# A New Dynamic Programming Algorithm for Multiple Sequence Alignment

Jean-Michel Richer, Vincent Derrien, Jin-Kao Hao
jean-michel.richer@univ-angers.fr
http://www.info.univ-angers.fr/pub/richer

University of Angers
LERIA - Laboratoire d'Etudes et de Recherche
en Informatique d'Angers, France

COCOA'07
Xi'an, China, August 12-15, 2007


A New Dynamic Programming Algorithm for Multiple Sequence Al

# Introduction

## Aim of this work

Express alignment of two alignments with linear or affine gap model penalty using the Dynamic Programming (DP) framework : aligning alignments.

# Expression of DP

- pairwise sequence alignment
  - linear gap penalty : yes
  - affine gap penalty : yes
- multiple sequence alignment
  - linear gap penalty : yes (3 sequences)
  - affine gap penalty : ?
- alignment of alignments
  - linear gap penalty : probably (our contribution)
  - affine gap penalty : no (our contribution)

# Outline

# A word on proteins

## Biological framework

- our genes contain the code of proteins
- proteins are the basic elements of living cells
- they have a 3D stucture that gives them their function
- but most of the time we only have their primary structure

## Similarity $\longrightarrow$ function

Thanks to sequence similarity we can infer proteins functions

# What is an alignment ?

## Aligning

Given a set of (Nucleic or Amino Acid) sequences, find the best way to display relevant information about conserved regions of the sequences.

# Example

## Example

1aab_ref1 from BaliBASE

| | |
|---|---|
| hmgl_trybr | KKDSNAPKRAMTSFMFFSS ... |
| hmgt_mouse | KPKRPRSAYNIYVSESFQE ... |
| hmgb_chite | ADKPKRPLSAYMLWLNSAR ... |
| hmgl_wheat | DPNKPKRAPSAFFVFMGEF ... |

Table: Set of unaligned sequences

Sequences are expressed over an alphabet $\Sigma = \{A, C, D, E, \ldots\}$ .

# Example

## Example

| | |
|---|---|
| hmgl_trybr | -KKDSNAPKR AMTSFMFFSS ... |
| hmgt_mouse | ------KPKR PRSAYNIYVS ... |
| hmgb_chite | ----ADKPKR PLSAYMLWLN ... |
| hmgl_wheat | ---DPNKPKR APSAFFVFMG ... |

Table: Set of aligned sequences

- we use the gap symbol - to align sequences.
- we define $\Sigma' = \Sigma \cup \{-\}$.
- edit operations : match, substitution, insertion/deletion

# Formal definition

## Definition (Alignment)

- let $S = \{S_1, \ldots, S_k\}$ be a set of sequences defined over an alphabet $\Sigma$,
- $\forall u \in \{1, \ldots, k\}, S_u = \langle x_1^u, \ldots, x_{|S_u|}^u \rangle$ where $|S_u|$ is the length of $S_u$.

An alignment $A^S$ is a matrix:

$$A^S = \begin{bmatrix} a_1^1 & \ldots & a_q^1 \\ \vdots & & \vdots \\ a_1^k & & q_q^k \end{bmatrix}$$

# Formal definition

## Definition (Alignment)

such that :

1. $a_j^i \in \Sigma'$
2. alignment length (number of columns) is coherent with sequences lengths : $max(|S_u|) \leq q \leq \sum_{u=1,\ldots,k} |S_u|$
3. there is no column composed only with gaps
4. removing gaps from $\langle a_1^i, \ldots, a_q^i \rangle$ gives $S_i$

# What type of alignment method do we use ?

- heuristic / approximated method : Blast
- exact method : Dynamic Programming

# What do we need to align sequences exactly ?

- a substitution matrix $w : \Sigma' \times \Sigma' \rightarrow \mathbb{R}$, to score matches and substitutions (PAM, BLOSUM, GONNET)
- a gap model $g : \mathbb{N} \rightarrow \mathbb{R}$, to score gaps (linear, affine, others)

# Gap model

## Definition (gap model)

A gap model is an application $g : \mathbb{N} \rightarrow \mathbb{R}$ which assigns a score, also called a penalty, to a set of $n$ consecutive gaps.

## Definition (linear gap model)

$g(n) = n \times g_o$ where $g_o < 0$ is the opening penalty.

