Smith–Waterman Algorithm

Advanced article

Richard Mott, *University of Oxford, Oxford, UK*

The Smith–Waterman algorithm is a computer algorithm that finds regions of local similarity between DNA or protein sequences.

Article contents

doi: 10.1038/npg.els.0005263

Introduction

A central task in bioinformatics is the alignment of pairs of deoxyribonucleic acid (DNA) or protein sequences in order to determine whether they are descended from a common ancestor. Because evolution may have caused the substitution, insertion and deletion of nucleotides or amino acids, the biologically correct sequence alignment – that is, that which reconstructs the true evolutionary history – will match conserved regions, while placing mismatches and gaps wherever mutations have occurred. In general, the alignment will not run from the start to the end of both sequences, but instead will comprise a local similarity. The Smith–Waterman algorithm is used to find these optimal local alignments automatically. The algorithm is important not only because it solves a core task elegantly and efficiently, but also because many other bioinformatics problems, such as predicting genes in genomic DNA, can be solved using a generalization of it, the hidden Markov model (HMM), and because of the impetus it has given to the study of sequence alignment statistics. (***See*** Sequence Alignment.)

Background and History

The Smith–Waterman algorithm belongs to a family of dynamic-programming methods that have been repeatedly rediscovered in molecular biology, engineering, computer science, linguistics and statistics. Needleman and Wunsch (1970) introduced dynamic programming to molecular biology in 1970, when they showed how to align two protein sequences from end to end (i.e. perform a global alignment). It was soon realized that a local alignment algorithm was required, because the constraint that the sequences' ends match often produced alignments that were biologically incorrect. Throughout the 1970s, various solutions were proposed, of which Sellers' metric method (Sellers, 1974) is probably the best known. However, all these algorithms are somewhat impracticable. (***See*** Dynamic Programming; Global Alignment.)

In 1981, Temple Smith and Michael Waterman (Smith and Waterman, 1981) published an algorithm that finds optimal local sequence alignments. With a refinement of it proposed by Gotoh (1982), this algorithm has become a cornerstone of biological sequence analysis.

Algorithm

The inputs to the algorithm are the DNA or protein sequences to be aligned and a scoring scheme, which determines how mismatches and gaps are to be scored relative to matches. The output is an optimal local alignment that maximizes the alignment score. The choice of scoring scheme strongly affects the optimal alignment, and much research has been devoted to finding scoring schemes that can discriminate between genuine and random similarities. This subject is closely related to the issue of alignment statistical significance. (***See*** Alignment: Statistical Significance.)

The scoring scheme comprises:

1. A gap penalty, in which the cost of inserting a gap of k consecutive symbols is some function $g(k)$. The affine gap penalty $g(k) = A + Bk$ is most common, where there is a high cost $A + B$ to start a gap, and a lower marginal cost B to extend it: gaps tend to occur as single evolutionary events rather than to accumulate from many adjacent mutations. Gap penalties that increase more slowly with (or are independent of) gap length are suitable in situations where very long gaps may be expected, for example, the alignment of spliced DNA to genomic DNA, where the presence of very long introns may be suspected.
2. A substitution matrix $\mathbf{M}(a, b)$ for aligning symbols a, b that is positive when the symbols are likely to be related, and negative otherwise. It is a requirement that the average symbol match score should be negative. For DNA sequences, $\mathbf{M}(a, b)$ is usually positive if $a = b$, and negative otherwise. Protein substitution matrices are more complicated, being derived from substitution frequencies observed in sequence alignments that have been validated from protein structure data. In general, amino acids are

likely to have a positive substitution score if they have similar chemical properties (e.g. are hydrophobic), because such substitutions tend to preserve protein structure and function, and are therefore favored during evolution. Two commonly used families of matrices are the BLOSUM and PAM series. (*See* Substitution Matrices.)

Every possible alignment of part of one sequence with part of the other can be scored using the substitution matrix and gap penalty. For example, a plausible local alignment of the DNA sequences `TTACCGGCCAACTAA`, `ACCGTGTCACTAAC` is shown in **Figure 1**, which contains nine matches, one mismatch, and two gaps each of length 1. If matches score +2, mismatches −3 and gaps $3 + k$, then the score of this alignment is $9 * 2 - 1 * 3 - 2 * 4 = 7$. Note that this is a local alignment, because the prefix '`tt`' has been discarded from the first sequence without penalty, along the suffixes '`a`', '`c`'. The corresponding global alignment, with score −1, is shown in **Figure 2**.

The objective of the Smith–Waterman algorithm is to find the local alignment(s) with the highest score. In this simple example, it is clear by inspection that we have found a best local alignment for this scoring scheme, but for sequences with thousands of nucleotides or hundreds of amino acids it may no longer be obvious what the optimum is. Furthermore, as the number of local alignments grows exponentially with the sequence lengths, it is clearly impossible to find the optimum by enumerating and scoring all alignments.

