The Fasta and Blast programs

Frédérique Galisson

July 18, 2000

The motivation that has led to the development of the blast and fasta programs was to be able to compare one sequence against all the sequences of a database in an reasonable time. The problem was already important in the 80's, when databases of sequences were much smaller than they are now, but computers were slower than now as well. The Dynamic Programming methods for global or local alignments allow one to obtain the optimal alignment under a given scoring system, in a time that is proportional to the product of the lengths of the two sequences being compared. Thus, when they are applied to a whole database, the computation time grows linearly with the size of the database. With current sequence databases, calculating a full Dynamic Programming alignment for each sequence of the database is too slow (except when implemented on specialized parallel hardware).

The programs from the Blast and Fasta families are *heuristics* that reduce computation time by sacrificing some sensitivity: they reduce the size of the problem by *selecting* the sequences of the database that are thought to share significant similarity with the query sequence, and by locating the similarity regions inside the sequences. These selective steps allow to confine the time expensive sequence alignments methods only to a subset of the database sequences, and to restrict the search for the best alignment to only sub-regions of the sequences. In order to be fast, they estimate the similarity between the sequences in an approximate manner, and thus introduce a risk of missing and eliminating sequences whose similarity with the query is more tricky to detect, or to end up with alignments that are sub-optimal. One defines the *sensitivity* of a method as its ability to detect all the significant similarities, and thus to generate as few false negative results as possible. In an analogous way, one defines the *selectivity* of a method as its ability to select only sequences considered as significantly similar, and thus to generate as few false positives as possible. Sensitivity and selectivity both refer to the notion of the significance of a result. The biological significance of a similarity between two sequences is an arbitrary notion that depends on the

biological context in which the search is being performed, and thus it is very difficult and probably impossible to estimate it directly from the score given to the comparison. The programs will give us a measure of the *statistical significance* of the results, with respect to a random model: a result will be considered statistically significant if the probability of getting it just by chance is very low. The scoring systems and algorithms that are being used are developed with the aim that the searched similarity is a biologically meaningful similarity, and thus that results that are the most statistically significant are also those that are the most biologically relevant. In spite of everything, it is important to keep in mind that a comparison may get a score that is statistically significant without being of any biological relevance (for example in the case of "low-complexity" sequences, in which the similarity is due to compositional biases more than to conserved biological patterns in the sequences. Even if a compositional bias may be interesting for itself, it is not a "sequence" similarity per se); conversely, some poorly statistically significant results may correspond to interesting biological features, the main difficulty being, of course, to be able to detect them, and being able to interpret results of low statistical significance.

1 The Fasta algorithm

The algorithm of Fasta ([1]) is made of four steps, the first two being identical to those of the Fastp program, developed by the same authors in 1985 ([2]), and grounded on the diagonals method developed in 1983 ([3]). These steps are illustrated on figure 1.

1.1 Identification of similarity regions

The first step consists of localizing the ten best regions of similarity between the query sequence and each sequence of the database. The process described below is applied to each of the database sequences, when compared with the query sequence: the authors call "$k-tuple$" a word, or sequence fragment, of length k; all the $k-tuples$ of the query sequence (obtained by successive slidings of a window of length k along the sequence) are compared with all those of the database sequence and the identities between $k-tuples$ are detected and stored. The method may be represented in a two-dimensional matrix, with each identity between two $k-tuples$ being represented as a dot. Each diagonal segment corresponds to an ungapped alignment of the regions of the two sequences, as a succession of $k-tuple$ matches and mismatches (the program does not actually build the matrix but uses a more efficient

