The probability and the magnitude of a step (an element of the BLOSUM matrix for a pair of amino acids) have significant variations for real data. However, to keep the discussion below simple we consider a model in which only steps of $\pm 1$ are allowed. Of course there are many more possibilities for real data but we can still imagine dividing the amino acids into two groups: hydrophobic H and hydrophilic P. If the pair under consideration is of the same type (i.e. H/H or P/P), then the score is set to $+1$, if it is a miss (H/P or P/H) then the score is $-1$. Since the H/P model was used successfully in a number of simplified and semi-quantitative models, it is expected to work similarly in the present case.

(Note however, that there is a fundamental problem in the above suggestion if all the pairs are equally probable. A possible correction is to set the score of P/P to zero. Can you explain why?)

The probability of going a step up is $p$ and a step down is $q$. Let $f_i$ be the probability that the walk terminates at the upper bound $b$ instead of the lower bound $a$ starting from position $i$. After a single step we may reach with probability $p$ the $i+1$ position and with a probability $q$ the $i-1$ position. Since the probability of termination at the upper boundary should conserve, we have

$$f_i = pf_{i+1} + qf_{i-1}$$
with the boundary conditions: \( f_a = 0 \) and \( f_b = 1 \).

As usual it is easy to guess a solution of the type \( f_i = \exp[i \cdot \theta] \) for the above homogenous equation. We have

\[
\exp[i \cdot \theta] = p \exp[(i + 1) \theta] + q \exp[(i - 1) \theta]
\]

multiplying on both sides by \( \exp[(1 - i) \theta] \) we have

\[
p \exp[2 \theta] - \exp[\theta] + q = 0
\]

for \( p \neq q \) we have two solutions: \( \theta = 0 \) and \( \theta = \log(q/p) \). The general solution is a linear combination of the two, which is

\[
f_i = A_1 + A_2 \exp[i \cdot \log(q/p)]
\]

The coefficient \( A_1 \) and \( A_2 \) are determined by the boundary conditions \( f_a = 0 \) and \( f_b = 1 \)

We write the solution below for the probability starting at zero.

\[
f_0 = \frac{1 - \exp(a \cdot \log(q/p))}{\exp(b \cdot \log(q/p)) - \exp(a \cdot \log(q/p))}
\]

For sufficiently high \( b \) \( (b \gg a) \) we have (we also set \( a = -1 \))

\[
f_0 = (1 - p/q) \exp(-b \cdot \log(q/p))
\]

Note that we already have a condition: \( q > p \). Explained. Without this condition the results of the last equation will be meaningless. The probability of hitting an upper boundary is exponentially small. Of course this is what we should expect from a random walk. In sequence alignments it means that high scoring segments can easily have vanishing small probabilities of being false positive. This probability is the geometric distribution and makes the core of the statistical arguments about random alignments.

**Length of an alignment**

Another fun question that we can ask is what is the typical length of the alignment.

Assume that we are at position \( i \) like before and the number of steps that the walk takes (before terminating either at \( -1 \), or at \( b \)) is \( L_i \). After one step the walk will be at position \( i - 1 \) with a probability \( q \) and at position \( i + 1 \) with a probability \( p \). The length of remaining walk after this single step is \( L_i - 1 \). Summarizing in a difference equation we have

\[
L_i - 1 = qL_{i-1} + pL_{i+1}
\]

Since the equation now is inhomogeneous, the solution is a linear combination of the homogenous solutions (that we saw before) and special solution. We have

\[
L_i = \frac{i}{q-p} + A_1 + A_2 \exp[i \cdot \log(q/p)]
\]

If the walk starts at \( a = -1 \) or at \( b \) then it is terminated immediately and the walk length is zero. We therefore have the conditions \( L_a = L_b = 0 \), which are sufficient to obtain the final solution for the length of the alignment starting at zero.
The extreme value distribution

A single alignment may include more than a single maximum before hitting the termination point. Alignment (and a random walk) can go up and down several times before reaching the absorbing boundary \( a \). We usually pick the local alignment with the largest score and trace it back from that position. It is therefore insufficient to find an arbitrary maximum of the alignment. We need instead to determine the maximum of the maxima. Consideration of the probability of the maximum of the maxima requires a little bit of more work that is outlined below.