Instead, a dynamic-programming algorithm is used to arrange the calculation so that the number of steps required is proportional to the product of the sequence lengths. This is a huge reduction in effort and means that it is possible to search protein databanks in a few minutes on a desktop computer.

Denote the two sequences to be aligned by the strings of symbols $\mathbf{a} = a_1a_2a_3 \ldots a_n$, $\mathbf{b} = b_1b_2b_3 \ldots b_m$. These symbols will be either the nucleotides `A`, `C`, `G`, `T` or from the 20-letter amino acid alphabet. Imagine an array of n rows and m columns, with sequence **a** arranged vertically down the left-hand side and sequence **b** horizontally along the top. There are two key ideas behind dynamic programming. First, every alignment corresponds to a path in this matrix. Where the alignment contains a match or mismatch between a_i and b_j, the path takes a diagonal route through the cell i,j. A gap inserted in **a** between a_i and a_{i+1} opposite $b_j \ldots b_k$ corresponds to a vertical path from cell i,j to i,k. Similarly, a gap inserted in **b** is represented by a horizontal path. Second, the optimal alignment terminating in the cell i,j must be built from one of the paths passing through the three preceding cells, diagonally $i-1, j-1$, horizontally $i-1, j$ or vertically $i, j-1$. Let $S_{i,j}$ be the score of the optimal alignment terminating at i,j. Let $H_{i,j}$ be the score of the top-scoring alignment to i,j entering the cell horizontally, and $V_{i,j}$ be the best alignment entering vertically. These quantities may be calculated as follows:

$$S_{i,j} = \max\ [0,\ S_{i-1,j-1} + M(a_i, b_j),\ H_{i,j}, V_{i,j}]$$
$$H_{i,j} = \max_{k<i}[0,\ S_{i-k,j} + g(k)]$$
$$V_{i,j} = \max_{k<j}[0,\ S_{i,j-k} + g(k)]$$

These are called dynamic-programming recurrence equations. The matrices **S**, **H**, **V** are computed cell by cell, starting from cell 1,1 and ending at m, n, usually row by row.

The score of optimal local alignment can be found by keeping a record of that cell with the overall maximum score. There may be more than one such

```
ttACCG-GCCAACTAa
  |||| |*|| |||
  ACCGTGTCA-CTAc
```

Figure 1 Local alignment of the DNA sequences `TTACCGGCCAACTAA`, `ACCGTGTCACTAAC`. Aligned portions are shown in upper case and unaligned portions of the sequences are shown in lower case.

```
TTACCG-GCCAACTAA
  |||| |*|| |||*
--ACCGTGTCA-CTAC
```

Figure 2 Global alignment for the example shown in **Figure 1**.

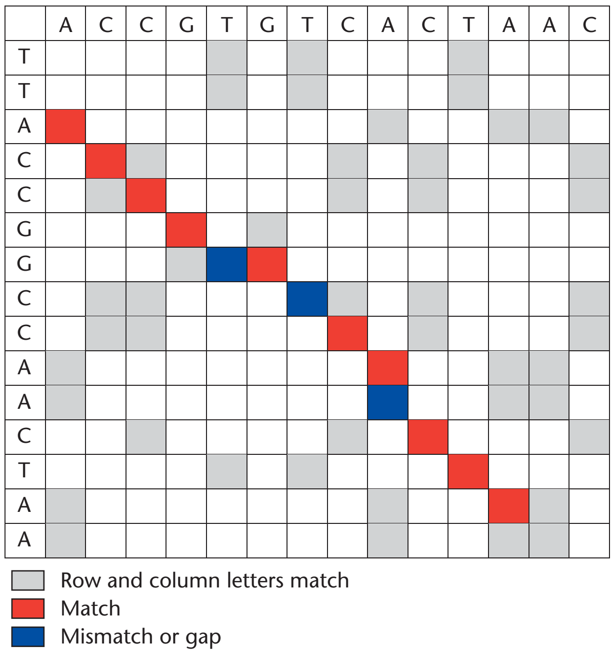
A C C G T G T C A C T A A C
T T A C C G G C C A A C T A A
▨ Row and column letters match
🟥 Match
🟦 Mismatch or gap

Figure 3 Dot matrix for the optimal local alignment between sequences, showing cells in which the corresponding row and column letters match; cells lying on the optimal alignment that match; and where the path through the cell is a mismatch or gap.

cell, implying that there are multiple optimal alignments. The alignment itself can be recovered from its path through the matrix, by keeping a record of each cell's predecessor in the recursion, and then backtracking through the matrix until an edge of the matrix or cell with zero score is reached (see **Figure 3**).

