# Bioinformatics Explained

# Bioinformatics explained: Smith-Waterman

May 1, 2007

CLC bio
Gustav Wieds Vej 10 · 8000 Aarhus C · Denmark
Telephone: +45 70 22 55 09 · Fax: +45 70 22 55 19
www.clcbio.com · info@clcbio.com

CLC bio
a result of science

# Bioinformatics explained: Smith-Waterman

The Smith-Waterman algorithm is a dynamic programming method for determining similarity between nucleotide or protein sequences. The algorithm was first proposed in 1981 by Smith and Waterman and is identifying homologous regions between sequences by searching for optimal local alignments. To find the optimal local alignment, a scoring system including a set of specified gap penalties is used [Smith and Waterman, 1981].

Homology identified by sequence database searches often implies shared functionality between sequences and further research and development might depend on the accuracy of the search results. The Smith-Waterman algorithm is build on the idea of comparing segments of all possible lengths between two sequences to identify the best local alignment. This means that the Smith-Waterman search is very sensitive and ensures an optimal alignment of the sequences. Unfortunately, this also has the effect that the method is both time and CPU intensive.

## The history of the Smith-Waterman algorithm

Needleman and Wunsch (1970) were the first to introduce a heuristic alignment algorithm for calculating homology between sequences. Later, a number of variations have been suggested, among others Sellers (1974) getting closer to fulfill the requests of biology by measuring the metric distance between sequences [Smith and Waterman, 1981]. Further development of this led to the Smith-Waterman algorithm based on calculation of local alignments instead of global alignments of the sequences and allowing a consideration of deletions and insertions of arbitrary length.

The Smith-Waterman algorithm is the most accurate algorithm when it comes to search databases for sequence homology but it is also the most time consuming, thus there has been a lot of development and suggestions for optimizations and less time-consuming models. One example is the well-known Basic Local Alignment Search Tool, BLAST [Shpaer et al., 1996].

## Database similarity searching

Database similarity searches are mathematical approaches to sequence comparisons and as similarity searches are among the best ways of gaining information about putative function of a given sequence, sequence comparisons are fundamental in bioinformatics.

The main reasons for performing database similarity searches between a nucleotide or protein query sequence of interest and sequences in a database are listed below:

- Identify conserved domains in nucleotide or protein sequences of interest to predict functions of new and uncharacterized sequences
- Compare known sequences and identify homology between these sequences
- Search sequences in a database for motifs or patterns similar to motifs or patterns in the sequence of interest
- Search for a nucleotide sequence matching a protein sequence of interest as well as the other way around
- Compare sequences within taxonomic groups

With the goals above and the fact that it might be difficult to identify any good alignments between sequences only distantly related, as they often contain regions of low similarity, searching for local instead of global alignments is very important. The use of local alignment searches makes it possible to analyze for homology even between sequences containing regions of genetic variations adding too much noise for a global similarity search to make sense.

## How does the Smith-Waterman algorithm work?

The Smith-Waterman algorithm is searching for homology by comparing sequences. When sequences are compared using local alignments, the total number of alignments can be considerable, and identification of the best alignments is of high importance to both the reliability and relevance of the data obtained. This identification of the optimal local alignment between two sequences is basically what the Smith-Waterman algorithm does.

Optimal local alignments are identified by comparing the query sequence and the sequences in the database on a character-to-character level. Contrary to the Needleman-Wunsch algorithm, on which the Smith-Waterman algorithm is built, the Smith-Waterman algorithm is searching for local alignments, not global alignments, considering segments of all possible lengths to optimize the similarity measure [Smith and Waterman, 1981].

The algorithm is based on dynamic programming which is a general technique used for dividing problems into sub-problems and solving these sub-problems before putting the solutions to each small piece of the problem together for a complete solution covering the entire problem. Implementing the technique of dynamic programming, the Smith-Waterman algorithm finds the optimal local alignment considering alignments of any possible length starting and ending at any position in the two sequences being compared.

The basis of a Smith-Waterman search is the comparison of two sequences $A = (a_1 a_2 a_3 \dots a_n)$ and $B = (b_1 b_2 b_3 \dots b_m)$

The Smith-Waterman algorithm uses individual pair-wise comparisons between characters as:

$$H_{ij} = \max \begin{cases} H_{i-1,j-1} + s(a_i, b_j), \\ max_k\{H_{i-k,j} - W_k\}, \\ max_l\{H_{i,j-l} - W_l\}, \\ 0. \end{cases}$$

[Smith and Waterman, 1981]

$H_{ij}$ is the maximum similarity of two segments ending in $a_i$ and $b_j$ respectively.