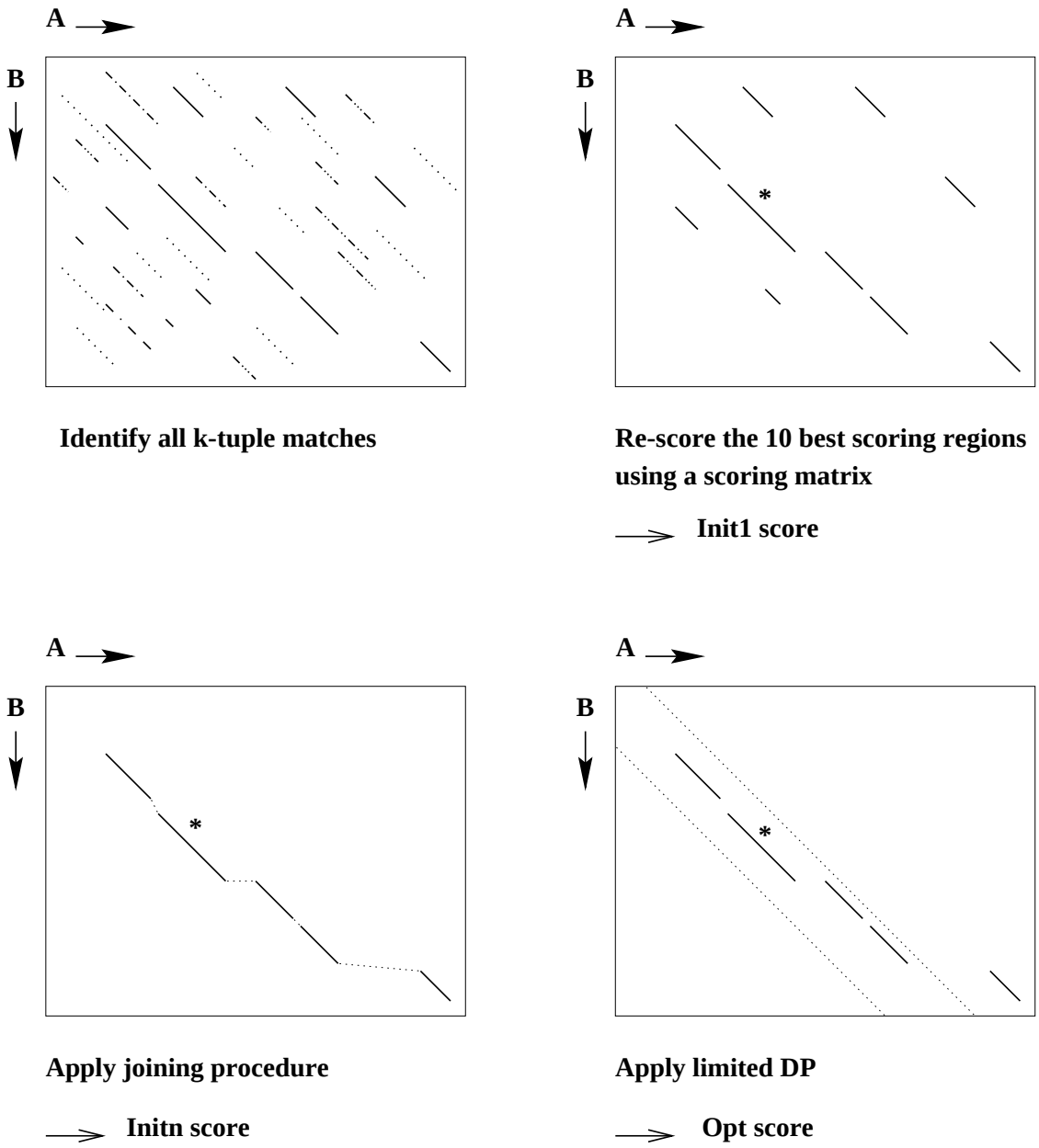


Figure 1: Fasta algorithm

computational structure known as "lookup table" in order to store all the pairs of identical $k-tuples$). Inside diagonals are regions of successive $k-tuple$ matches and mismatches, that are scored, with the score being simply a function of the respective numbers of matches and mismatches in the region. The ten regions containing the higher densities in $k-tuple$ matches are selected for the next step.

This step is very important with respect to the sensitivity of the method: only the regions that are selected at this step will be considered for the next one, and the selection of the database sequences that will be considered later will be done by examining only these regions of the sequences. The value chosen for the k parameter is important: the smaller k is, the greater the sensitivity, for example, if $k = 1$, one will take into account all the matches between individual amino acids, else, one will consider only stretches of k successive identities and then be more stringent. The greater k is, the faster the whole program will be, because less regions will be considered in the next steps. The choice of k will then reflects a trade-off between sensitivity and speed issues. Its also important to notice, that during this step the notion of similarity between amino acids is not taken into account.