Smith and Waterman's insight was to put the zeros in the above equations; these have the effect of trimming alignments with a negative score back to their positive cores, and ensure that the algorithm does find optimal local alignments. Smith and Waterman define an alignment to be locally optimal if it cannot be extended or shrunk without reducing its score. This is equivalent to saying that a path is locally optimal if no subpath of it has a higher score and if it is not a subpath of a higher-scoring path. To see why the algorithm works, suppose there is an optimal local alignment from cells i, j to k, l. This means that there cannot be a path ending at i, j with a positive score, for if there were, then the alignment could be extended backwards. Similarly, if the alignment could be extended to the right, then k, l would no longer be the global maximum in the matrix, for another cell would exceed it. Finally, if the alignment could be shrunk, then either a prefix or a suffix of the alignment must have a negative score. Now the effect of the zeros in the recursion is to automatically trim away paths with negative score, so, combined with the global maximum rule, we see that the algorithm will find the optimal local alignment.

Gotoh's contribution was to note that if the gap penalty is affine, then the recurrences can be recast in a more efficient form, which reduces the time taken by the algorithm from a quantity proportional to $mn(m+n)$ to mn, i.e. a 100- to 1000-fold saving in a typical sequence comparison:

$$S_{i,j} = \max[0, S_{i-1,j-1} + M(a_i, b_j), H_{i,j}, V_{i,j}]$$

$$H_{i,j} = \max[0, H_{i-1,j} + b, S_{i-1,j} + a + b]$$

$$V_{i,j} = \max[0, V_{i,j-1} + b, S_{i,j-1} + a + b]$$

This is a considerable saving in computer time, and renders the algorithm practical for searching protein databanks. It is also easy to implement on specialist computer hardware which can perform many operations in parallel. **Table 1** lists a number of implementations of the Smith–Waterman algorithm.

Extensions and Applications

There are now many variations on the basic Smith–Waterman algorithm. These include the use of specialized gap penalties for aligning spliced DNA (for instance, a complementary DNA, cDNA) onto genomic DNA, to identify gene structure (Mott, 1997; Florea *et al.*, 1998). Other variants are for aligning a protein sequence to a DNA sequence, allowing for the possibility that the translated reading frame jumps (e.g. due to a sequencing error or the presence of an intron). Algorithms for finding local sequence alignments with nonaffine gap penalties are described by Mott (1999). Recently Arslan *et al.* (2001) have shown how to identify local alignments with maximal per cent homology, rather than maximum score. Several systems such as Dynamite have been developed to help create a dynamic-programming code for novel problems. (***See*** Expressed-sequence Tag (EST).)

The most widely used development has been profile alignment. A profile, also known as a position-specific scoring matrix (PSSM), or a profile HMM (Durbin *et al.*, 1999), is a way to represent the variation in sequence conservation typically observed across a multiple alignment of several related sequences. Often certain positions in the multiple alignment are tightly conserved, indicating active sites or other key functional or structural domains, while other parts are highly variable, perhaps corresponding to loops in the protein structure. In the profile, the substitution matrix differs at each position to reflect the level of conservation. Profiles are accommodated very easily in the Smith–Waterman framework. (***See*** BLAST Algorithm; Hidden Markov Models; Profile Searching.)

Profiles are generally more sensitive at detecting distantly related sequences than are ordinary sequence comparisons, and this insight has led to the creation of databases of protein multiple alignments, corresponding to conserved protein domains, and their associated profiles (Pfam; SMART). Indeed, over 60% of human genes contain at least one known protein domain.

Strategies

It is generally inefficient to find gene homologies from direct DNA to DNA interspecies sequence comparisons. The redundancy in the genetic code means that even quite closely related genes may exhibit only 60% identity at the DNA sequence level, whereas protein comparisons, which take advantage of substitution matrices that model likely amino acid replacements, are much more sensitive. When a novel gene is discovered, either by *de novo* gene prediction algorithms applied to genomic DNA, or from sequencing a cDNA library, a better characterization is usually obtained by performing the following order of searches:

1. Compare the translated sequence to a library of annotated protein domains such as Pfam or SMART, using profiles or HMMs, or a frameshift-tolerant DNA–protein comparator such as GENEWISE.