Similarity of residues $a_i$ and $b_j$ is given by a weight matrix considering match, substitution or insertion/deletion.

First term considers an extension of the alignment by extending the two sequences compared by one residue each.

Second and third term handle an extension of the alignment by inserting a gap of length *k* into sequence *A* or sequence *B*, respectively.

Finally, fourth term placing a zero in the recursion ignores a possible negative alignment score. Preceding calculations will be started neutral and the allowance of the similarity score to be zero in the expression for $H_{ij}$ means that a local alignment can restart at any position performing a character-to-character comparison.

The algorithm assigns a score to each residue comparison between two sequences. By assigning scores for matches or substitutions and insertions/deletions, the comparison of each pair of characters is weighted into a matrix by calculation of every possible path for a given cell. In any matrix cell the value represents the score of the optimal alignment ending at these coordinates and the matrix reports the highest scoring alignment as the optimal alignment (see figure 1).

For constructing the optimal local alignment from the matrix, the starting point is the highest scoring matrix cell. The path is then traced back through the array until a cell scoring zero is met. Because the score in each cell is the maximum possible score for an alignment of any length ending at the coordinates of this specific cell, aligning this highest scoring segment will yield the highest scoring local alignment - the optimal local alignment.

| | | Sequence A | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | C | A | G | C | C | U | C | G | C | U | U | A | G |
| | A | 0,0 | 1,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 0,0 |
| | A | 0,0 | 1,0 | 0,7 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 0,7 |
| | U | 0,0 | 0,0 | 0,8 | 0,3 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 1,0 | 0,0 | 0,7 |
| | G | 0,0 | 0,0 | 1,0 | 0,3 | 0,0 | 0,0 | 0,7 | 1,0 | 0,0 | 0,0 | 0,7 | 0,7 | 1,0 |
| | C | 1,0 | 0,0 | 0,0 | 2,0 | 1,3 | 0,3 | 1,0 | 0,3 | 2,0 | 0,7 | 0,3 | 0,3 | 0,3 |
| | C | 1,0 | 0,7 | 0,0 | 1,0 | 3,0 | 1,7 | ? | | | | | | |
| Sequence B | A | | | | | | | | | | | | | |
| | U | | | | | | | | | | | | | |
| | U | | | | | | | | | | | | | |
| | G | | | | | | | | | | | | | |
| | A | | | | | | | | | | | | | |
| | C | | | | | | | | | | | | | |
| | G | | | | | | | | | | | | | |
| | G | | | | | | | | | | | | | |

Figure 1: *Calculation of the similarity matrix considering penalties for gap initiations and extensions. Values are assigned to each cell based on the parameter settings. Adapted from [McLysaght, ].*

## An example

The Smith-Waterman algorithm can be exemplified by the comparison of two sequences:

```
Sequence A: CAGCCUCGCUUAG
Sequence B: AAUGCCAUUGACGG
```

Parameters for the scoring matrix being:

$match = 1$

$mismatch = -\frac{1}{3}$

$gap = -(1 + \frac{1}{3}k)$, *k* being the gap extension number.

The similarity matrix is filled as shown in figure 1.

As any cell value represents the score of the optimal alignment ending at the cell coordinates, the highest scoring position in the matrix reports the ending point of the highest scoring and thereby the optimal alignment between the two sequences compared. To construct the optimal alignment, the starting point is the cell with the highest scoring value representing the last residue in this alignment. The complete alignment is identified by tracing back through the array from this highest scoring matrix cell until a cell scoring zero is met.

Illustrated in figure 2, the highest scoring cell is identified with the score of 3.3 and is traced back six steps. The search for local alignments allowing any position to be starting point and any position to be ending point means that the optimal alignment can be of any possible length and is thereby identified as the optimal *local* alignment.