Consider a set of \( N \) random numbers that are independent and sampled from the same distribution function \(- \{T_1, T_2, \ldots, T_N\} \). With respect to the alignments we assume that the maxima along the walk are independent. One of these numbers is larger than the rest, and we call it \( T_{\text{max}} \). The random numbers (local maxima of the alignment) \( T_i \)-s are assumed for simplicity to be continuous and sampled from a probability density, \( p(T) \);

\[
T \in [-\infty, \infty] : \int_{-\infty}^{\infty} p(T) dT = 1.
\]

We are asked to compute the probability density of \( T_{\text{max}} \), \( p_{\text{max}}(T_{\text{max}}) \). This probability can be estimated using computer experiments as follows:

Generate \( N \) random numbers sampled from the \( p(T) \) probability density. From the \( N \) random numbers we select the maximum, \( T_{\text{max}}(1) \). This experiment is repeated; say \( L \) times, to produce \( L \) maximal numbers \( \{T_{\text{max}}(j)\}_{j=1}^{L} \). The resulting numbers are histogrammed to estimate their probability density, \( p(T_{\text{max}}) \).

Analytically the following procedure is used to estimate \( p(T_{\text{max}}) \) from \( p(T) \). We consider the distribution functions \( Q_{\text{max}}(T) \) and \( Q(T) \) that are defined as follows

\[
Q_{\text{max}}(T) = \int_{-\infty}^{T} p_{\text{max}}(T) dT
\]

\[
Q(T) = \int_{-\infty}^{T} p(T) dT
\]

The relationships between the \( Q \)s and the \( p \)s are obvious. However, we only know \( p(T) \) and therefore also \( Q(T) \). \( Q_{\text{max}}(T) \) is the probability of sampling a \( T_{\text{max}} \) that is smaller than \( T \). The probability that the maximum value of the set, \( T_{\text{max}} \), is less than a particular value, \( T \), is the same as the probability that each member of the set - \( T_i \) is also smaller than \( T \), that is

\[
L_0 = \frac{f_o \cdot b + (1 - f_o) a}{q - p}
\]
\[ Q(T_{\text{max}} \leq T) = \prod_i [Q(T_i \leq T)] \]

In terms of the probability density, \[ p_{\text{max}}(T) \left( Q(T_{\text{max}} \leq T) = \int_{-\infty}^T p_{\text{max}}(T_{\text{max}})dT_{\text{max}} \right) , \] we can write
\[ p_{\text{max}}(T) = \sum_i \left( \frac{dQ(T_i < T)}{dT_{\text{max}}} \right) \prod_{j \neq i} Q(T_j < T) \]

If all the \[ Q(T_j < T) \] are the same and are now denoted \( Q(T) \), the expression is simplified to
\[ p_{\text{max}}(T) = n \cdot p(T) \cdot Q(T)^{n-1} \]

To demonstrate that the new probability density for the maximum number, \( p_{\text{max}}(T) \), is quite different from \( p(T) \) let us consider a simple example. Let \( p(T) \) be a constant, \( 1/L \), between 0 and \( L \) (a uniform probability density). Then \( Q(T) \) is
\[ Q(T) = \int p(T')dT' = \left( \frac{1}{L} \right) \cdot T \]
and \( p_{\text{max}}(T) \) is
\[ p_{\text{max}}(T) = \left( \frac{n}{L} \right)^n \cdot T^{n-1} \]

Note also that in contrast to our prior guess of a Gaussian distribution, the corresponding \( p_{\text{max}}(T) \) if \( p(T) \) is a Gaussian is not yet another Gaussian.

The probability density that we are really interested in (in the context of BLAST) is the geometric distribution. Here we only state the results of a (conceptually) similar analysis to what we did. Let \( T_{\text{max}} \) be the maximum of \( n \) random numbers distributed according to the geometric distribution. The probability of obtaining a random number \( T \) by chance that is larger than \( T_{\text{max}} \) is bound by the so-called extreme value distribution
\[ \exp[-n\exp(-\lambda T)] \leq P(T_{\text{max}} \leq T) \leq \exp[-n\exp(-\lambda (T+1))] \]
This asymptotic formula is essentially the same formula we wrote BLAST
\[ P(T > T_r) = 1 - e^{-y} \]

where \( y = K \cdot m \cdot n \cdot \exp[-\lambda T_r] \)

Sometimes we also encounter the \( E \)-value defined as, \( E = -\log(1 - p) \)
This probability is also called the P-value. The smaller is the P-value, the more significant is the score (i.e. it is less likely to be obtained by chance). The parameter \( K \) is determined (approximately) by 
\[ K = \left( \frac{C}{L} \right) \exp(-\lambda), \]
where \( C \) is the coefficient of the geometric distribution analogous to the estimate we made earlier 
\[ P(T_{\text{max}} \geq T) = C \exp(-\log(q/p) \cdot T) \]
and \( L \) is the typical length of the alignment between sequential maxima. We did not discuss the calculation of \( L \).