Table 1 Implementations of the Smith–Waterman and related algorithms

Program	Web address	Comments
Standard implementations, mainly for protein–protein comparisons		
SSEARCH	ftp://ftp.virginia.edu/pub/fasta	FTP server; part of FASTA. Assesses statistical significance by function-fitting
MPSRCH	http://www.anedabio.com/products/mpsrch/index.html	Fast standard implementation
ARIADNE	http://www.well.ox.ac.uk/rmott/ARIADNE/ariadne.html	Assesses protein statistical significance by a formula; ideal for self-comparisons
Fast implementations using heuristics to accelerate databank searches		
BLAST	http://www.ncbi.nlm.nih.gov/BLAST/	Standard database search package. Family of related programs for DNA and protein searches. Statistical significance assessed by formulas
FASTA	http://www.people.virginia.edu/~wrp/pearson.html	Comprehensive, popular database search software; includes programs for DNA and protein comparisons
CrossMatch	http://www.genome.washington.edu/UWGC/analysistools/Swat.cfm	Optimized for all-versus-all comparisons
Profile-based comparisons		
HMMR	http://hmmer.wustl.edu/	Use hidden Markov models to identify protein similarities with multiple alignments
SAM	http://www.cse.ucsc.edu/research/compbio/sam.html	
ARIADNE	http://www.well.ox.ac.uk/ariadne	
Specialized applications – gene annotation		
EMBOSS: est2genome	http://www.uk.embnet.org/Software/EMBOSS/Apps/est2genome.html	All align a spliced DNA to genomic DNA, skipping long introns if necessary
SIM4	http://globin.cse.psu.edu/	
GenSeqer	http://bioinformatics.iastate.edu/cgi-bin/gs.cgi	
Dynamite	http://www.sanger.ac.uk/Software/Dynamite/	Creates customized dynamic-programming algorithms for specialized applications
Wise2	http://www.sanger.ac.uk/Software/Wise2/	Dynamic-programming algorithms, including DNA–protein comparisons

2. Perform a PSI-BLAST search of its predicted protein product against a protein sequence database such as NR or SPTREMBL.
3. Perform a Smith–Waterman search using FASTA or BLASTP.
4. Perform a DNA to DNA BLASTN search of genomic DNA, filtered to remove low-complexity sequence and known repeats.

However, DNA–DNA comparison is very useful for discovering conserved short sequences that may have a gene regulation function, e.g. a promotor or transcription-factor binding site. In this context, large regions (tens or hundreds of kilobase pairs long) of genomic DNA from, for instance, human and mouse, and containing genes known to be syntenic are compared. This is an active area of research, and a number of hybrid methods that combine the Smith–Waterman algorithm with fast, dictionary-based matching algorithms have been proposed (Batzoglou *et al.*, 2000). (***See*** Comparative Genomics; FASTA Algorithm; Homologous, Orthologous and Paralogous Genes; Promoters.)

During the first decade of the twenty-first century, the number of fully sequenced and annotated genomes will increase dramatically. Almost every gene will become fully characterized in terms of its homologs and paralogs and domain structure. We may find that the central role sequence comparison has played in discovering these relationships is subsumed into databases of precomputed links, and the emphasis is shifted to a synthesis of complementary data such as domain occurrence, metabolic pathways and expression data. (***See*** Genetic Databases: Mining.)

See also

Dynamic Programming
Global Alignment
Hidden Markov Models
Sequence Alignment
Substitution Matrices

References

Arslan AN, Egecioglu O and Pevzner PA (2001) A new approach to sequence comparison: normalized sequence alignment. *Bioinformatics* **17**: 327–337.

Batzoglou S, Pachter L, Mesirov JP, Berger B and Lander ES (2000) Human and mouse gene structure: comparative analysis and application to exon prediction. *Genome Research* **10**: 950–958.

Durbin R, Krogh A, Michison G and Eddy S (1999) *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids.* Cambridge, UK: Cambridge University Press.

Florea L, Hartzell G, Zhang Z, Rubin GM and Miller W (1998) A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Research* **8**: 967–974.

Gotoh O (1982) An improved algorithm for matching biological sequences. *Journal of Molecular Biology* **162**: 705–708.

Mott R (1997) EST_GENOME: a program to align spliced DNA sequences to unspliced genomic DNA. *Computer Applications in the Biosciences: CABIOS* **13**: 477–478.

Mott R (1999) Local sequence alignments with monotonic gap penalties. *Bioinformatics* **15**: 455–462.

Needleman SB and Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology* **48**: 444–453.

Sellers P (1974) An algorithm for the distance between two finite sequences. *Combinatorial Theory* **16**: 253–258.

Smith TF and Waterman MSW (1981) Identification of common molecular subsequences. *Journal of Molecular Biology* **147**: 195–197.

Further Reading

Waterman MS (1995) *Introduction to Computational Biology Maps, Sequences and Genomes.* Boca Raton, FL: CRC Press.

Web Links

Pfam (protein families database of alignments and HMMs). Updated May 2002
http://www.sanger.ac.uk/Pfam

SMART (simple modular architecture research tool). Updated May 2002
http://smart.embl-heidelberg.de