| | | Sequence A | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | C | A | G | C | C | U | C | G | C | U | U | A | G |
| | | A | 0,0 | 1,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 0,0 |
| | | A | 0,0 | 1,0 | 0,7 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 0,7 |
| | | U | 0,0 | 0,0 | 0,8 | 0,3 | 0,0 | 0,0 | 0,0 | 0,0 | 0,0 | 1,0 | 1,0 | 0,0 | 0,7 |
| | | G | 0,0 | 0,0 | **1,0** | 0,3 | 0,0 | 0,0 | 0,7 | 1,0 | 0,0 | 0,0 | 0,7 | 0,7 | 1,0 |
| | | C | 1,0 | 0,0 | 0,0 | **2,0** | 1,3 | 0,3 | 1,0 | 0,3 | 2,0 | 0,7 | 0,3 | 0,3 | 0,3 |
| | | C | 1,0 | 0,7 | 0,0 | 1,0 | **3,0** | 1,7 | 1,3 | 1,0 | 1,3 | 1,7 | 0,3 | 0,0 | 0,0 |
| Sequence B | | A | 0,0 | 2,0 | 0,7 | 0,3 | 1,7 | 2,7 | 1,3 | 1,0 | 0,7 | 1,0 | 1,3 | 1,3 | 0,0 |
| | | U | 0,0 | 0,7 | 1,7 | 0,3 | 1,3 | **2,7** | 2,3 | 1,0 | 0,7 | 1,7 | 2,0 | 1,0 | 1,0 |
| | | U | 0,0 | 0,3 | 0,3 | 1,3 | 1,0 | 2,3 | **2,3** | 2,0 | 0,7 | 1,7 | 2,7 | 1,7 | 1,0 |
| | | G | 0,0 | 0,0 | 1,3 | 0,0 | 1,0 | 1,0 | 2,0 | **3,3** | 2,0 | 1,7 | 1,3 | 2,3 | 2,7 |
| | | A | 0,0 | 1,0 | 0,0 | 1,0 | 0,3 | 0,7 | 0,7 | 2,0 | 3,0 | 1,7 | 1,3 | 2,3 | 2,0 |
| | | C | 1,0 | 0,0 | 0,7 | 1,0 | 2,0 | 0,7 | 1,7 | 1,7 | 3,0 | 2,7 | 1,3 | 1,0 | 2,0 |
| | | G | 0,0 | 0,7 | 1,0 | 0,3 | 0,7 | 1,7 | 0,3 | 2,7 | 1,7 | 2,7 | 2,3 | 1,0 | 2,0 |
| | | G | 0,0 | 0,0 | 1,7 | 0,7 | 0,3 | 0,3 | 1,3 | 1,3 | 2,3 | 1,3 | 2,3 | 2,0 | 2,0 |

Figure 2: *Identification of optimal local alignment from similarity matrix. To identify the optimal local alignment comparing two sequences according to the Smith-Waterman algorithm, the highest scoring cell in the similarity matrix is identified. As any cell value represents the value of local alignments of arbitrary length ending at these specific coordinates, back tracing from the highest scoring cell leads to the highest scoring alignment - the optimal alignment. Adapted from [McLysaght, ].*

The alignment represented by the path shown in red in the similarity matrix in figure 2 is:

```
Sequence B:  G C C A U U G
Sequence A:  G C C - U C G
```

## Options/settings

Matrices, gap penalties including gap initial costs and gap extension costs, E-value etc are to be considered to get an optimal performance from a Smith-Waterman search.

See *Bioinformatics explained: BLAST* on http://www.clcbio.com/be/ to learn more about matrices, gap penalties and other options for parameter settings related to database similarity searching.

## How to search using the Smith-Waterman algorithm

Through the Japanese Institute of Bioinformatics Research and Development (BIRD) a public available software version of Smith-Waterman, SSEARCH, is accessible: http://www-btls.jst.go.jp/cgi-bin/Tools/SSEARCH/index.cgi. There are also commercial software packages available which perform Smith-Waterman searches.

## Smith-Waterman output

The result of the Smith-Waterman algorithm searching for optimal alignments is only returning one result - the optimal alignment - for each pair of compared sequences.

The output from the SSEARCH implementation of the Smith-Waterman algorithm is a list of optimal alignments between query sequence and database sequences together with each of the individual alignments.

**Acceptance No : 1178014090**
**Accepted in 2007/05/01 19:08:16 JST**
**Program : SSEARCH34.26**
**Database** Swiss-Prot Release 52.3 of 17-Apr-2007 (UniProt Knowledgebase Release 10.3)

