CH1: BLAST and Karlin-Altschul Statistics

CS 182 Spring 2022

Scribes (from past years): `eyouth`, `daluthge`, `berdogdu`, `kclark5`, `ewoo`, `ale22`, `cjordan3`, `jzhan204`

Compiled & edited by `eyouth`, `smaffa`

Minor edits made in 2024 by `cbaker20`

Reach out to cs1820tas@lists.brown.edu for any clarifications or corrections.

Overview of BLAST

BLAST (*B*asic *L*ocal *A*lignment *S*earch *T*ool) is one of the most widely-used bioinformatics algorithms. It compares a query sequence with target sequences stored in a database or library to identify homologous regions, which has earned it a reputation as the "Google of biological research".

Input: A query sequence Q and a database of sequences DB

Output: All regions of DB which have significant ungapped local alignment scores with Q

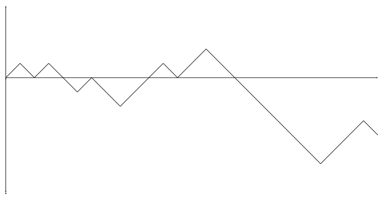
We seek the alignments with maximal score, as they indicate regions of the database which exhibit high similarity with the query sequence. BLAST can therefore be used to identify homologous regions across the genomes or proteomes of different species!

Scoring Matrices and Random Walks

Scoring matrices are of central importance to alignment algorithms such as BLAST. Commonly-employed examples include PAM (*P*oint *A*ccepted *M*utation) matrices (developed by Margaret Dayhoff in the 1970s) and BLOSUM (*BLO*cks *SU*bstitution *M*atrix) matrices (developed by Henikoff & Henikoff in the 1990s).

The "rolling" alignment score between the query sequence and the database can be visualized as a *random walk* in which matches increase the score and mismatches decrease it. A simple example using a uniform $+1/-1$ scoring scheme for two DNA sequences is shown below:

```
ggagactgtagacagctaatgctata
| |  |  ||| ||         |||
gaacgccctagccacgagccattatc
```



Test statistics for evaluation of alignment significance can be obtained at points along the random walk. Several additional features of random walks are of interest.

Defn: A *ladder point* is a point on the random walk which is lower than any previously-reached point.

Defn: An *excursion* is a point on the random walk which is the highest point between two ladder points.

Random walks for protein sequences are often more complex due to greater variability in scores between pairs of amino acids. Significance can be evaluated through hypothesis testing (e.g., evaluating the likelihood of an alignment score being generated by random chance). The probability of observing a given amino acid a_i is p_i, with i ranging from 1 to 20. These probabilities form a distribution; i.e.,

$$\sum_{i=1}^{20} p_i = 1$$

Under the null hypothesis (alignment due to random chance), the probability of observing a given pair of amino acids (a_i, a_j) is simply $p_i p_j$. Under the alternative hypothesis (alignment due to non-random factors such as convergent evolution), this probability is instead q_{ij}. Karlin-Altschul statistics provide a means of deriving significance scores from comparisons of these probabilities.

The objective of BLAST is to identify *high-scoring segment pairs* (HSPs). The significance of each HSP can be evaluated as the log-likelihood of the probabilities of observing the alignment under the alternative and null hypotheses. Highly significant HSPs are returned as the "best" alignments between the query sequence and the database.

The BLAST Algorithm

BLAST represents a more efficient alternative to the Smith-Waterman algorithm for local sequence alignment (introduced previously in CS 181). For aligning two sequences of length n, an n-by-n dynamic programming table is constructed. The time complexity for the algorithm is quadratic; i.e., $O(n^2)$.

The Smith-Waterman algorithm is "global" in the sense that it searches over all possible local alignments and finds the optimum alignment in the space of local alignments. In contrast, the BLAST algorithm looks for local optima. The time complexity for BLAST is linear in the total size of the database; i.e., $O(n)$.

The BLAST algorithm has three phases: *seeding*, *extension*, and *evaluation*. The seeding and extension phases are built on algorithmic (deterministic) methods, while the evaluation phase relies on statistical (probabilistic) methods.

Note that there are multiple BLAST programs. Each BLAST program takes as input a query sequence Q, a database of sequences DB, and a similarity matrix M (e.g. PAM250, BLOSUM62). Some examples of BLAST programs are as follows:

Program	Query (Q)	Database (DB)
BLASTN	DNA	DNA
BLASTP	protein	protein
BLASTX	DNA	protein
TBLASTN	protein	DNA
TBLASTX	protein	protein

To reiterate: Smith-Waterman only finds the *global optimum*, while BLAST can find multiple *local optima*. Yet BLAST is more efficient!

Defn: *Sensitivity* is the proportion of "true positives" (i.e., HSPs) that are correctly identified (i.e., the probability that some HSP is a true positive, given it is an HSP selected by BLAST).