1.2 Re-scoring the ten best regions

In the second step, the regions selected at the first step are re-scored, using a similarity matrix, and a sub-region whose similarity score is maximum (i.e. it cannot be increased by shortening or by extending the region) is determined. The score of the best of the ten regions is called the $init1$ score.

Given that the computation of these similarity scores is done only on the ten regions selected from the first step (using a density of $k-tuples$ identities criterium), it may happen that these ten regions are not the ten most *similar* between the sequences (e.g. in cases where there are many conservative substitutions and few identities between the sequences being compared).

1.3 Selection of the "more similar" sequences

The old Fastp program used this $init1$ score to rank the sequences of the database. Fasta attempts when possible to join some of these ten regions. For the joining process, only the regions having a score above a given threshold are considered. A combination of compatible regions is scored by summing the scores of the regions it contains and substracting a penalty for the region of "junction" (this penalty is analogous to a gap penalty since the region of

"junction" is the same of an unaligned region containing gaps). Fasta uses a Dynamic Programming optimization algorithm for computing the best combination, which is kept and whose score is called the *initn* score ([4]).

All the sequences of the database are processed through steps one to three and get an *initn* score. Then, sequences with an *initn* score greater than a threshold are kept for the next step, the others are eliminated (with current implementations only 10% of the sequences do not enter the fourth step, [5]).

1.4 Alignments of the selected sequences

Each of the selected sequences is aligned with the query sequence, using an algorithm that is a variation of the Smith-Waterman algorithm: a local alignment is calculated by Dynamic Programming, but restricting the region of the matrix that is being explored to only a band centered around the diagonal region that had got the best *init*1 score (this algorithm is named "banded SW" and is also illustrated on figure 4C). The score of the calculated alignment is called the *opt* score and that is the one that is used to rank the alignments. Because the Dynamic Programming process is applied only to a selected band (whose width is a parameter of the program) inside the whole comparison matrix, this may lead to an alignment that is only sub-optimal.

Conclusion

In 1988 the Fasta algorithm increased by a factor of 10 to 100 the speed of the similarity searches in sequence databases. This, nevertheless is at the price of sacrificing some sensitivity when compared to a pure Dynamic Programming algorithm that guarantees to find the best local alignment.

2 The Blast programs

The family of Blast programs was born in 1990 with the development of Blast1, which is very fast and dedicated to the search for regions of local similarities without gaps ([6]). In 1996-1997, two new versions of Blast appeared, that are slightly different from one another. Both differ from Blast1 in that they allow for the insertion of gaps. Both have been called Blast2 by their respective authors, and we shall distinguish them by using the initials of the institutions their authors belong to: NCBI-Blast2 and WU-Blast2 (NCBI is "National Center for Biotechnology Information" and WU is "Washington

University", both located in the USA). Both Blast2 algorithms are derived from Blast1.

2.1 Algorithm of Blast1

There are three distinct steps, that are represented on figure 2.

2.1.1 Preprocessing of the query

As with Fasta, the goal of the first step is to quickly locate ungapped similarity regions between the query sequence and sequences from the database. Similarly, all the words of length w (so, $w-tuples$, called "words" by Blast authors) of the query are compared with those of all the database sequences. Nevertheless, conversely to what is done with fasta, the similarities at the level of individual amino acids are taken into account at the first step.

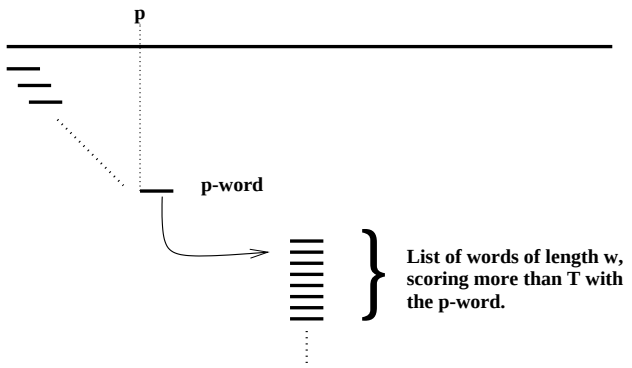
An obvious manner of doing that would be to compare each word of the query with each word of the database sequences, and calculate the similarity score for all the pairs of words (by summing the scores of the paired amino acids that are part of a paired word). Then, one has to decide upon a threshold such that all the word pairs with a score greater than or equal to the threshold are considered to be pairs of "similar" words, and one can obtain all the words in the database that are "similar" to each word of the query.