### List of Entries with Homologous Regions

| ACCESSION | DEFINITION | Length | s-w | bits | E-Value |
|---|---|---|---|---|---|
| sp:O97649 | Mitochondrial uncoupling protein 3 | 308 | 2045 | 595.1 | 1.8e-169 |
| sp:Q9N2I9 | Mitochondrial uncoupling protein | 311 | 1913 | 557.0 | 5.4e-158 |
| sp:O77792 | Mitochondrial uncoupling protein | 311 | 1870 | 544.6 | 3e-154 |
| sp:P55916 | Mitochondrial uncoupling protein | 312 | 1863 | 542.5 | 1.2e-153 |
| sp:P56499 | Mitochondrial uncoupling protein 3 | 308 | 1797 | 523.5 | 6.5e-148 |
| sp:P56501 | Mitochondrial uncoupling protein | 308 | 1781 | 518.9 | 1.6e-146 |
| sp:O97562 | Mitochondrial uncoupling protein 2 | 309 | 1525 | 445.0 | 2.8e-124 |
| sp:Q3SZI5 | Mitochondrial uncoupling protein | 309 | 1499 | 437.5 | 5.1e-122 |
| sp:P70406 | Mitochondrial uncoupling protein | 309 | 1497 | 436.9 | 7.6e-122 |
| sp:P55851 | Mitochondrial uncoupling protein | 309 | 1491 | 435.2 | 2.5e-121 |
| sp:P56500 | Mitochondrial uncoupling protein 2 | 309 | 1490 | 434.9 | 3.1e-121 |
| sp:Q9N2J1 | Mitochondrial uncoupling protein | 309 | 1485 | 433.4 | 8.3e-121 |
| sp:Q9W725 | Mitochondrial uncoupling protein | 310 | 1464 | 427.4 | 5.6e-119 |

Figure 3: *List of optimal alignments. Performing the SSEARCH with NP_999214 as the query sequence against the Swissprot database gives these results (the list is only an excerpt from the web page) [SSEARCH, ].*

```
Query     1 MVGLKPPEVPPTTAVKLLGAGTAACFADLLTFPLDTAKVRLQIQGENQAARSAQYRGVLG  60
            :::::.: :::::::.::.:::::::::::::::::::::::::::::::: .:.:.:::::::
Subject   1 MVGLQPSEVPPTTVVKFLGAGTAACFADLLTFPLDTAKVRLQIQGENPGAQSVQYRGVLG  60

Query    61 TILTMVRNEGPRSPYNGLVAGLQRQMSFASIRIGLYDSVKQLYTPKGSDHSSITTRILAG 120
            :::::::.::::::::.:::::::.:::::::::::::::::::::.::::::.:::.. ::::: 
Subject  61 TILTMVRTEGPRSPYSGLVAGLHRQMSFASIRIGLYDSVKQFYTPKGADHSSVAIRILAG 120

Query   121 CTTGAMAVTCAQPTDVVKVRFQASIHAGPRSNRKYSGTMDAYRTIAREEGVRGLWKGILP 180
            :::::::::::::::::::::::: :. :  ..::: :::::::::::::::::::::  :
Subject 121 CTTGAMAVTCAQPTDVVKVRFQAMIRLGTGGERKYRGTMDAYRTIAREEGVRGLWKGTWP 180

Query   181 NITRNAIVNCAEMVTYDVIKEKVLDYHLLTDNLPCHFVSAFGAGFCATVVASPVDVVKTR 240
            :::::::::::::::::::.::::.:. ::.:::.:::::::::::::::::::::::::::
Subject 181 NITRNAIVNCAEMVTYDIIKEKLLESHLFTDNFPCHFVSAFGAGFCATVVASPVDVVKTR 240

Query   241 YMNSPPGQYQNPLDCMLKMVTQEGPTAFYKGFTPSFLRLGSWNVVMFVSYEQLKRALMKV 300
            :::.: :.:..:: ::::::.:::::::::::.::::::.:::.:::.:::::::::::::
Subject 241 YMNAPLGRYRSPLHCMLKMVAQEGPTAFYKGFVPSFLRLGAWNVMMFVTYEQLKRALMKV 300
```

Figure 4: *Optimal alignment of query sequence UCP3_Sus Scrofa and UCP3_Mouse [SSEARCH, ].*

Figure 5 shows the output of a similar Smith-Waterman search performed with the *CLC Bioinformatics Cell* using the *CLC Combined Workbench*.

