UNIVERSITE D'EVRY
VAL D'ESSONNE

Département Informatique

LaMI

Laboratoire de Méthodes Informatiques

Sequence alignment : Extension of dynamic programming to pattern

J-P. Comet and J. Henry

e-mail : comet@lami.univ-evry.fr, Jacques.Henry@inria.fr

Rapport de Recherche n^o 60-2001

Mai 2001

Université d'Evry–Val d'Essonne
Cours Monseigneur Roméro
F–91025 Evry Cedex France

Abstract

The existing methods for alignments are based on edition costs computed additionnally position by position, according to a fixed substitution matrix: a substitution always has the same weight regardless of the position. Nevertheless the biologist favours any similitudes according to his knowledge of structures or functions of the sequences. In the particular case of proteins, we present a method consisting in making other information, such as patterns of the databank PROSITE*, operate in the classical dynamic programming algorithm. The method consists in making an alignment by dynamic programming taking a decision not only letter by letter as in the Smith & Waterman algorithm but also by giving a reward when aligning patterns.*

keywords: Dynamic programming, Sequence alignment, PROSITE, Databank scanning.

Introduction

Sequence comparison has become a central notion in modern molecular biology. It aims at generating by induction some information by comparing a new sequence to all sequences in annotated databanks. The alignment of two sequences allows to figure out different zones of similarity which are present in both sequences. To evaluate their similarity, a lot of methods are now available, allowing global alignments [Needleman and Wunsch, 1970] and gapped or ungapped local alignments[Smith and Waterman, 1981a, Smith and Waterman, 1981b]. The Smith & Waterman algorithm [Smith and Waterman, 1981b] answers exactly to the question of the search of the alignments with the best score. It finds one among the best local gapped alignments. It leads to an alignment score that can be used as a basis for determining a possible homology. Most of other approaches are based on heuristics[Lipman and Pearson, 1985, Karlin and Altschul, 1990, Altschul et al., 1997].

All these different algorithms use a substitution matrix independent of the position. The edition cost is additive position by position with a fixed substitution matrix and a fixed function for gap penalties. But when the biologist makes an alignment without computer, he favours some similitudes depending on his knowledge on structures and/or functions of sequences. In the case of proteins he tries to put in correspondence patterns which are known to be pertinent for the considered sequences. To explore this direction, some extension of the dynamic programming algorithm of Smith & Waterman have been developped in order to simulate a non-homogeneous substitution matrix. Wilbur and Lipman [Wilbur and Lipman, 1984] have presented a general frame for evaluating subsitution depending on the context. The main reason for introducing the context dependence is that biological mutations appear in a non-uniform way along the sequences [Lewin, 1990]. This implies a non-uniform distribution of the exact matches in the sequence alignments: there are regions where exact matches are more concentrated. On the other hand, if we align two random sequences which are independent and uniformly distributed, one observes that the exact matches are uniformly distributed [Huang, 1994]. Then an alignment with 5 consecutive exact matches is more significant than another alignment with 5 non consecutive exact matches. Huang also presents an algorithm taking into account the context favouring exact match if this one appears in a well conserved region.

We present here an alignment method which take into account the non-homogeneity of sequences. This heterogeneity comes from information not present in primary sequences. This method implements the dynamic programming which takes into account some information linked to biological “pattern”. Typically one thinks to protein sequences and functional patterns like these described in the databank PROSITE[Hofmann et al., 1999]. One would like to favour the alignment of patterns but one cannot impose it because this can outcome to impossibilities : if both sequences share two different patterns but in different order, the alignment of one pattern makes impossible the second alignment. This case could result from an inversion process during evolution.

Our method proceeds like the Smith & Waterman algorithm [Smith and Waterman, 1981b] by dynamic programming letter by letter, attributing a supplementary bonus/reward when patterns are matched. The first step is to determine the occurrences of databank patterns in both sequences with a classical "pattern-matching" algorithm. If one or more patterns are shared by both sequences, the second step implements a new dynamic algorithm for which the decision space has been widened : for each couple of position indices (i, j) which corresponds to the end of an alignment of two occurrences of the same pattern, one has the possibility of aligning patterns weighting this new path.

In the first section we show that the Smith & Waterman algorithm does not always align patterns in the predicted alignment. Several behaviours are observed depending on status of occurrences. Then the proposed method can improve results of the dynamic programming. The second focuses on the algorithm called SWP (Smith & Waterman algorithm with patterns). We decompose the algorithm in two parts : the first one consists in aligning two occurences of the same pattern respecting the matches in the motif, the second one is the general dynamic programming loop modified to take into account the possibility of aligning occurrences of patterns. The third section shows results of our algorithm in large scale environment : the observed distribution of aligned occurrences respects our expectation. The larger the weight, the more occurrences are aligned. We tested also our algorithm in databank scanning, and observed that SWP makes clearer some relationships between sequences. This algorithm has been tested on all patterns from the databank PROSITE which are present in the protein databank Swissprot Rel. 35. When both sequences share several patterns the algorithm allows to bind two similarity zones which are very far from each other.

1 Smith & Waterman alignments and Prosite patterns

1.1 Prosite patterns

PROSITE consists of a database of biologically significant sites and patterns formulated in such a way that with appropriate computational tools it can rapidly and reliably identify to which known family of protein (if any) the new sequence belongs. The databank PROSITE is a database of functionnal patterns determined by regular expression. In this databank there are also other descriptions of biological function (*Matrix, Rules*). In the PROSITE databank one can found 1275 patterns, 56 matrices and 4 rules. But here we will only consider pattern data.

Between all these patterns we eliminated some patterns because they appear very frequently in the protein databank Swissprot Rel. 35. These non-informative patterns which have been eliminated are :

Pattern PROSITE	Occurrence number in Swissprot	associated regular expression
PS00001	141147	N-P-[ST]-P
PS00004	42117	[RK](2)-x-[ST]
PS00005	332212	[ST]-x-[RK]
PS00006	377147	[ST]-x(2)-[DE]
PS00008	372978	G-{EDRKHPFYW}-x(2)-[STAGCN]-{P}
PS00009	23386	x-G-[RK]-[RK]
PS00013	7969	{DERK}(6)-[LIVMFWSTAG](2)-[LIVMFYSTAGCQ]-[AGS]-C
PS00016	4152	R-G-D
PS00029	4007	L-x(6)-L-x(6)-L-x(6)-L

Some of them are very short and often appear without the associated biological function. After elimination the databank consists of 1266 patterns.

To manipulate the PROSITE databank several tools are available. For example ProfileScan uses the method of Gribskov et al. [Gribskov et al., 1988] to find structural and sequence motifs in protein sequences. ProSearch [Kolakowski et al., 1992] allows direct searching of regular expressions on protein sequences, and for that purpose the PROSITE patterns must be translated into Unix-style regular expressions. ScanProsite allows to scan a sequence for the occurrence of PROSITE patterns from Swissprot or TrEMBL [Bairoch and Apweiler, 1997] - for the occurrence of patterns stored in the PROSITE database, and also allows to scan Swissprot and TrEMBL with a particular pattern.

Patterns are a certain characterization of biological properties shared by all sequences in a protein family. Nevertheless a pattern can appear in a protein without the associated biological property being expressed. Sequences which share the regular expression but not the biological property are called *false positive*, they are indexed in the PROSITE databank. In the same way some sequences share the biological property but do not present the regular expression. These sequences are called false negative, they generally are not indexed in the database.

In the sequel, we will use the UNIX syntax for regular expression and the classical algorithm for regular expression recognition, regexp. Every character represents one occurence of the character. The character "." stands for every character, the logical "or" is written "—", "[...]" stands for every charcater present in the bracket and "[ˆ...]" stands for every character not present in the bracket. "ˆ" and "$" represent the beginning and the end of the sequence. Finally "*", "+" have the classical significations.

For example, the regular expression "ˆ[ˆA][BC]A∗B." describes strings which are at the beginning of the line, of which the first letter is not "A" followed by "B" or "C" and by a certain number of "A" and ending with a "B" followed by another letter.

1.2 Does Smith & Waterman algorithm align patterns ?

Since we are looking for sequence alignments taking into account the information brought by PROSITE pattern, we naturally verify first that the classical Smith & Waterman alignment does not always match patterns shared by both sequences.