Blast uses a more efficient manner to do this, with the result being exactly the same: all the words of length w formed with the alphabet of the sequences are generated (for example, with amino acid sequences, if $w = 2$ there are 400 possible words, and 8000 if $w = 3$) and each word of the query is compared with each word of this exhaustive set and a threshold T for the similarity between words is set. Each position of the query sequence is associated with a list of words that score more than T when compared with the word of the query starting at this position. The similar words are also called "neighbors".

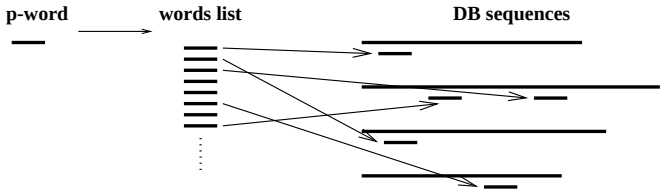
2.1.2 Generation of *hits*

Let D be a sequence of the database, and Q be the query sequence. After the first step, Q is now represented by lists of "neighbors", one list at each position of the query. Comparing Q with D yet consists in looking for identities between the "neighbors" at each position of Q and the words of D. So, each position of Q is compared with each word of D, and if one of the neighbor words at that position of Q is identical to the word of D, a *hit* is recorded (see figure 2B). A *hit* is made with one or several successive

A: For each position p of the query, find the list or words of length w scoring more than T when paired with the word starting at p:



B: For each words list, identify all exact matches with DB sequences:



C: For each word match («hit»), extend ungapped alignment in both directions. Stop when S decreases by more than X from the highest value reached by S.

Figure 2: Blast1 algorithm

(overlapping) pairs of similar words, and characterized by its position in each of the two sequences. All the possible *hits* between the query sequence and sequences from the database are calculated in that way.

2.1.3 Extension of the *hits*

Every *hit* that has been generated is now extended, without gaps, in order to determine whether this *hit* may be part of a larger segment of similarity. So, each hit is extended in both directions, and in order to make this extension step fast enough, an extension is stopped as soon as the score of the extended hit decreases more than X (the value chosen for X is a parameter of the program) when compared with the best score that has been reached during the extension process. One may notice that this manner of searching the best scoring paired segment containing a hit is an approximative one and does not guarantee that the resulting similarity segment pair is the best one. This is because of the "no more than X" drop off requirement.

Every extended segment pair that scores the same or better than S (set as a parameter of the program) is kept and called an HSP (*High scoring Segment Pair*). In a comparison, the best scoring HSP is called the MSP (*Maximal Segment Pair*).

Conclusion

The Blast1 program is thus a heuristic method looking for the best segments of local similarity without gaps between the query sequence and sequences from the database. Compared to Fasta, one advantage of Blast1 is that it takes into account the similarities between amino acids in the first selection step. On another hand, it does not allow for gaps in the similar segments it is seeking, and this makes it less sensitive than Fasta in some cases.

Historically, the development of Blast programs has paralleled advances made in our ability to evaluate the statistical significance of local alignments scores. Indeed, one of the strengths of Blast1 is that a theory describing the distribution of MSP scores has been developed at the same time as the Blast1 algorithm. Under this theory, it is possible to associate the score of any MSP with a $p - value$, which is the probability to get at least one score equal or greater than that of the MSP by *chance*; i.e. the $p - value$ is the probability that there exists one or more MSP, obtained from the comparison of two random sequences (having same length and composition as the sequences of interest), whose score is equal or greater than the one of the *MSP* obtained with the actual sequences (see section 3 for more details).

Blast1 ranks the results, according to their $p - values$. With earlier versions of Blast1, when there were several ungapped segments of similarity between the query and one sequence of the database, they were treated independently. Poisson statistics with Blast1.3, then Karlin-Altschul "Sum statistics" with Blast1.4, were introduced for the joint evaluation of multiple ungapped local alignements between two sequences. "Sum statistics" calculates a $p - value$ for the combination of several HSP, corresponding to the probability of getting several HSP whose sum of scores is equal or greater than the one of interest, in a comparison of random sequences ([7]). Because it makes sense in biology that two sequences may share several ungapped regions of similarity separated by unrelated sequence segments, the ability to take into account (through statistics) the joint occurences of several HSP between two sequences, increases the sensitivity of the method towards "biological similarity".