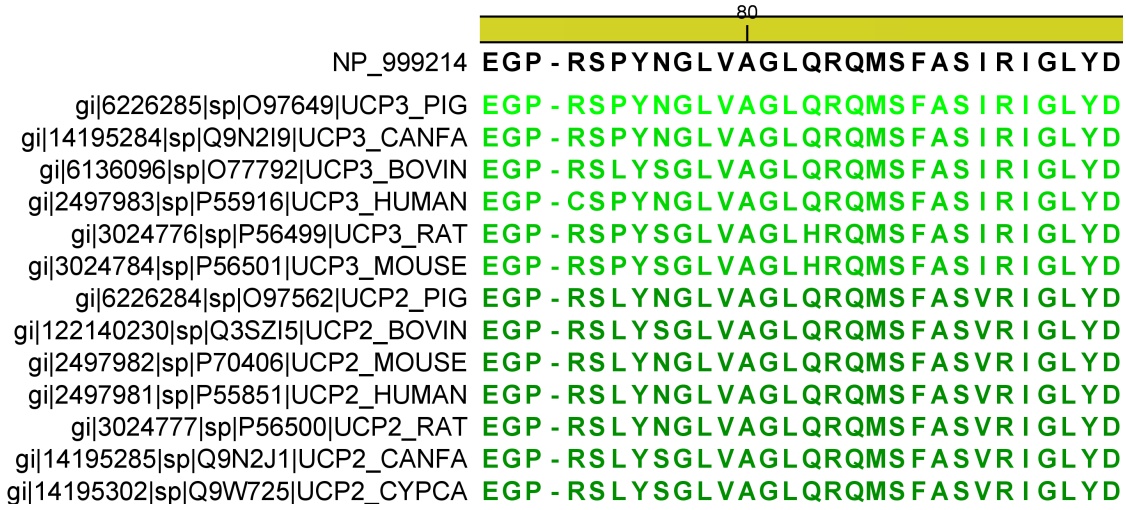


Figure 5: *Result of a similar Smith-Waterman search using the* CLC Bioinformatics Cell *using the* CLC Combined Workbench*. The alignment of all the hit sequences from the database is shown immediately.*

## Should I use Smith-Waterman or other algorithms for sequence similarity searching?

The Smith-Waterman algorithm is quite time demanding because of the search for optimal local alignments, and it also imposes some requirements on the computer's memory resources as the comparison takes place on a character-to-character basis.

The fact that similarity searches using the Smith-Waterman algorithm take a lot of time often prevents this from being the first choice, even though it is the most precise algorithm for identifying homologous regions between sequences.

BLAST and FastA are heuristic approximations of the Needleman-Wunsch and Smith-Waterman algorithms. These approximations are less sensitive and do not guarantee to find the best alignment between two sequences. However, these methods are not as time-consuming as they reduce computation time and CPU usage [Shpaer et al., 1996].

Today's research requires fast and effective data analysis. Algorithms like BLAST have therefore largely replaced Smith-Waterman searches as demands to time of handling large amounts of data are still getting stronger. On the other hand, large-scale projects are getting more and more prevalent and the researchers are becoming more and more concerned about the risk of missing important information if not using the most sensitive algorithm for database searches. As a consequence, the use of the Smith-Waterman algorithm is again becoming more and more widespread.

As the Smith-Waterman search guarantees to find optimal local alignments and returns only one result per comparison, it has to perform a larger number of computations than e.g. BLAST and this is the reason for the significantly slowed down process. Therefore a basic rule is that the Smith-Waterman algorithm should be used when getting the exact answer is more important than time.

An acceleration of the Smith-Waterman search is of great significance to the researcher today to

meet the request of both accuracy and a suitable time frame for handling large-scale research projects and the ever growing amount of sequence data. The Smith-Waterman algorithm can e.g. be accelerated based on FPGA chips or by using the SIMD technology (Single Instruction, Multiple Data) which parallelize and thereby accelerate the computations. An example of such acceleration is the CLC Bioinformatics Cell running the Smith-Waterman algorithm with a cell speed up to 5.2 GCUPS, which is around 110 times faster than a traditional software implementation of the algorithm [CLC bio, 2007].

### Other useful resources

Public available Smith-Waterman implementation from the Japanese Institute for Bioinformatics Research and Development http://www-btls.jst.go.jp/cgi-bin/Tools/SSEARCH/index.cgi

*Bioinformatics explained: BLAST*: Explanation of parameter settings and options for database sequence similarity searches http://www.clcbio.com/be/

### Creative Commons License

## References

[CLC bio, 2007] CLC bio (2007). CLC Bioinformatics Cell white paper. Available upon request to support@clcbio.com.

[McLysaght, ] McLysaght, A. Biological sequence comparison. http://www.gen.tcd.ie/molevol/.

[Needleman and Wunsch, 1970] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453.

[Sellers, 1974] Sellers, P. (1974). On the theory and computation of evolutionary distances. *SIAM J. Appl. Math.*, 26:787–793.

[Shpaer et al., 1996] Shpaer, E. G., Robinson, M., Yee, D., Candlin, J. D., Mines, R., and Hunkapiller, T. (1996). Sensitivity and selectivity in protein similarity searches: a comparison of smith-waterman in hardware to blast and fasta. *Genomics*, 38(2):179–191.

[Smith and Waterman, 1981] Smith, T. F. and Waterman, M. S. (1981). Identification of common molecular subsequences. *J Mol Biol*, 147(1):195–197.

[SSEARCH, ] SSEARCH. Japanese Institute of Bioinformatics Research and Development. http://www-btls.jst.go.jp/cgi-bin/Tools/SSEARCH/index.cgi.