In order to test this fact, one has to separate five types of sequences sharing a particular pattern. All these five types are annotated in the DR line of each entry of PROSITE :

1. The **true positives** (annotated by **T**) are sequences presenting the regular expression modeling the pattern and which belong also to the biological family associated with the pattern.

2. The **false negative** (annotated by **N**) are sequences which belong to the family under consideration, but which have not been picked up by the regular expression.

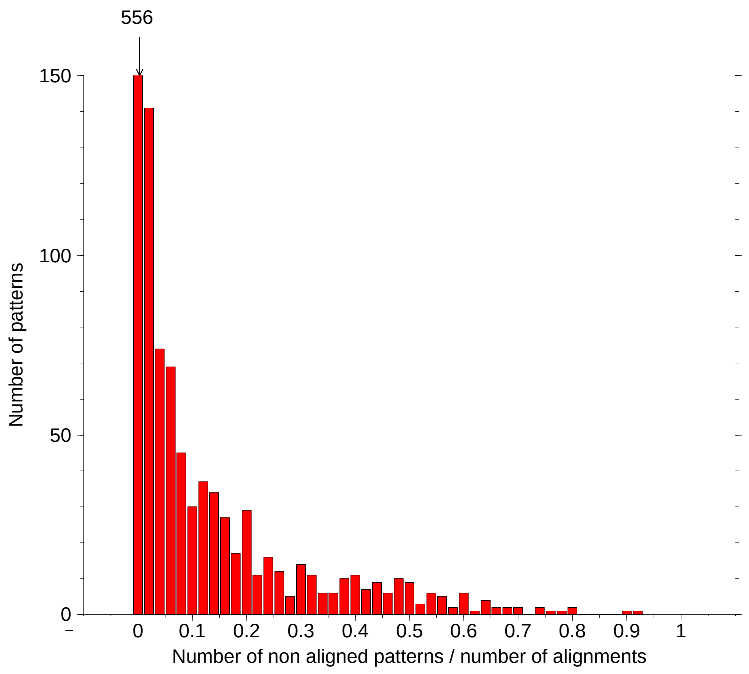


Figure 1: **Distribution of patterns from PROSITE Rel. 14.0 (true positives)** according to the number of patterns which are not aligned by dynammic programming. The databank is Swissprot Rel. 35. For the first coordinate value 0, one observes 556 patterns. For these 556 motifs, the classical dynamic programming aligns by default both occurrences of patterns (both sequences are true positive).

Sequences which do not belong to the set under consideration and do not present the regular expression, will be called **true negatives**.

3. The **false positives** (annotated by **F**) are sequences sharing the regular expression but which do not belong to the family.

4. The **potential** sequences (annotated by **P**) are the false negatives which do not present the regular expression because the region(s) that are used as a 'fingerprint' (pattern or profile) is not yet available in the data bank (partial sequence).

5. Finally the **unknown** (annotated by " **?**") are sequences for which one does not know whether they belong to the family or not.

For each type which presents the regular expression, we present the statistical behavior of the Smith & Waterman algorithm.

1.2.1 True Positives

We want to know if the classical dynamic programming algorithm does or does not align true positives. Figure 1 shows the distribution of patterns from databank PROSITE Rel. 35 according to the number of aligned patterns divided by the total number of possible alignments. In other words for each pattern of the databank,

- one computes the Smith & Waterman alignment for all pairs of true positives,
- one tests whether the Smith & Waterman alignment put both occurences of the regular expression in front of each other,
- one computes the number of alignments which do not align both patterns,
- finally the index p is the ratio of this number by the number of computed alignments.

The closer to 0 is the index p, the more numerous are the aligned occurrences of the regular expression. If $p = 0$, then for all pairs of true positives, the classical dynamic programming does align the occurrences of the pattern. At the opposite if $p = 1$, for all pairs of sequences sharing the regular expression, the Smith & Waterman algorithm does not align instances of the pattern.

Generally the Smith & Waterman algorithm aligns occurrences of the same pattern when occurrences are close to the region of the highest similarity. The algorithm does not align them when occurrences are not similar or when they are very short and do not belong to the most similar region.

One can notice that for a lot of patterns the classical dynamic programming allows to highlight a region including the pattern. There are 556 patterns for which the classical dynamic programming aligns the pattern in all cases. These patterns are not very informative because the alignment regroups instinctively these regions. In fact, either the pattern is very restrictive (only one word) or the occurrence of the pattern appears always in a region of high similarity. In the later case, one may ask whether the pattern is well defined. Since in all cases the region of high similarity is larger than the regular expression, it may be possible to extend the definition of the regular expression.

In some other cases the dynamic programming does not align the pattern. However for all true positives of Swissprot Rel. 35, the dynamic programming does align at least in one case the occurrences of the regular expression. In other words one cannot find one single pattern for which the index p is equal to 1. For a value of p close to 1 the histogram of figure 1 has some classes absolutely empty.

1.2.2 False positives

False positives versus false positives : In the same way, we want to know if the dynamic programming tends to align the same pattern in the case of two false positives. Generally the dynamic programming does not align patterns. Figure 2 shows the distribution of patterns.

False positives versus true positives : In that case the results are similar to the case of two false positives (see figure 2).

In such cases even if the biological function is not shared, the dynamic programming algorithm does align some patterns (see fig. 2, $p = 0$). This peak is much higher in the plot concerning two false positives than in the other concerning one false positive versus a true positive. One can align two false positive occurrences of the same pattern when they occur in the most similar zone between sequences. For example for the pattern PS00881 the Smith & Waterman algorithm aligns false positive occurrences in TIE1_BOVIN, TIE1_HUMAN and TIE1_MOUSE. All these three sequences belong to the same family and the pattern occurs in the characteristic and similar

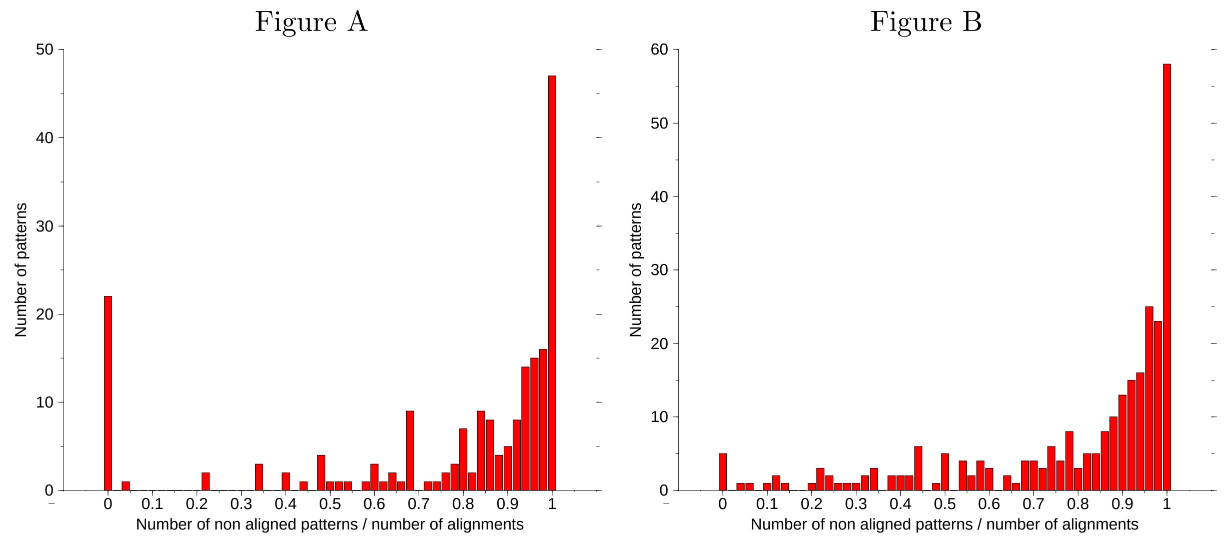


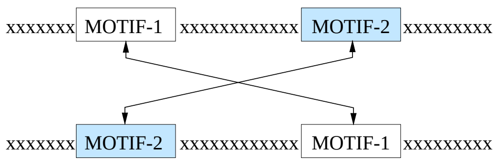
Figure 2: **Distribution of patterns from PROSITE Rel. 14.0 (false positives)** according to the number of non aligned motifs.
Figure A : False positive versus false positive.
Figure B : False positive versus true positive

region of this family. This phenomena does not happen in the case of aligning one false positive versus a true one. These homologies of two false positives could result from common biological properties different from the one related to the PROSITE motif.