Defn: *Specificity* is the proportion of "true negatives" that are correctly identified (i.e., the probability that some segment pair is a true negative, given it is not selected by BLAST). We sometimes care about $1-$ *Specificity*, which indirectly tells us how many true positives we are falsely predicting to be negative.

Phase 1: Seeding

Seeding is based on "words", or k-mers. For example, DNA codons are 3-mers (words of length 3).

Defn: The *neighborhood* of a k-mer w is the set of all words of the same length whose ungapped alignment score with w exceeds some threshold T.

Defn: A *hit* is a word which is in the neighborhood of a word of interest.

Note that the set of "hits" depends on the choice of scoring matrix. For example, consider a 3-mer w drawn from the alphabet of amino acids. The score threshold may be set at $T = 14$. To find all relevant "hits", the ungapped alignment scores between w and all other 3-mers in the protein alphabet are computed using a given scoring matrix. Using PAM, such a list may look like this for some specific word w:

$$\{RGD\ (18), KGD\ (17), RGN\ (16), KGE\ (15)\}$$

Any 3-mers for which the score is less than or equal to 14 are excluded from the set. If instead we use BLOSUM, however, an exclusionary list may instead look like this for a word w':

$$\{KGD\ (14), QGD\ (13), RGE\ (13)\}$$

In this example, all three 3-mers above would be excluded from the final set of "hits", as none of their alignment scores with w exceed the threshold T.

In BLASTN, the parameter k is set to $k = 7$, and the parameter T is not used (i.e. the neighborhood for a word is the set of all words of the same size).

In BLASTP, the parameter k is set to $k = 3$, and the parameter T is set to $T = 999$.

The selection of T is dependent on the values in the scoring matrix and the desired balance between sensitivity (proportion of true alignments identified) and speed.

Phase 2: Extension

Extension involves extending all "hits" identified in the seeding phase to find longer high-scoring alignments (HSPs). A "budget" X is set as a cutoff threshold to terminate extension in either direction from a starting seed. Specifically, extension continues until the current alignment score has decreased from the maximum observed thus far by an amount which exceeds X (i.e., X is the maximum-allowed score deficit permitted). When X is exceeded, the alignment is trimmed back to the base at which the most recent maximal score was reached. Intuitively, this enables the algorithm to extend from one "island" of high-scoring alignment to another across (short) regions of dissimilarity.

Phase 3: Evaluation

Once extension has been carried out in both directions from the original seed, a number of high-scoring local alignments are isolated and evaluated for statistical significance. The statistical theory is known as *Karlin-Altschul statistics*. A score threshold S may then be used to separate high-scoring and low-scoring alignments.

Karlin-Altschul Statistics

Karlin-Altschul statistics are employed to compute p-values for local alignments returned by BLAST, using the mathematics of random walk theory.

Random Walk Theory

Consider the simple random walk on the set of integers $\mathbb{Z}$. Let $a, b \in \mathbb{Z}$ be integers such that $a < b$.

Set the initial position of the random walk at an arbitrary $h \in \mathbb{Z}$ such that $a < h < b$. The walk moves independently of the previous history of the process; i.e., it has the Markov property. It moves to the right with probability p and to the left with probability $q = 1 - p$, and stops when it reaches either a or b.

Two fundamental questions arise:

1. What is the probability that eventually the walk finishes at b?
2. What is the expected number of steps until the walk stops?

Solution to Question 1

Let w_h be the probability that the simple random walk eventually ends at b (rather than a), given that its initial position is h as defined above. w_h is the *absorption probability* for point b. Then a homogeneous difference equation arises with the following boundary conditions:

$$\begin{aligned} w_h &= pw_{h+1} + qw_{h-1} \\ w_a &= 0 \\ w_b &= 1 \end{aligned}$$

Postulate that this boundary value problem has a solution of the form

$$w_h = e^{\theta h}$$

for some constant θ (where e represents the base of the natural logarithm). Then substituting and multiplying through by $e^{\theta - \theta h}$ yields

$$e^{\theta h} = pe^{\theta(h+1)} + qe^{\theta(h-1)}$$

$$\implies e^{\theta - \theta h} e^{\theta h} = pe^{\theta - \theta h} e^{\theta(h+1)} + qe^{\theta - \theta h} e^{\theta(h-1)}$$

$$\implies e^{\theta} = pe^{2\theta} + q$$

Letting $x = e^{\theta}$ yields a quadratic equation:

$$px^2 - x + q = 0$$

This equation has two solutions: $x = 1$ or $x = \frac{q}{p}$ (to verify this, substitute the values into the equation above and recall that $p + q = 1$). Thus,

$$\begin{aligned} e^{\theta} = 1 &\implies \theta = 0 && \implies w_h = 1 \\ e^{\theta} = \frac{q}{p} &\implies \theta = \log\left(\frac{q}{p}\right) && \implies w_h = e^{\theta^* h}, \text{ where } \theta^* = \log\left(\frac{q}{p}\right) \end{aligned}$$