This concludes our brief description of the BLAST algorithm. To summarize, BLAST uses statistical theory to estimate what is the optimal score of two random sequences depending on the length of the alignment and the properties of the substitution matrix. Whatever score we obtain when aligning two real sequences it needs to be much higher than the score of random sequences in order to be significant. BLAST provides a theoretical estimate what are the chances that a random sequence will score the same. The searching for significant matching segments can be done efficiently using hash tables and related computer science techniques.

**Use of multiple sequence alignments in signal enhancement**

So far we discussed only alignments of pairs of sequences. There is significant information in multiple alignments of related proteins. For example, if a residue is conserved over a range of related sequence it is more likely to be important for the well being of that protein. On the other hand if a specific position in the sequence has amino acids all over the map (e.g. in ten sequences different amino acid is observed at that position) then this site along the sequence is expected to be less significant and should contribute little to the overall score of the alignment. It is suggestive to define an average score \( \langle T \rangle \) such that 
\[ \langle T \rangle = \sum_{i,\alpha} p(i,\alpha) s(\alpha,\beta) \]
where \( p(i,\alpha) \) is the probability of finding amino acid \( \alpha \) at the \( i \) column of the multiple sequence alignment, and \( \beta \) is the amino acid from the database sequence we compare the multiple sequence alignment to. Alternative way of computing the score is 
\[ \langle T \rangle = \sum_{i,\beta} \log \left( \frac{p(i,\beta)}{p(\beta)} \right) \]
which is putting emphasis on the probability of observing amino acid \( \beta \) at a specific column.

We hope to estimate the above probability directly from the multiple sequence alignment.

A multiple sequence alignment will look something like
\[ a_1 \ldots a_k \ldots a_n \]
\[ b_1 \ldots b_k \ldots b_n \]
\[ c_1 \ldots c_k \ldots c_n \]
\[ d_1 \ldots \ldots d_m \]
\[ \ldots e_1 \ldots e_m \]

We do not discuss here how the multiple sequence alignment was obtained and assume that it is given. Multiple sequence alignment is actually a hard problem that we shall discuss later. Typically we are able to find and align an order of 10 different sequences. It is clear that we will suffer from a severe sampling problem. Since there are twenty amino acids it is impossible that the limited sampling described above will yield frequencies that are good approximation to the true probabilities. For example, it is possible that amino acid F will not appear at all at a given column. This will immediately creates a downgrading for a novel sequence that includes F in that position.

It is possible to overcome (partially) the under sampling problem by using the “null” hypothesis. Adding more statistics from a known distribution of amino acids not necessarily related to the sequences at hand. We compute the probability \( p(i,\alpha) \) as

\[
p(i,\alpha) = \frac{N_i}{N_i + B_i} \cdot \frac{n_{ia}}{n_i} + \frac{B_i}{N_i + B_i} \cdot \frac{b_{ia}}{B_i}
\]

where \( B_i \) is the total number of pseudo-counts at column \( i \) and \( b_{ia} \) is total number of pseudo count at column \( i \) of amino acid \( \alpha \). Similarly \( N_i \) (and \( n_{ia} \)) are the actual numbers of sequences (and amino acid type \( \alpha \)) we have in the multiple sequence alignment at column \( i \). The pseudo counting parameters are unknown.

The most straightforward approach of estimating \( b_{ia} \) is \( b_{ia} = p(\alpha) \cdot B_i \). Hence, we are getting \( p(\alpha) \) from the known (general) distribution of amino acids. This choice is not optimal since it is not using at all the information we have so far. It is true, the information is limited, but it is still more than zero. A possible way of generating pseudo counts is

\[
b_{ia} = B_i \sum_{\beta} p(\alpha, \beta) \cdot p(i, \beta)
\]

The probability, \( p(i, \beta) \), is computed directly from the multiple sequence alignment. It is the raw frequencies extracted from the limited number of sequences that we have. The raw frequencies are multiplied by a conditional probability of pairs of amino acids, \( p(\alpha, \beta) \), which is extracted (for example) from the BLOSUM matrix. The substitution probabilities of the BLOSUM matrix are not zero, and therefore the probability of observing any amino acid will not be zero. Though, in principle, some of the amino acids will be highly unlikely. For example if we have only W at column \( i \), the probability of
observing $R$ will be particularly small. The proposed way of generating pseudo-counts is attractive since it is using the known statistics of amino acids to generate additional counts that we were likely to miss.

How should we handle the total number of pseudo counts, $B_i$? We have a number of expectations. For example, if we have a LOT of data then $B_i$ should decreases. Also if there is a great diversity of amino acids at a given column then pseudo counts are more important. However, if we get ten times exactly the same amino acid, it is less likely that we miss something and that we need to generate a lot of pseudo-counters.

A common sense choice is $B_i = N \cdot V_i$, where $N$ is an empirical constant and $V_i$ is the measure of amino acid diversity at position $i$. 