2 The Smith & Waterman algorithm with pattern : SWP

Since the Smith & Waterman algorithm does not always align occurrences of the patterns, the alignment of these motifs can be a guide for an improvement of the sequence alignment. A better match between the occurrences is sought. This is an intermediate solution between the Smith & Waterman algorithm and the method used by biologists which consists in forcing the alignment of occurrences of motifs.

The aim of this new algorithm is to favor alignments which make two occurrences of the same pattern facing one another. If there is only one pattern shared by both sequences, or if one seeks to align only one pattern, it is possible to impose the alignment of the motif and then to extend the alignment on both sides with the Smith & Waterman algorithm.
However if both sequences share several motifs which appear in two different orders, one can no more force the alignment of patterns. One would come up to impossibilities. Let us consider that both sequences share two motifs in a different order :


xxxxxxx MOTIF-1 xxxxxxxxxxxx MOTIF-2 xxxxxxxxx
xxxxxxx MOTIF-2 xxxxxxxxxxxx MOTIF-1 xxxxxxxxx

If a biological relationship between these sequences exists, one faces to an inversion process. Classical algorithms are not able to align both motifs simultaneously, and one has to use algorithms which deal with inversion process [Holloway and Cull, 1994]. Note that our algorithm does not solve this inversion problem but chooses which pattern will be aligned.

The first stage is to determine all patterns which are shared by both sequences. Different programs allows to find in a sequence all occurrences of patterns. After having two lists of patterns present in both sequences, an intersection of the two lists gives the result. Another solution consists in taking advantage of the list given in PROSITE of all occurrences of a pattern in the databank Swissprot.

2.1 Alignment of two instances of the same pattern

The problem comes from the fact that a pattern can present insertions/deletions. For example, if the regular expression contains the term "$x\{6,8\}$", there is then at this place a string of which the length is between 6 and 8. Two instances of this pattern can have two different lengths. In that case the alignment has to present an insertion/deletion in the zone defined by the term "$x\{6,8\}$".

On the other hand if the motif has a constant length, the alignment is very simple because there is no insertion/deletion. For example the pattern named ASP_Protease is defined by the following regular expression :

```
[LIVMFGAC]-[LIVMTADN]-[LIVFSA]-D-[ST]-G-[STAV]-[STAPDENQ]-x-[LIVMFSTNC]-x-[LIVMFGTA]
```

The sequences bar1_yeast and tryp_astfl share this motif. The pattern has 2 instances in bar1_yeast at the positions 60 and 284 and one in tryp_astfl at the position 194 :

```
bar1_yeast:                (V)(L)(F)D(T)G(S)(A)x(F)x(V)
             60:   SQSLT   V  L  F D T G S  A D F W V     MDSSN

                           (V)(L)(L)D(S)G(T)(S)x(L)x(A)
             284   TTKYP   V  L  L D S G T  S L L N A     PKVIA

tryp_astfl:                (A)(A)(S)D(T)G(S)(T)x(L)x(G)
             194   SGGPL   A  A  S D T G S  T Y L A G     IVSWG
```

Alignment of the first occurrence in bar1_yeast with the instance in tryp_astfl will be :

```
      60     V L F D T G S A D F W V
                   | | | |
     194     A A S D T G S T Y L A G
```

If we decide to align the second instance in bar1_yeast, one will obtain :

```
     284     V L L D S G T S L L N A
                   |   |       |
     194     A A S D T G S T Y L A G
```

We define the score of the alignment as the sum of the substitution costs :

$$Score(Motif_A, Motif_B) = \sum_{k=0}^{long-1} S\Bigg(A[\beta_A + k], A[\beta_B + k] \Bigg)$$

where

- $Motif_A$ and $Motif_B$ are the occurrences of the same pattern in sequences A and B respectivelly,

- β_A and β_B are indices of the beginning of occurrences,

- *long* is the common length of both instances.

Nevertheless the alignment of occurrences of non constant length patterns is tricky. For example the Snake_Toxin is :

```
CPx{6,8}(L,I,V,Y,S,T)xCC
```

This pattern occurs in sequences hcy_octdo and nxl2_bunfl with two different lengths :

```
hcy_octdo :                CP   x{6}   (Y)xCC
             468   HGSTL   CP  SPEEPK   Y  ACC     LHGMP
nxl2_bunfl:                CP   x{8}   (L)xCC
              46   GCAAT   CPEFTSRYKS  L  LCC      TTDNC
```

In that sense one cannot simply apply the Needleman & Wunsch algorithm since insertions/deletions have to occur in a well specified region. The Needleman & Wunsch algorithm does not assure that the insertion will take place in the right region. At the opposite, the alignment makes sense if one tries to align first the brackets and braces : (,), { and }. The idea is to align the sequences

```
              {CP{SPEEPK}(Y)A(CC)}
       and
              {CP{EFTSRYKS}(L)L(CC)}
```

on an alphabet composed of 24 letters (20 letters for the amino acids and 4 for the brackets and braces). A new substitution is then built on this new alphabet :

- The values of a substitution of two amino acids do not change.
- the values for substitution $\Big(\text{“\{”},\alpha\Big)$, $\Big(\text{“(”},\alpha\Big)$, $\Big(\text{“\}”},\alpha\Big)$, $\Big(\text{“)”},\alpha\Big)$ where α is an amino acid are arbitrarily high,
- the value for perfect matches $\Big(\text{“\{”, “\{”}\Big)$, $\Big(\text{“\}”, “\}”}\Big)$, $\Big(\text{“(” , “(”}\Big)$ and $\Big(\text{“)”, “)”}\Big)$ are also arbitrarily high,
- and values for substitutions $\Big(\text{“\{”,“(”}\Big)$, $\Big(\text{“\{”,“)”}\Big)$, $\Big(\text{“\{”,“\}”}\Big)$, $\Big(\text{“\}”,“(”}\Big)$, $\Big(\text{“\}”,“)”}\Big)$, and $\Big(\text{“(”,“)”}\Big)$ are -∞.

The Needleman & Wunsch algorithm on this alphabet and with this substitution matrix gives a good alignment of patterns preserving insertion/deletion area. For the previous instances of Snake_Toxin pattern, one has the following alignment :

```
{CP{--SPEEPK}(Y)A(CC)}             CP--SPEEPKYACC
||||         || | |||||   that is  ||          ||
{CP{EFTSRYKS}(L)L(CC)}             CPEFTSRYKSLLCC
```

The score of alignment is the score obtained by the previous method from which one has substracted the weights due to the alignments of brackets and braces.

A better solution : During the scanning of sequences for hypothetical pattern matching, one can seek for possible positions of insertions/deletions. For example if the motif Snake_Toxin appears in the form CPSPEEPKYACC, the possible insertions/deletions during aligning with another occurence of the same pattern, will take place in the region *SPEEPK* and only in this zone. When aligning two instances of the same pattern, there are 2 stages :

1. Research of possible positions of insertions/deletions. Following information is stocked :
   - the number of insertion/deletion area
   - for each insertion/deletion area, the indices (b_i, e_i) representing the beginning and the end.

   Two tables are built, one for the sequence A (indices (b_i^A, e_i^A)) and the other one for sequence B (indices (b_i^B, e_i^B)).

2. Alignment of occurrences of the pattern : let (i, j) be the current indices of alignment.
   - If (i, j) does not belong to any insertion/deletion area, the letter A_i faces the letter B_j in the alignment and the indices are set to $(i + 1, j + 1)$.
   - If (i, j) belongs to the insersion/deletion area number k, the Needleman & Wunsch algorithm is used on subsequences $A[b_k^A, e_k^A]$ and $B[b_k^B, e_k^B]$, and one goes to the indices $(e_k^A + 1, e_k^B + 1)$.

   This second solution is actually implemented.