A few years after Fasta and Blast became to be widely used, it was shown that even though there was no theory describing the distribution of the scores of optimal local gapped alignments, empirically and under certain conditions, this distribution seemed to be the same as the one of optimal local alignments without gaps (an *extreme value* distribution, see section 3). William Pearson, Fasta's author, have shown empirically that the scores obtained with the Fasta algorithm were also following the same kind of distribution. Furthermore, the authors of Blast1 have developed another program, called Blast2, allowing to search for gapped local alignments. For some reasons, two Blast2 programs have been developed that are slightly different.

2.2 NCBI-Blast2

This NCBI-Blast2 program has been developed at NCBI (« National Center for Biotechnology Information »), and its algorithm has been published in 1997) ([8]). The first two steps, leading to the generation of primary *hits* are the same as with Blast1.

The first important difference concerns the selection of the hits that are going to be extended: the goal of the authors is to generate local gapped alignments from the computed *hits*; this requires an additional step (and time) for calculating the gapped alignments; on the other hand, *hits* that could not be joined through ungapped extensions may be part of the same gapped alignment, and it may not be necessary to extend all of them. In order to save time, they select fewer *hits* for further ungapped extension, using a requirement for a *hit* to be extended: they require that there be another *hit*, on the same diagonal, within a distance smaller than A (a pa-

rameter of the program, which value may be changed by the user). This process is illustrated on figure 3. Because this new requirement makes the program be less sensitive (not all the *hits* will satisfy this new condition), they suggest the use of a smaller value for the T parameter (threshold for the similarity between words, used at the first step, when generating lists of "neighbors"), in order to generate more *hits* at the second step. Of course, these two parameters, A and T do not influence the sensitivity at the same level.

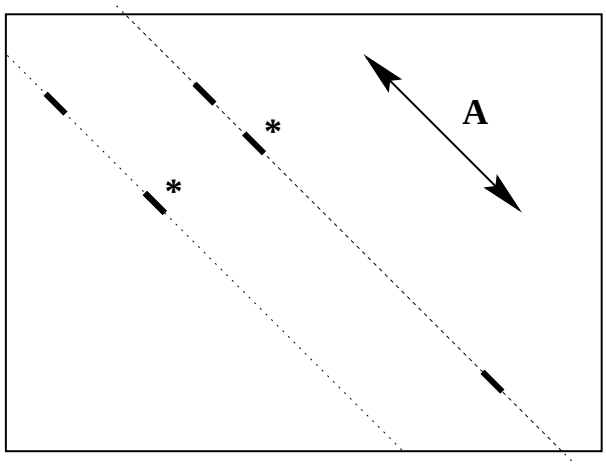
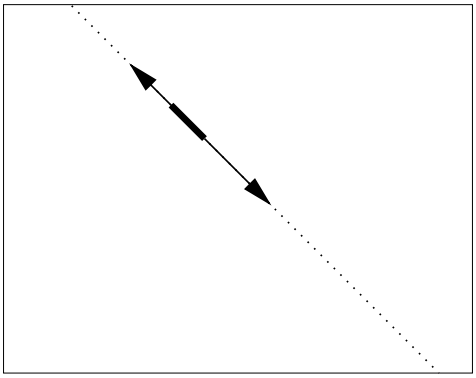


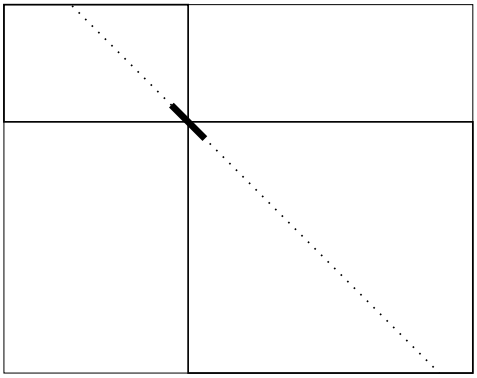
Figure 3: The "two-hits" requirement: those hits that are going to trigger an ungapped extension are labelled with a star

All the *hits* that fullfill these requirements are selected for an ungapped extension (like is done at the third step of Blast1). Among the generated HSP, those that have a score above a threshold will trigger a gapped extension: they are used as start points for performing dynamic programming local alignments. The algorithm used for computing these local gapped alignments is a modification of the Smith-Waterman algorithm: the Dynamic Programming matrix is explored in both directions (see figure 4D) starting from the middle point of the highest scoring $11 - tuple$ within the HSP, and the search for the optimal path is restricted to cells of the matrix such that the score of the alignment does not drop off more than X_g when compared to the maximal score reached since beginning the extension (the same idea as in the case of the ungapped extension).