The latter case is of interest, and there are two possibilities: $p = q$ or $p \neq q$. Focusing on the subcase in which $p \neq q$ (as $p = q$ is not very interesting in the context of BLAST), the general solution is

$$w_h = c_1 + c_2 e^{\theta^* h}$$

where c_1 and c_2 are constants determined by the boundary conditions. These produce

$$w_h = \frac{e^{\theta^* h} - e^{\theta^* a}}{e^{\theta^* b} - e^{\theta^* a}}$$

Recalling that w_h is the probability that the simple random walk ends at b, let u_h be the probability that the walk ends at a. Since $w_h + u_h = 1$, it follows that

$$u_h = \frac{e^{\theta^* b} - e^{\theta^* h}}{e^{\theta^* b} - e^{\theta^* a}}$$

Solution to Question 2

Let m_h denote the mean (i.e. expected) number of steps taken until the walk stops.

Solving the following *in*homogeneous difference equation:

$$\begin{aligned} m_h - 1 &= pm_{h+1} + qm_{h-1} \\ m_a &= 0 \\ m_b &= 0 \end{aligned}$$

...will result in:

$$m_h = \frac{h-a}{q-p} - \left(\frac{b-a}{q-p}\right)\left(\frac{e^{\theta^* h} - e^{\theta^* a}}{e^{\theta^* b} - e^{\theta^* a}}\right)$$

Karlin-Altschul Statistical Theory

Karlin-Altschul statistical theory rests upon five axioms:

1. A positive score must be possible
2. The expected value of the score must be negative
3. The letters in the sequence of the model are independent and identically distributed (i.i.d.)
4. Sequences are infinitely long
5. Alignments do not contain gaps

The Karlin-Altschul equation is as follows:

$$E = kmne^{-\lambda S}$$

where E is the number of alignments with a given score S expected by chance when comparing a query of length m against a database of size n. λS represents the normalized score of a given alignment, while k is a fixed normalizing constant.

Log Odds Ratio Scores

The log odds ratio (LOD) score of a pair of amino acids (i, j) is defined as follows:

$$\text{LOD}(i,j) = \log_2\left(\frac{\text{observed frequencies given the data}}{\text{expected frequencies under the random model}}\right)$$

The raw score s_{ij} is then defined as follows:

$$s_{ij} = \log\left(\frac{q_{ij}}{p_i p_j}\right)$$

where p_i and p_j denote the frequencies of amino acids i and j, respectively, q_{ij} denotes the observed frequency of the pair (i, j) given the data, and log indicates the natural logarithm (base e).

If the observed frequency q_{ij} is equal to the expected frequency $p_i p_j$, then the LOD is zero. If the LOD is strictly greater than zero (i.e., $q_{ij} > p_i p_j$), then the pairing of amino acids i and j is common. If the LOD is strictly less than zero (i.e., $q_{ij} < p_i p_j$), then the pairing of amino acids i and j is unlikely. LOD values are real numbers, but are rounded to integers to obtain the simpler "raw scores" found in scoring matrices.

PAM and BLOSUM scoring matrices make use of LOD scores to compute the "scores" of alignments between pairs of amino acids. Target frequencies represent the underlying evolutionary models.

Karlin-Altschul statistical theory states that if a scoring scheme satisfies the first two axioms listed above, then the scoring scheme has implicit target frequencies. Thus, they are based on log odds scoring schemes.

Target frequencies are given by the values q_{ij}. By symmetry, a half-sum of the scoring matrix satisfies

$$\sum_{i=1}^{20} \sum_{j=1}^{i} q_{ij} = 1$$

Analogously to the definition of the raw score s_{ij} above, the normalized score is defined as follows:

$$\lambda S_{ij} = \log \left(\frac{q_{ij}}{p_i p_j} \right)$$

$$\implies q_{ij} = p_i p_j e^{\lambda S_{ij}}$$

By substitution, the following condition must hold:

$$\sum_{i=1}^{20} \sum_{j=1}^{i} p_i p_j e^{\lambda S_{ij}} = 1$$

BLAST will solve for λ and present its value in the output report.