2.2 Local alignment with pattern

After having defined an alignment for two occurrences of the same pattern, we present an algorithm for aligning sequences taking into account the different patterns shared by them. For simplicity we consider here only linear penalty function for gaps (gap-open penaly and gap-extend penalty are considered to be equal to δ). In the case of affine penalty function the modifications are strictly the same.

The Smith & Waterman algorithm consists in computing for each pair of indices (i, j) the score for aligning the beginnings of sequences A[1..i] and B[1..j]. This is done with the following recurrence :

$$M(i,j) = \max \begin{pmatrix} M(i-1,j) - \delta, \\ M(i-1,j-1) + S(i,j), \\ M(i,j-1) - \delta, \\ 0 \end{pmatrix}.$$

A first method to favor the alignment of motifs consists in giving an additive reward (bonus) to the score when motifs match. With this new definition of the score, the dynamic programming algorithm can be applied but now at each step the decision variable consists in the choice of substitution, insertion, deletion of letters or in the alignment of a pattern. Let us suppose now that the sequence A contains a pattern at position $(i - long_1 + 1, i)$ and that the sequence B contains the same pattern at the position $(j - long_2 + 1, j)$, where $long_1$ and $long_2$ are lengths of pattern instances in sequences A and B respectively.

We consider another path beside the substitution, the insertion and the deletion : this new path is just the alignment of the pattern. The recurrence is then :

$$M(i,j) = \max \begin{pmatrix} M(i-1,j) - \delta, \\ M(i-1,j-1) + S(i,j), \\ M(i,j-1) - \delta, \\ 0, \\ M(i - long_1, j - long_2) + Score(motif_1, motif_2) + Bonus \end{pmatrix} \tag{1}$$

where

- $Score(motif_1, motif_2)$ is the score of the pattern alignment,
- $Bonus$ is a positive value rewarding the pattern alignment,
- $M(k,l)$ is the best score obtained when aligning $A_1A_2...A_k$ and $B_1B_2...B_l$.

In this new recurrence the score at the position (i,j) depends on the three neighbors like in the classical dynamic programming algorithm and also on another cell which can be far away from the current position : the score at the position corresponding to the beginning of pattern alignment.

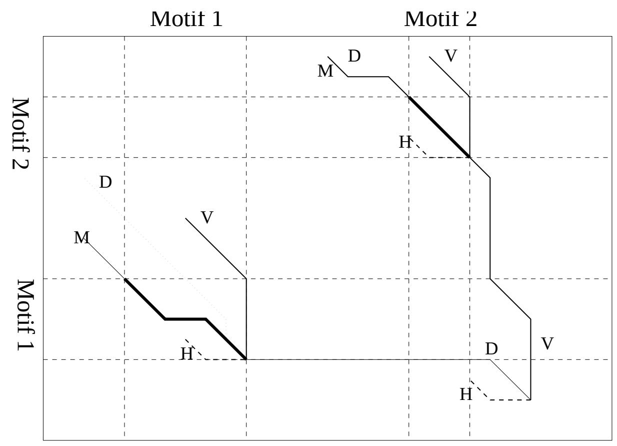


Figure 3: **Competition between patterns :** If both sequences share two different patterns which are incompatible, the algorithm chooses which motif will be aligned according to the weight of each pattern. Lines with M correspond to the alignment of motif, lines with H, D, or V to insertion, substitution and deletion of the last character.

In this method the alignment of motifs is not imperative and incompatible situation as in fig. 3 can be dealt with.

2.3 The basic recurrence of SWP

The SWP algorithm is a modification of the previous equation 1 taking into account the case when several patterns end at the current position.
Let A and B be two sequences of length n and m respectively. A couple of subsequences $(word_A, word_B)$

will be called a *pattern pair* if $word_A$ and $word_B$ belong to sequence A and B and if they are two occurrences of a same pattern.

Both sequences A and B can share several patterns. The same pattern can appear several times in the same sequence, and a position (i,j) can correspond to the end of several patterns. In that case the algorithm has to take the maximum among all alignments ending at the current position.

The general recurrence of SWP algorithm is the following :

$$M(i,j) = \max \left(\begin{array}{c} M(i-1,j-1) + S(A_i,B_j), \\ M(i-1,j) - \delta, \\ M(i,j-1) - \delta, \\ 0, \\ \max_{\left\{ \begin{array}{c} m, motif \\ \text{ending at } (i,j) \end{array} \right\}} \left(\begin{array}{l} M(a_1^m - 1, b_1^m - 1) + \\ Score(Word_m^A, Word_m^B) \\ \times Reward(Motif_m) \end{array} \right) \end{array} \right) \tag{2}$$

where

- $Score(Word_m^A, Word_m^B)$ is the alignment score of two occurrences of the motif m,
- $Reward(Motif_m)$ is a coefficient associated to the motif m weighting alignments of occurrences of this motif. Previously we introduced an additive weight (Cf. eq. 1) but here we present a multiplicative coefficient in order to try to weight according to the similarity of both instances of the motif. This multiplicative model will then favor the alignment of occurrences which are the most similar or the longest, that is which have the highest alignment score.

In all the sequel one takes into account the multiplicative model but shortcomings of this model will be discussed later.

2.4 Complexity

Three different stages compose the entire algorithm : searching all patterns shared by both sequences, the dynamic programming (see eq. 2) and aligning occurrences of patterns.

For two sequences of length m and n respectively, the complexity of the pattern matching algorithm is linear in $O(m+n)$ because the search has to be done in both sequences. When all patterns from a databank are taken into account, the complexity of this stage increases proportionnally with the size of the databank.

The complexity for the dynamic programming stage is quadratic that is in $O(m \times n)$ although a supplementary maximum operation has to be done for some cells.

For the third stage, alignment of the occurrences of a motif, two cases have to be considered.

- If the pattern has a fixed length, the algorithm is linear : the score is simply the sum of substitution costs corresponding of each position of the alignment.
- On the other hand if the length of the pattern is variable, the computation of the score needs the knowledge of possible insertion/deletion areas. Then the dynamic programming is used for each insertion/deletion area. The complexity is :

$$O\left(\sum_{i=1}^{Np} (e_i^A - b_i^A) \times (e_i^B - b_i^B)\right) + O\left(l_{m_A} + l_{m_B}\right)$$

where

- Np is the number of insertion/deletion areas defined by indices $(b_i^A, e_i^A)_{i\in[1,Np]}$ and $(b_i^B, e_i^B)_{i\in[1,Np]}$,
- l_{m_A} and l_{m_B} are the lengths of occurrences of the pattern m in sequences A and B respectively.

Table 1 sketches the complexity of the different stages of algorithm SWP.

	Complexity
searching shared patterns	$O((n+m)\times t)$
motif alignment	$O\left(l_{m_A} + l_{m_B} + \sum_{i=1}^{Np}(e_i^A - b_i^A)\times(e_i^B - b_i^B)\right)$
Dynamic Programming	$O(m\times n)$

Table 1: **Complexity of the SWP algorithm.** t stands for the motif databank size.

3 Discussion

3.1 Weight influence on alignments

Generally one expects the algorithm SWP to align more patterns than without weighting. Figure 4 shows distributions of patterns according to two different weights and for true positive versus true positive or for false positive versus false positive. For true positives one observes the concentration of the distribution nearby 0. The higher the weight, the higher the peak of the distribution. For false positives the distribution presents also a peak nearby 0 (see figure 4), but the proportion of non aligned motifs is higher.

3.1.1 Sequences sharing only one pattern

When there is only one pattern shared by sequences, the behavior of the SWP algorithm is simple. If the motif is not aligned by the classical dynamic programming, then increasing weight for the pattern leads to favor all paths aligning also the occurrences of pattern. Exception occurs when both occurrences are not similar (see further).

Figure 5 gives an example where the occurrences of the motif do not belong to the most similar region. When the weight is set to 2, the similarity area has changed, the algorithm retains only the similarity region containing the pattern.

Generally increasing weight for pattern can make appear an alignment with 2 similarity zones. The reward can balance the gap cost necessary to connect the two distant similarity regions. For the previous example (see fig. 5) the gap cost is so important that the pattern weight cannot balance it to connect both similarity regions.