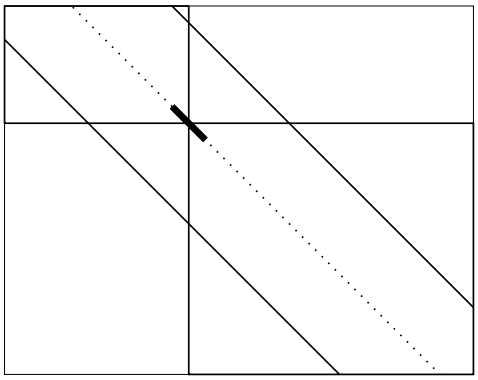
The alignments are ranked by increasing $E - values$. The $E - value$



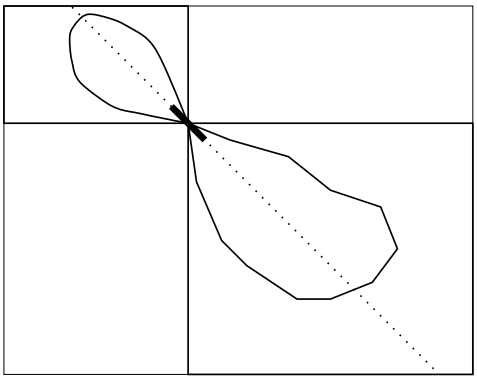
Ungapped extension



Gapped extension by full DP



Gapped extension by «banded DP»



Gapped extension by «score-limited DP»

Figure 4: Gapped and ungapped extensions

of an alignment having score S is the number of times one expects to find alignments having score equal to or greater than S in a comparison of random sequences having the same length and composition as the query and the database. It is related to the $p - value$ described with Blast1, but perhaps makes more intuitive sense than a probability value.

2.3 WU-Blast2

Another Blast2 has been developed by Warren Gish from Washington University. WU-blast2 was released in 1996, one year before NCBI-blast2; its algorithm has never been published and the description that follows takes its sources from the available documentation and from personal communications from the author.

The WU-blast2 program proposes several parameters allowing the emulation the different blast versions. Its default behaviour is such that the three first steps are the same as with Blast1, and in the fourth step, the HSP having a score greater than a threshold trigger a gapped extension. The algorithm used for the gapped extension is an amalgam of a "banded" (what Fasta does) and a "score-limited" (like NCBI-Blast2) Dynamic Programming algorithm (see figures 4C and 4D). If one selects the "nogap" option, this last step is not performed, and WU-blast2 becomes identical to Blast1. There is also an option called "hitdist", that when activated allows to perform the same "two-hits" requirement as with NCBI-blast2 at the third step. Thus, combinations of these two options allow to emulate four different blast algorithms.

Another feature of WU-Blast2 is its ability to consider the case of several local similarity regions between two sequences. If a given database sequence yields multiple ungapped HSP, the gapped extension routine will be applied to those of them whose scores exceed the chosen threshold, and may give rise to reports of multiple gapped alignments. Instances of multiple local alignments are then evaluated jointly using Sum statistics ([7]), like is done by the Blast1 program when there are several HSP scoring above the S threshold. The Sum statistics allow to take into account the simultaneous presence of several local regions of similarity between two sequences. Fasta reports only one local alignment per sequence of the database, and when there are several of them, NCBI-blast2 treats them independently. In the situation where there are two local similarity regions, separated by a low similarity region, with NCBI-Blast2 either they will lead to different independent reports (with Fasta only the best one is reported), or they will be included in the same alignment (when the gapped extension triggered from one of

them reaches the other one) but it may be at the price of forcing a meaningless alignment between them. The way WU-Blast2 deals with mutiple local alignments enables to take into account their simultaneous occurences, without attempting to align regions of low similarity between them.