The Expected Score of a Scoring Matrix

The expected score of a scoring matrix ($\mathbb{E}$) is the sum of its raw scores weighted by their respective frequencies of occurrence. (Note that this quantity is distinct from E as given by the Karlin-Altschul equation above). By axiom 2 above, the expected score must be negative:

$$\mathbb{E} = \sum_{i=1}^{20} \sum_{j=1}^{i} p_i p_j s_{ij} < 0$$

Relative Entropy

The relative entropy (denoted H) of a scoring matrix summarizes the general properties of the matrix. This value is similar to $\mathbb{E}$, but is calculated from normalized scores instead of raw scores. H is the average number of bits per position in an alignment, and is always positive:

$$H = -\sum_{i=1}^{20} \sum_{j=1}^{i} q_{ij} \lambda S_{ij}$$

Evaluating Alignment Significance

There is a close connection between Karlin-Altschul E-values and p-values:

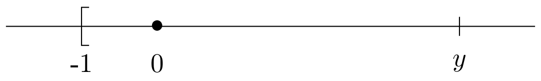
$$p = 1 - e^{-E}$$

$$\implies E = -\log(1-p)$$

In particular, for small values (e.g. values ≤ 0.001), $p \approx E$.

The Probability Distribution of Alignment Scores

The significance of an alignment is informed by the probability of observing an alignment with the given score purely by chance. This problem can be formalized under the random walk theory described above. Since the mean step size (i.e., expected score of aligning two amino acids) is negative, eventually such a random walk will decrease in score. In the general form of the problem, this implies that $a = -1$ is the left endpoint of the interval of interest, $b = y \geq 1$ is the arbitrary right endpoint, and $h = 0$ is the starting point of the random walk:



Again w_h is the quantity of interest: the probability that the random walk eventually reaches some arbitrary positive score y before the cumulative score becomes negative. With a, h and b as defined above, the equation takes the form

$$w_h = \frac{1 - e^{-\theta^*}}{e^{\theta^* y} - e^{-\theta^*}}$$

where $\theta^* = \log\left(\frac{q}{p}\right)$ as defined previously. Since $\theta^* > 0$, the term $e^{\theta^* y}$ dominates the denominator, so

$$\lim_{y \to \infty} w_h = (1 - e^{-\theta^*})e^{-\theta^* y}$$

Defining Y as the maximum (positive) score achieved by the walk, the probability distribution is then

$$P(Y \geq y) \approx ce^{-\theta^* y}$$

where $c = (1 - e^{-\theta^*})$.

This probability distribution can be used to compute the likelihood of observing a random alignment of score y purely by chance.

Let $Y_1, Y_2, \ldots$ be the respective maximum excursion heights of the random walk, relative to the height of the most recent ladder point in each case. Assume that $\{Y_i\}$ are all independent and identically distributed. Then $Y_{\text{MAX}} = \text{MAX}(\{Y_i\})$ is the test statistic for BLAST. Its asymptotic distribution under the null hypothesis is geometric, as shown above. The probability that Y_{MAX} is at most y can be bounded as follows:

$$e^{-ne^{-\lambda y}} \leq P(Y_{\text{MAX}} \leq y) \leq e^{-ne^{-\lambda(y+1)}}$$

where n is the length of the alignment and λ is the score-normalizing constant defined previously.

This framework demonstrates why the expected score of aligning two amino acids must be negative: if this were not the case, the random walk could contain arbitrarily long upward increases from ladder points to excursions and the testing procedure would break down. This would essentially reduce BLAST to carrying out simple global alignment!

The expected length of a *max segment pair* (MSP) is

$$\mathbb{E}(l) = \frac{\log(Kmn)}{H}$$

where H is the relative entropy of the scoring matrix. As a heuristic, alignments are generally considered to be statistically significant if $l > \mathbb{E}(l)$. This length is shorter for PAM50 than for PAM250.

Normalized Bit Scores

How long must an alignment be in order to be considered non-random? In general, it needs to be long enough to achieve significance compared to the "noise" of shorter random alignments; however, it also depends on the specific scoring matrix employed. In order to compare the significance of alignments obtained from different databases using different scoring matrices, BLAST make use of *normalized bit scores* (developed by Karlin and Altschul), defined as follows:

$$S' = \lambda Y_{\text{MAX}} - \log(nK)$$

where $K = \frac{c}{A}e^{-\lambda}$ and A is the average distance between ladder points. Then the probability of randomly observing a given normalized score can be bounded as follows:

$$\begin{aligned} e^{-e^{\lambda}e^{-s}} &\leq P(S' \leq s) \leq e^{-e^{-s}} \\ \implies P(S' &\geq s) \approx 1 - e^{-e^{-s}} \end{aligned}$$

When s is large, the formula above can be approximated further:

$$P(S' \geq s) \approx e^{-s}$$

The *bit score* is closely related to S':

$$S_b = \frac{\lambda Y_{\text{MAX}} - \log K}{\log 2}$$

Using this bit score, the Karlin-Altschul equation can be reformulated as follows:

$$E = mn \cdot 2^{-S_b}$$

Importantly, this formula for E is independent of K, λ and S itself, which enables direct comparison of alignment scores obtained using different scoring matrices.