When there are several occurrences of the same pattern in both sequences, each corresponding to a distinct similarity region, tuning weight can allow to align simultaneously these similarity zones in the same alignment. The maximum number of aligned patterns is also equal to the minimum between the two numbers of patterns occurrences in each sequence A and B.

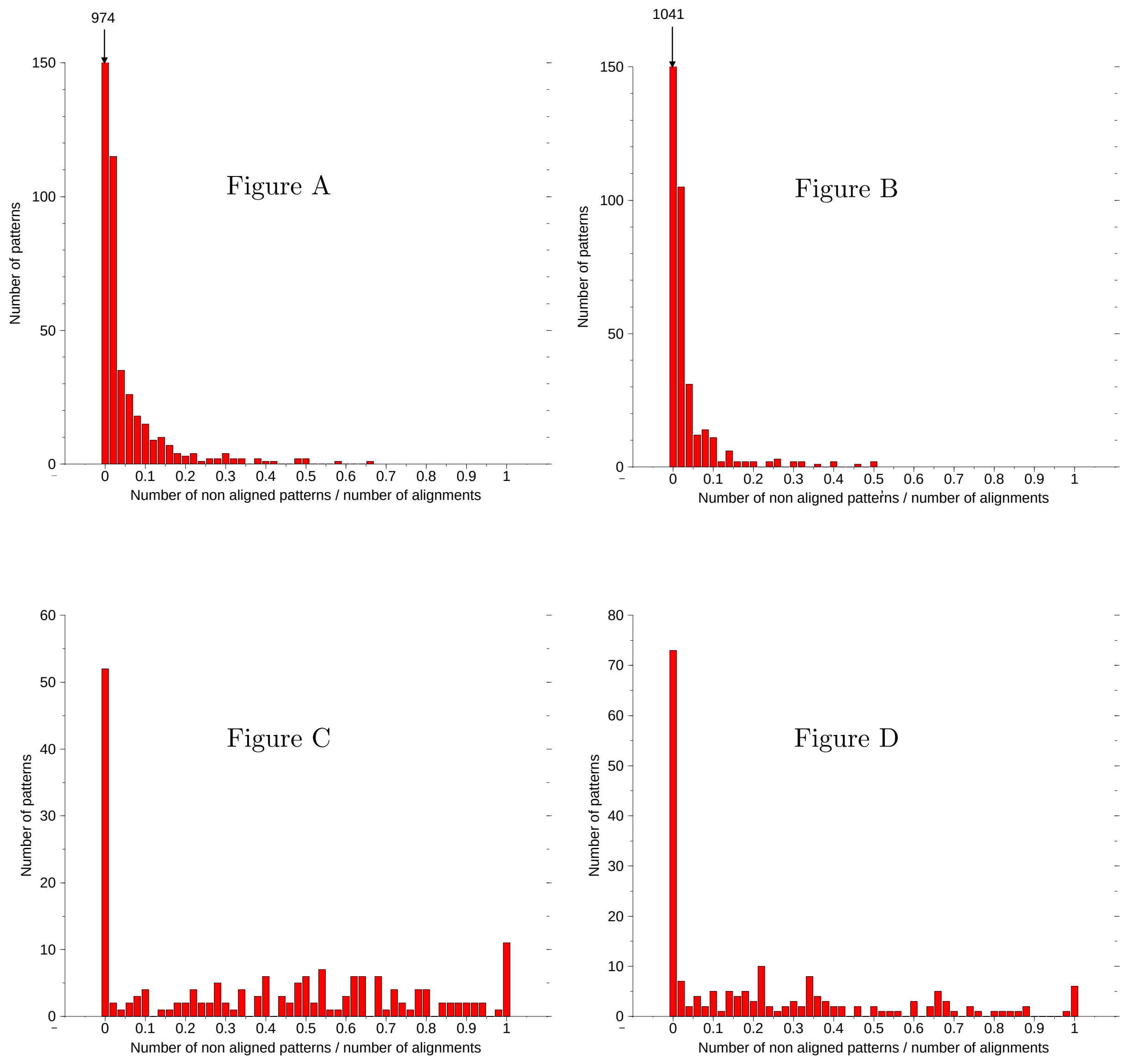


Figure 4: **Distribution of patterns from** PROSITE **Rel. 14.0 (true positive and false positive)** according to the number of not aligned motifs.
Figure A (true positive) : value for pattern weight = 2.
Figure B (true positive) : value for pattern weight = 3.
Figure C (false positive) : value for pattern weight = 2.
Figure D (false positive) : value for pattern weight = 3.

```
Motif PS00841 : [VI][KRE]P.[FYIL]VFDG.\{2\}[PIL].[LVC]K
```

```
            Weight = 1

125 VTPEMAWKLIIALREHGIESIVAPYEADAQLVYLEKENIIDGIITEDSDM
    ||  |       |   ||  |||| || ||   |       | || |||
797 VTGQMCLESQELLQLFGIPYIVAPMEAEAQCAILDLTDQTSGTITDDSDI

175 LVFGAQTVLFKMDGFGNCITIRRNDIANAQDLNLRLPIEKLRHMAIFSGC
      |||  |  |   |           |      | |   ||   |   |
847 WLFGARHV-YK-NFFSQNKHVEYYQYADIHN-QLGLDRSKLINLAYLLGS

225 DYTDGVAGMGLKTALRYLQKYP
    ||| |    |   |   |   |
894 DYTEGIPTVGYVSAMEILNEFP

{
            Score : 141


}
```

```
            Weight : 2

  1 GIKGLLGLLKPMQKSSHVEEFSGKTLGVDGYVWLHKAVFTCAHELAFNKE
    |  ||  ||             || | ||   ||  ||   |     |
  1 GVQGLWKLLECSGRPINPGTLEGKILAVDISIWLNQAVKG-ARDRQGNAI

 51 TDKYLKYAIHQALMLQYYGVKPLIVFDGGPLPCKASTEQKRKERRQEAFE
        |    |    |      |  ||||     |  |  ||  |   |
 50 QNAHLLTLFHRLCKLLFFRIRPIFVFDGEAPLLKRQTLAKRRQRTDKASN

101 LGKK
       |
100 DARK

 {
            Score : 150
            SW Score : 119
            Motif alignment Score : 31
}
```

Figure 5: **Weight influence on alignments.** For sequences XPG_XENLA and EXO1_SCHPO, the Smith & Waterman algorithm does not align the motif PS00841. Weighting the path corresponding to the pattern alignments allows to highlight another similarity region with SW score (119=150-31) slightly lower than the SW score. The frame corresponds to the only aligned occurrences of pattern PS00841.

3.1.2 Sequences sharing several patterns

As noticed before the region containing an occurrence of pattern can appear far away from a high similar region. Let us suppose that there are two similarity zones very distant from each other. In other words a very long insertion/deletion is needed to vizualize in the same alignment these two regions. If a pattern is present in each of these regions, SWP algorithm allows to connect these regions in the same alignment.

Let us choose a motif from PROSITE for which the proportion of non aligned occurrences among true positives is far from 1. The motif $PS01288$ has 4 true positives in Swissprot Rel. 35 : RTCB_ECOLI, Y682_METJA, YQ01_MYCTU and YT6J_CAEEL. Six alignments are possibles. the Smith & Waterman algorithm does not align pattern for 3 possible pairs (for example alignment between RTCB_ECOLI and Y682_METJA). The index p is then equal to 0.5. The sequences RTCB_ECOLI and Y682_METJA share also other patterns :

NAME	PROSITE ID	Regular Expression
PKC_PHOSPHO_SITE	PS00005	[ST]-x-[RK]
CK2_PHOSPHO_SITE	PS00006	[ST]-x(2)-[DE]
MYRISTYL	PS00008	G-EDRKHPFYW-x(2)-[STAGCN]-P
UPF0027	PS01288	Q-[LIVM]-x-N-x-A-x-[LIVM]-P-x-I-x(6)-[LIVM]-P-D-x-H-x-G-x-G-x(2)-[IV]-G

Alignment without weight does not align pattern UPF0027 since the more similar region is far away from occurrences of this pattern. This region contains two patterns PS00006 and PS00008. The Smith & Waterman alignment is given in figure 6. Weighting pattern alignment leads to an alignment which contains clearly two similar zones separated by a long gap. The weight has been sufficient to balance the penalty due to the long gap.