Conclusions about the algorithms

If we were using the Smith-Waterman algorithm for comparing one query sequence against a database, we would end up with a collection of local alignments, each of them being the optimal one between the two sequences being compared. With Fasta, or one of the Blast algorithms, since the optimization procedures they use are *heuristic* ones, the optimality of the alignments is not guaranteed. So, the score that is given for the comparison of two sequences using these algorithms may not be the best possible one, and as a consequence, the order in which the alignments are ranked, directly or not based on their respective scores, may also not be the "good" one.

Nevertheless, when trying to estimate the significance of the results, we will make the assumption that these methods succeed in finding the optimal alignments and scores between the query and the database's sequences. So, we will consider that the result of a search against a sequence's database gives us the best local similarities relative to the query. Of course, the notions of "best", "optimal", and even "local" depend on the scoring system that is used to quantify the similarities between biological residues, and to penalize the introduction of gaps. If this scoring system models some notions of biological similarity, we may expect the alignments having the best scores to be the more biologically meaningful. Any biological inference (like homology) that is done directly and solely from the interpretation of an alignment score relies on this hypothesis. Given that there are many ways of considering biologically similarity, it is obvious that there is not one ideal choice of scoring parameters that will model all the biological viewpoints, if there is any that will correctly model one.

For these reasons, we will not try to evaluate the biological significance or relevance of an alignment only from its score. This has to be done with respect to the biological context of the search, and influenced by external knowledge. But, what we may want to know is whether or not a given result (an alignment and its score) is statistically significant. It means being able to say when two sequence's fragments have to be considered "similar". That is what will be developped in the next section.

3 Statistics

The question we want to answer here is: given an alignment having score S, how strong is the similarity it represents? If "strong" means that we "could not" have got it by chance in the context of the search, then the question becomes: what is the probability of chance occurence of an alignment having a score S or greater when looking for the best alignments in a database of sequences? Answering this question requires to know the statistical distribution of the scores in a *random model*. Whereas very little is known about the statistical distribution of global optimal alignments scores, the statistics of local optimal alignments scores is now well understood. Particularly, in the case of local ungapped optimal alignments, there exists a theory describing their random distribution.

3.1 statistics of local ungapped alignments

In 1990, Samuel Karlin and Stephen Altschul developped a theory that allows one to calculate analytically the random distribution of optimal local ungapped alignment scores ([9], [10], [11], [12]). In order to calculate this distribution, one needs to define a random model of sequences; the one used here consists of sequences of residues taken independently with background probabilities p_i, p_j ($1 \leq i, j \leq 20$ for proteins), under identical distributions. These background probabilities are supposed to be the relative frequencies of the residues in the query and database sequences. The random variable for which the theory holds is the score S of the optimal local segment of similarity without gaps calculated from the comparison of two sequences. It is also called the *Maximal Segment Pair* (MSP) between the two sequences being compared. The aim of the Blast1 algorithm is to find such MSP. One may notice that applying this theory in the context of a database search is an approximation where one considers each HSP as an MSP. The score of an MSP depends on the scoring system that is used, consisting of a set of similarity scores $S_{i,j}$ which, for the theory to hold, has to fulfill two requirements: there must be at least one $S_{i,j}$ strictly positive, and the expected score for a random pair of residues, $\sum_{i,j} p_i p_j S_{i,j}$ has to be negative. Another requirement is that the lengths (m and n) of the two sequences involved in the comparison are large.

Under this random model, the random variable S may be shown to follow an *extreme-value* distribution ([6], [10], [12]). This allows one to calculate the $p - value$ of an MSP having score S. It is the probability, under the random model described above, to get an MSP with a score greater than or

equal to S. If the random model fits the reality of an actual database search, the $p-value$ will be the chance probability of an MSP of score at least S, in this context. This $p-value$ may be expressed as a function of S:

$$p(score \geq S) = 1 - exp(-Kmne^{-\lambda S})$$

where m and n and the lengths of the sequences being compared (one considers the database as one big sequence), K is a constant that depends on the p_i and $S_{i,j}$, and λ is the unique positive-valued solution for x in

$$\sum_{i,j} p_i p_j e^{s_{i,j}x} = 1.$$

So, for a given search, m and n are known, and K and λ may be calculated knowing the parameters of the random model. The $E-value$, the expected number of alignments having score at least S may also be calculated as $Kmne^{-\lambda S}$.