Higher the weight, greater the number of aligned motifs (see figure 7). But how many occurrences can be aligned ? Figure 7 shows that the maximum number of pattern alignments we can observe is 12. This number is maybe not the maximum number of occurrences which can be aligned, but due to non adapted weights.

Computing the maximum number of aligned motifs. The following table shows the number of occurrences of all patterns present in both sequences RTCB_ECOLI and Y682_METJA.

	RTCB_ECOLI	Y682_METJA	Symbol
PKC_PHOSPHO_SITE	8	11	P
CK2_PHOSPHO_SITE	2	6	C
MYRISTYL	11	13	M
UPF0027	1	1	V
Total	22	31	

Some of these occurrences overlap each other and the SWP algorithm is not able to align simultaneously such occurrences. On the other side one has to consider also the order of patterns. If two motifs occur in both sequences in different orders, it is impossible to align them.

The following procedure gives a method to know the maximum number of aligned motifs.

1. An alphabet with as many letters as different patterns shared by sequences is created. In the last example the alphabet is $\mathcal{A} = \{P, C, M, V\}$ (Cf. previous table).

	Coefficients
Below	1.0
Right	8.0

```
 22 EADARQQLINTAKMPFIFKHIAVMPDVHLGKGSTIGSVIPTK---GAIIP
    |     |  | |  | | |    ||||| | |  || |       | | |
 39 EPEVLEQIANVACLPGIYKYSIAMPDVHYGYGFAIGGVAAFDQREGVISP

 69 AAV-----------------------------------------------
      |
 89 GGVGFDINCLTSNSKILTDDGYYIKLEKLKEKLDLHIKIYNTEEGEKSSN

                           . . .

 74 --------------------------------------------GVDIGC
                                                |  ||
289 ASRIYSRKREVEIRNAYGDEYTSLCEDNSIKITSKAFALFMHKLGMPIGK

 78 GMN-----------------------------------------------

339 KTEQIYKIPEWIKKAPKWVKRNFLAGLFGADGSRAVFKNYTPLPINLTMS

                           . . .

 81 ALGTALTAEDLPENLAELRQAIETAVPHGRTT-GRCKRDKGAWENPPVNV
       | || |       ||       || |    |   |  |
589 LIRTNLTKEEVQSKIKELIKTLFKNVPSGLGSKGILKFSKSVMDDVLEE-

130 DAKWAELEAGYQWL----------------------TQKYPRFLNTNNYKHL
       ||  | || |                       |            |
638 GVRWAVKE-GYGWKEDLEFIEEHGCLKDADASYVSDKAKE----RGRVQL

160 GTLGTGNHFIEICL-----DESD---------QVWIMLHSGSRGIGNAIG
    | || |||| |        ||           ||     | |||| |  |
683 GSLGSGNHFLEVQYVEKVFDEEAAEIYGIEENQVVVLVHTGSRGLGHQIC

196 TYFIDLAQKEMQETLETLPSRDLAYFMEGTEYFDDYLKAVAWAQLFASLN
    |       |        || | ||      |    | ||       |  |
733 TDYLRIMEKAAKNYGIKLPDRQLACAPFESEEGQSYFKAMCCGANYAWAN

246 RDAMMENVVTALQSITQKTVRQPQTLAMEEINCHHNYVQKEQHFGEEIY-
    |      |                           |   || |
783 RQMITHWVRESFEEVFKIHAEDLEMNIVYDVAH--NIAKKEEHIIDGRKV

295 ---VTRKGAVSARA-------------GQYGIIPGSMGAKSFIVRGLGNE
       | ||||  |               ||  |||| ||  |   |  | |
831 KVIVHRKGATRAFPPKHEAIPKEYWSVGQPVIIPGDMGTASYLMR--GTE

329 ESFCSCS----HGAGRVMSRTKAKKLFSVEDQIRATAHVECR--KDAEVI
               |||||  || || ||       |  |        |
879 IAMKETFGSTAHGAGRKLSRAKALKLWKGKEIQRRLAEMGIVAMSDSKAV

373 D--EIPMAYKDIDAVMAAQSDLVEVIYTLR
       | | |||  | |             |
929 MAEEAPEAYKSVDLVADTC---HKAGISLK
```

```
 78 GMNALGTALTAEDLPENLAELRQAIETAVPHGRTT-GRCKRDKGAWENPP
    |     | || |       ||       || |    |  |  |
586 GVRLIRTNLTKEEVQSKIKELIKTLFKNVPSGLGSKGILKFSKSVMDDVL

127 VNVDAKWAELEAGYQWLT------------QKYPRFLNTNNYKH----LG
          ||    || |                                ||
636 E-EGVRWAVK-EGYGWKEDLEFIEEHGCLKDADASYVSDKAKERGRVQLG

161 TLGTGNHFIEICL-----DESD---------QVWIMLHSGSRGIGNAIGT
     || |||| |        ||           ||     | |||| |  | |
684 SLGSGNHFLEVQYVEKVFDEEAAEIYGIEENQVVVLVHTGSRGLGHQICT

197 YFIDLAQKEMQETLETLPSRDLAYFMEGTEYFDDYLKAVAWAQLFASLNR
           |        || | ||      |    | ||       |  ||
734 DYLRIMEKAAKNYGIKLPDRQLACAPFESEEGQSYFKAMCCGANYAWANR

247 DAMMENVVTALQSITQKTVRQPQTLAMEEINC-HHNYVQKEQHFGEEI--
          |                 | |       ||   || |
784 QMITHWVRESFEEVFKIHA---EDLEMNIVYDVAHNIAKKEEHIIDGRKV

294 --YVTRKGAVSARA-------------GQYGIIPGSMGAKSFIVRGLG--
       | ||||  |               ||  |||| ||  |   ||
831 KVIVHRKGATRAFPPKHEAIPKEYWSVGQPVIIPGDMGTASYLMRGTEIA

327 NEESFCSCSHGAGRVMSRTKAKKLFSVEDQIRATAHVE--CRKDAEVID-
      | | |  |||||  || || ||       |  |        |
881 MKETFGSTAHGAGRKLSRAKALKLWKGKEIQRRLAEMGIVAMSDSKAVMA

375 -EIPMAYKDIDAV
     | | |||  | |
931 EEAPEAYKSVDLV
```

Figure 6: **Weight influence on alignments (sequences RTCB_ECOLI and Y682_METJA).** Increasing pattern weight allows to introduce long gaps to connect distant similarity regions. Frames correspond to the pattern alignments.

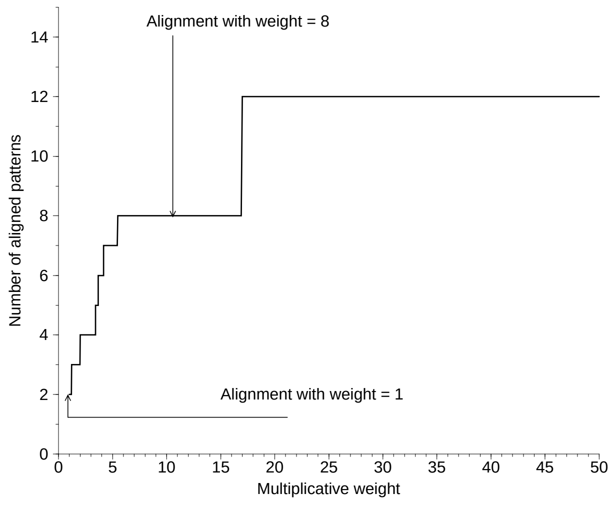


Figure 7: **Weight influence on the number of aligned occurrences of patterns (sequences RTCB_ECOLI and Y682_METJA).** Higher the weight, greater the number of aligned patterns. The saturation point seems to occur for 12 aligned motifs.

2. For each sequence one builds the series of letters corresponding to the orders of occurrences of different patterns. There can be several series of letters in case of overlap. In such a case only one pattern can be aligned. For the sequence Y682_METJA, one has two sequences:

$\boxed{V}$ P P M P C P P M M C P P M P P M C C M P M M M C M M P

$\boxed{M}$ P P M P C P P M M C P P M P P M C C M P M M M C M M P

   since motifs V and M overlap at the beginning of the sequence.

3. For each pair of sequences (A_i, B_j) where all A_i belong to the first databank and all B_j belong to the second, the longest subsequence is computed. The result can be achieved by the Smith & Waterman algorithm with the folowing parameters:
   - the substitution matrix is the identity matrix
   - gap-open and gap-extend penalties are set to 0.

4. The maximum score of all possible pairwise comparisons gives the maximum number of motifs which can be aligned. For the previous example there are $12 \times 2 = 24$ possible pairwise comparisons for which the maximum score is 15. Several alignments have this maximum score:

```
1 VPPMPCPPMMCPPMPPMCCMPMMMCMMP         3 PMPCPPMMCPPMPPMCCMPMMMCMMP
  |  | ||||| || | |  | |  |  |  S=15    || ||||| || ||      |||  |   S=15
1 V--M-CPPMM-PP-P-M--M-M--C--P         1 PM-CPPMM-PP-PP------MMC--P

1 VPPMPCPPMMCPPMPPMCCMPMMMCMMP         1 MPPMPCPPMMCPPMPPMCCMPMMMCMMP
  |  | ||||| || |     | |||  |  S=15    |  | ||||| || | |  | |  |  | S=15
1 V--M-CPPMM-PP-P-----P-MMC--P         1 M--M-CPPMM-PP-P-M--M-M--C--P

3 PMPCPPMMCPPMPPMCCMPMMMCMMP           1 MPPMPCPPMMCPPMPPMCCMPMMMCMMP
  || ||||| |  |     |||||  |    S=15    |  | ||||| || |     | |||  | S=15
1 PM-CPPMM-P--P-----PMMMC--P           1 M--M-CPPMM-PP-P-----P-MMC--P
```

We noticed before that the maximum number of pattern alignments is 12 for sequences RTCB_ECOLI and Y682_METJA. The order of the aligned motifs is given by the following sequence :

```
VPPMPPPMMMCP
```

The three missing pattern alignments could not to be aligned because of the multiplicative way of weighting.

3.2 Shortcomings of the multiplicative model

When regular expression are somewhat slack, the alignment of two occurrences of the motif can have a negative score. In such a case the multiplicative weight will never favor the alignment of occurrences. For example the pattern PS00011 is defined by the following regular expression :

```
.{12}E.{3}E.C.{6}[DEN].[LIVMFY].{9}[FYW]
```

The beginning of the pattern is defined by the presence of 12 non defined letters. Biologically that means that the useful pattern "E.{3}E.C.{6}[DEN].[LIVMFY].{9}[FYW]" cannot occur at less than 12 positions from the beginning of the sequence. Let us consider occurrences of the pattern PS00011 in OSTC_MACFA and FA10_BOVIN :

```
OSTC_MACFA : WLGAPAPYPDPLEPKREVCELNPDCDELADHIGFQEAY
FA10_BOVIN : FLEEVKQGNLERECLEEACSLEEAREVFEDAEQTDEFW
```

The pattern alignment is :

```
      .{12}     E .{3} E.C  .{6} [E].[F]    .{9}    [W]
   43 FLEEVKQGNLER E CLE   EAC SLEEAR E V F EDAEQTDEF  W
      |            |       | | |              |     |
    4 WLGAPAPYPDPL E PKR   EVC ELNPDC D E L ADHIGFQEA  Y
      .{12}     E .{3} E.C  .{6} [E].[L]    .{9}    [Y]
```

and the alignment score is −1. This final score is the sum of prefixe alignment score ("FLEEVKQGNLER" and "WLGAPAPYPDPL") and of the alignment score of the *useful pattern*. The first one is −14 and the second is +13. The prefixes are at the origin of the negative score of the alignment : the prefixes do not bring any information to the motif. To favor this alignment one could implement an additive reward. But the additive model does not favor similar occurrences any more than non similar ones.

3.3 Data-Bank scanning with SWP

The problem consists in finding all known sequences from a databank which are similar to a query sequence. The different algorithms give a score which allows to rank sequences from the most similar to the most distant. Is the SWP algorithm able to improve ranking of sequences ? Is the information brought by the pattern alignment sufficient to improve this databank ranking ?

Since SWP algorithm differs from Smith & Waterman algorithm only for pairs of sequences which share at least one pattern, we present results of databank scanning with a query sequence with a maximum number of patterns. If query sequence does not contain any pattern, the SWP scores are strictly the same than the Smith & Waterman score. Table 2 gives the distribution of PROSITE patterns in Swissprot.

Number of PROSITE patterns (n)	Number of sequences from Swissprot Rel. 35 with n patterns
1	24176
2	7610
3	2605
4	563
5	173
6	140
7	38
8	4

Table 2: **Distribution of PROSITE patterns in sequences from Swissprot.**

For example we choose the sequence FA12_HUMAN which has 7 different motifs. we compare then all sequences from Swissprot which share at least one pattern with FA12_HUMAN : PS00021, PS00022, PS00023, PS00134, PS00135, PS01186 and PS01253. Table 3 shows for each pattern present in FA12_HUMAN, the number of sequences from Swissprot which share these patterns and also the total number of occurrences.

PROSITE pattern	Number of sequences with pattern	Pattern occurrence number
PS00021	44	135
PS00022	321	1045
PS00023	29	55
PS00134	288	288
PS00135	287	287
PS01186	337	1071
PS01253	21	69

Table 3: **Number of pattern occurrences present in FA12_HUMAN.**

The patterns PS00022 and PS01186 have been moved apart for complexity reason since the number of occurrences was too large. Pattern PS00134 has not been retained because it was not so informative : [LIVM]-[ST]-A-[STAG]-H-C. Only patterns PS00021, PS00023, PS00135, and PS01253 have been retained. The databank of sequences sharing at least one of these 4 patterns has been created and contains 321 sequences.

	Coefficient = 1 (SW)			Coefficient = 2			Coefficient = 3		
	sequences	scores	(1)	sequences	scores	(1)	sequences	scores	(1)
1	FA12_CAVPO	2507	4	FA12_CAVPO	3019	4	FA12_CAVPO	3531	4
2	FA12_BOVIN	2425	4	FA12_BOVIN	2935	4	FA12_BOVIN	3445	4
3	HGFA_HUMAN	1193	4	HGFA_HUMAN	1485	4	HGFA_HUMAN	1777	4
4	UROT_HUMAN	712	3	URT2_DESRO	852	3	URT2_DESRO	997	3
5	URT1_DESRO	708	2	URT1_DESRO	850	3	URT1_DESRO	993	3
6	URT2_DESRO	707	3	UROT_HUMAN	849	3	UROT_HUMAN	986	3
7	UROT_RAT	694	3	UROT_RAT	832	3	UROT_RAT	970	3
8	UROT_BOVIN	687	3	UROT_BOVIN	826	3	UROT_BOVIN	965	3
9	UROT_MOUSE	675	3	UROT_MOUSE	808	3	UROT_MOUSE	941	3
10	URTB_DESRO	666	2	URTB_DESRO	769	2	URTB_DESRO	872	2
11	UROK_BOVIN	663	2	UROK_BOVIN	763	2	UROK_BOVIN	863	2
12	UROK_RAT	656	2	UROK_HUMAN	754	2	UROK_HUMAN	854	2
13	UROK_HUMAN	654	2	UROK_RAT	750	2	UROK_RAT	844	2
14	UROK_PIG	639	2	UROK_PIG	739	2	UROK_PIG	839	2
15	UROK_PAPCY	628	2	UROK_PAPCY	728	2	UROK_PAPCY	828	2
16	UROK_MOUSE	623	2	UROK_MOUSE	718	2	UROK_MOUSE	813	2
17	PLMN_PIG	587	2	PLMN_PIG	687	2	PLMN_PIG	787	2
18	PLMN_BOVIN	582	2	PLMN_BOVIN	682	2	PLMN_BOVIN	782	2
19	PLMN_HUMAN	582	2	PLMN_HUMAN	682	2	PLMN_HUMAN	782	2
20	URTG_DESRO	571	2	URTG_DESRO	674	2	URTG_DESRO	777	2
87	CFAD_RAT	347	1	ENTK_MOUSE	412	1	**FINC_HUMAN**	475	2
88	TRY2_ANOGA	345	1	TRY2_ANOGA	412	1	**FINC_RAT**	474	2
100	EL2_BOVIN	334	1	TRYA_HUMAN	397	1	***FINC_BOVIN***	462	2
218	COGS_UCAPU	247	1	**FINC_HUMAN**	304	2	CTRB_GADMO	379	1
219	SNAK_DROME	247	1	**FINC_RAT**	301	2	TRYZ_DROME	377	1
224	NKP1_RAT	237	1	***FINC_BOVIN***	295	2	EL3B_HUMAN	372	1
273	**FINC_HUMAN**	147	1	TRYP_CHOFU	243	1	GRAB_MOUSE	305	1
274	***FINC_BOVIN***	147	1	MPRI_MOUSE	240	1	MCP1_PAPHA	303	1
279	**FINC_RAT**	142	1	GRL2_RAT	229	1	MCP2_MERUN	285	1

Table 4: **Databank scanning with SWP (Extract).** Sequence FA12_HUMAN has been compared with all sequences from Swissprot Rel. 35 which do share at least one of the following patterns: PS00021, PS00023, PS00135 and PS01253. When the coefficient is 1, the SWP conincides with SW algorithm. Columns (1) give the number of aligned patterns.