So, in the case of the optimal local alignments without gaps, the Karlin-Altschul theory gives analytical formulas allowing to estimate the statistical significance of the scores.

3.2 Local alignments with gaps

The Smith-Waterman, Fasta and Blast2 algorithms calculate local alignments with gaps. In this case, there is no theory describing the expected chance distribution of the scores ([13], [14]). Nevertheless, several empirical simulations ([10] and [15]) indicate that under some requirements about the scoring system, these scores seem to follow also an extreme-value distribution. The problem for calculating the $p-value$ or $E-value$ of a gapped alignment score is that in the absence of theory, we lack analytical formulas for calculating the parameters of the distribution (dependent on those of the random model). In practice these parameters are estimated from empirical simulations of the random distribution. Simulating a random distribution may be done by using either artificial random sequences (generated under the random model), or by using real but unrelated sequences:

- With both Blast2 programs, the statistical parameters are obtained from pre-computed tables created using Monte Carlo simulations, as described in [10]. Because these values depend on the scoring system (scoring matrix and associated gap penalties), only sets of scoring parameters for which the pre-computation has been done are available with Blast2 programs. One problem that may arise with this approach

is if the query or database sequences have compositions that differ significantly from those of the random model sequences. Then, the statistics will loose accuracy.

- Fasta computes specific statistical parameters for each search, taking into account its specific parameters. The random distribution is simulated by real sequences from the database, that are considered to be unrelated with the query sequence ([16]). The advantage of this approach is that the computation of the statistical parameters is tailored to the specific query, database and scoring system chosen by the user. The difficulty is to distinguish between sequences that are related to the query from those that are not, or said differently, to distinguish between random and nonrandom similarity.

Discussions about the interpretation of statistical estimates in the context of a database search and their relationships with biological significance may be found in [12] and [16].

Acknowledgments

I am grateful to Stephen Altschul, Warren Gish and William Pearson for their critical reading of the text and their suggestions which improved it. I also wish to thank Emmanuel Chang for his careful reading and editing of the manuscript.

References

[1] W.R. Pearson and D.J. Lipman. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, 85:2444–2448, 1988.

[2] D.J. Lipman and W.W.R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227:1435–1441, 1985.

[3] W.J. Wilbur and D.J. Lipman. Rapid similarity searches of nucleic acid and protein data banks. *Proc. Natl. Acad. Sci*, 80:726–730, 1983.

[4] William R. Pearson. Rapid and sensitive sequence comparison with fastp and fasta. *Methods in Enzymology*, 183:63–98, 1990.

[5] William Pearson, personal communication.

[6] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215:403–410, 1990.

[7] Samuel Karlin and Stephen Altschul. Applications and statistics for multiple high-scoring segments in molecular sequences. *Proc. Natl. Acad. Sci. USA*, 90:5873–5877, 1993.

[8] S.F. Altschul, T.L. Madden, A.A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D.J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.

[9] S. Karlin and S.F. Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, 87:2264–2268, 1990.

[10] Stephen Altschul and Warren Gish. Local alignment statistics. *Methods in Enzymology*, 266:461–480, 1996.

[11] Stephen F. Altschul. Sequence comparison and alignment. In M.J. Bishop and C.J. Rawlings, editors, *DNA and Protein sequence analysis*, The practical Approach Series, D. Rickwood and B.D. Hames series editors, chapter 7. IRL Press, 1997.

[12] Stephen Altschul, Mark S. Boguski, Warren Gish, and John C. Wootton. Issues in searching molecular sequence databases. *Nature Genetics*, 6:119–129, 1994.

[13] Michael S. Waterman and Martin Vingron. Sequence comparison significance and poisson approximation. *Statistical Science*, 9(3):367–381, 1994.

[14] Michael S. Waterman and Martin Vingron. rapid and accurate statistical estimates of statistical significance for sequence data base searches. *Proc. Natl. Acad. Sci. USA*, 91:4625–4628, 1994.

[15] William R. Pearson. Empirical statistical estimates for sequence similarity scores. *J. Mol. Biol.*, 276:71–84, 1998.

[16] William R. Pearson. Flexible sequence similarity searching with the fasta3 program package. *Methods in Molecular Biology*, 1999.