Table 4 gives the 20 greatest scores when comparing sequence FA12_HUMAN against one of the 321 considered sequences. Several coefficients for weighting have been tested. Globally one observed a stability of the order in which sequences appear. But some sequences have been catched up when weight is greater than 1 (see sequences FINC_HUMAN, FINC_BOVIN, FINC_RAT, and EL3B_HUMAN).

For example the sequence FINC_HUMAN occurs at the rank 273 with pattern weight equal to 1 and is ranked at the position 87 with coefficient equal to 3. Increasing weight has allowed to connect two similarity regions (see figure 8).

The advantage of the SWP algorithm resides not only in ranking sequences but also in alignment itself. It allows in a large measure to correct the lack of sensibility of Smith & Waterman algorithm, allowing creation of long insertions/deletions.

```
            Weight : 1

394 DQKYSFCTDHTVLVQTQGGNSNGALCHFPFLYNNHNYTDCTSEGRRDNMK
      ||     |||     |       ||||| |    |  ||  ||
27  EHKYKAE-EHTVVLTVTG-----EPCHFPFQYHRQLYHKCTHKGRPGPQP

444 WCGTTQNYDADQKFGFC
    || || | | ||  | |
71  WCATTPNFDQDQRWGYC


Motif 1 :

C.\{2\}PF.[FYWI].\{7\}C.\{8,10\}WC.\{4\}[DNSR][FYW]
.\{3,5\}[FYW].[FYWI]C


Motif 2 :

C.\{6,8\}[LFY].\{5\}[FYW].[RK].\{8,10\}C.C.\{6,9\}C


            Score : 147
```

```
            Weight : 3

394 DQKYSFCTDHTVLVQTQGGNSNGALCHFPFLYNNHNYTDCTSEGRRDNMK
      ||     |||     |       ||||| |    |  ||  ||
27  EHKYKAE-EHTVVLTVTG-----EPCHFPFQYHRQLYHKCTHKGRPGPQP

444 WCGTTQNYDADQKFGFCPMAAHEEICTTNEGVMYRIGDQWDKQHDMGHMM
    || || | | ||  | |                           | |   |
71  WCATTPNFDQDQRWGYCLE------------------PKKVKDHCSKHSP

494 RCTCVGNGRGEWTCIAYSQLRDQCIVDDITYNVNDTFHKRHEEGHMLNCT
       |   |    ||                       |
103 ---CQKGG----TCVNM----------------PSGPHCL----------

544 CFGQGRGRWKCDPVDQCQDSETGTFYQIGDSWEKYVHGVRYQCYCYGRGI
    |     |   |     |       |      |          | | |
120 CPQHLTGNH-CQKEK-CFEPQLLRFFHKNEIWYRTEQAAVARCQCKGPDA

594 GEWHCQPLQTYPSSSGP
       ||| |        |
168 ---HCQRLASQACRTNP

      Score : 475
      SW  score : 133
      pattern alignment scores : 34 and 137
```

Figure 8: **Alignment of sequences FA12_HUMAN and FINC_HUMAN with two different weights.**

3.4 Weighting problem

The main recurrence 2 uses coefficients to favor the pattern alignment. The $Reward(Motif_m)$ has to depend on aligned pattern since the information brought by a pattern depends on its length and on its grammar. It remains hard to assign some numerical values for each pattern of the PROSITE databank.

The weight has to depend on the information level of the pattern. The more informative the pattern, the greater the associated weight. The information brought by a pattern is inversely proportionnal to the probability of observing an occurrence of the pattern in a random sequence. The information theory says that the information of the event is equal to $-log_2(p)$ [McEliece, 1977] where p is the probability of the event. One has to compute or estimate the probability law of observing a pattern in a random sequence. Nicodème and all [Nicodème et al., 1999, Nicodème et al.,] presented a complete analysis of the statistics of number of pattern occurrences in a random text. The characteristics of the distribution of the number of pattern occurrences are effectively computable, both exactly and asymptotically. The probability of pattern occurrences depends also on the string length.

Conclusion

We proposed a new dynamic programming algorithm called SWP which is a modification of the SW algorithm. It incoporates more biological information coming from the occurrences of PROSITE motifs. In databank scanning SWP can reveal similarities between sequences which otherwise would be hidden. Furthermore the method allows alignments with large insertion/deletion areas. The rewarding method remains one possibility among others and further works are needed to give it statistical and theoretical basis.

Acknowledgement

The authors are much indebted to J.-J. Codani and E. Glémet for the use of LASSAP [Glémet and Codani, 1997] to implement this algorithm in large scale environment.

References

[Altschul et al., 1997] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped blast and psi-blast : a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402.

[Bairoch and Apweiler, 1997] Bairoch, A. and Apweiler, R. (1997). The swiss-prot protein sequence data bank and its supplement trembl. *Nucleic Acids Res.*, 25(1):31–36.

[Glémet and Codani, 1997] Glémet, E. and Codani, J.-J. (1997). Lassap : a large scale sequence comparison package. *Comp. Appl. BioSci.*, 13(2):137–143.

[Gribskov et al., 1988] Gribskov, M., Homyak, M., Edenfield, J., and Eisenberg, D. (1988). Profile scanning for three-dimensional structural patterns in protein sequences. *Comput. Appl. Biosci.*, 4(1):61–66.

[Hofmann et al., 1999] Hofmann, K., Bucher, P., Falquet, L., and Bairoch, A. (1999). The prosite database, its status in 1999. *Nucleic Acids Res.*, 27:215–219.

[Holloway and Cull, 1994] Holloway, J. L. and Cull, P. (1994). Aligning genomes with inversions and swaps. In *ISMB*, pages 195–202.

[Huang, 1994] Huang, X. (1994). A context dependent method for comparing sequences. In *5th Annual Symposium CPM*, volume 807 of *LNCS*, pages 54–63. Springer-Verlag.

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

[Kolakowski et al., 1992] Kolakowski, L., Leunissen, J., and Smith, J. (1992). Prosearch: fast searching of protein sequences with regular expression patterns related to protein structure and function. *Biotechniques*.

[Lewin, 1990] Lewin, B. (1990). *Genes IV.* Cell Press.

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

[McEliece, 1977] McEliece, R. J. (1977). *The theory of information and coding.* Encyclopedia of mathematics and its applications. Addison-Wesley.

[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.*, 48:443–453.

[Nicodème et al.,] Nicodème, P., Salvy, B., and Flojolet, P. Motif statistics. *Theoretical Computer Science, to appear.*

[Nicodème et al., 1999] Nicodème, P., Salvy, B., and Flojolet, P. (1999). Motif statistics. Technical report, INRIA, France.

[Smith and Waterman, 1981a] Smith, T. F. and Waterman, M. S. (1981a). Comparision of biosequences. *Advan. Appl. Math.*, 2:482–489.

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

[Wilbur and Lipman, 1984] Wilbur, W. J. and Lipman, D. J. (1984). The context dependent comparison of biological sequences. *SIAM J. Appl. Math.*, 44:557–567.