## Definition (affine gap model)

$$g(n) = \begin{cases} 0 & \text{if} \quad n = 0 \\ g_o + (n-1) \times g_e & \text{if} \quad n \geq 1 \end{cases}$$

# How can we score and obtain an alignment ?

## Definition (Sum-of-pairs)

Let $A^S$ be an alignment of a set of sequences $S = \{S_1, \ldots, S_k\}$. The sum-of-pairs is given by the following formula:

$$sop(A^S) = \sum_{c=1}^{q} sop^c(A_c^S)$$

where $sop^c(A_c^S)$ is the score of the $c$ column of the alignment given by:

$$sop^c(A_c^S) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} w(a_c^i, a_c^j) - \#gaps$$

# Pairwise Sequence Alignment

## Definition (Needleman-Wunsch-Sellers, 1970)

Given two sequences :

- $x = \langle x_1, \ldots, x_m \rangle$
- $y = \langle y_1, \ldots, y_n \rangle$

we can compute an alignement of $x$ and $y$ using a matrix $M[(n+1) \times (m+1)]$ :

$$M_{i,j} = max \begin{cases} M_{i-1,j-1} & + & w(x_j, y_i) \\ M_{i-1,j} & + & g_o \\ M_{i,j-1} & + & g_o \end{cases}$$

# Pairwise Sequence Alignment with Linear Gap

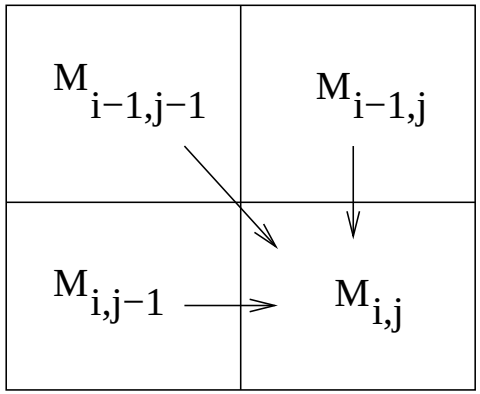
| | | $x_1$ | $x_2$ | | | $x_m$ |
|---|---|---|---|---|---|---|
| | 0 | $-g$ | $-2g$ | | | $-mg$ |
| $y_1$ | $-g$ | | | | | |
| | | | | .... | | |
| | | | | | | |
| $y_n$ | $-ng$ | | | | | |

Given to sequences :

- $x = \langle x_1, \ldots, x_m \rangle$
- $y = \langle y_1, \ldots, y_n \rangle$

we can compute an alignement of $x$ and $y$ using a matrix $M[(n+1) \times (m+1)]$

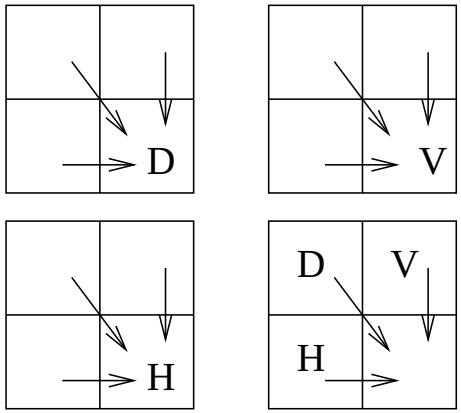
# Pairwise Sequence Alignment with Linear Gap




$$M_{i,j} = max \begin{cases} M_{i-1,j-1} & + & w(x_j, y_i) \\ M_{i-1,j} & + & g_o \\ M_{i,j-1} & + & g_o \end{cases}$$

with $g_o < 0$

# Pairwise Sequence Alignment with Affine Gap




We need to use 3 matrices to make a disctinction between :

- an opening penalty
- or an extension of an existing penalty

# Pairwise Sequence Alignment with Affine Gap

$$M_{i,j} = max \begin{cases} D_{i,j} = max \begin{cases} H_{i-1,j-1} + w(x_j, y_i) \\ D_{i-1,j-1} + w(x_j, y_i) \\ V_{i-1,j-1} + w(x_j, y_i) \end{cases} \\ H_{i,j} = max \begin{cases} H_{i,j-1} + g_e \\ D_{i,j-1} + g_o \\ V_{i,j-1} + g_o \end{cases} \\ V_{i,j} = max \begin{cases} H_{i-1,j} + g_o \\ D_{i-1,j} + g_o \\ V_{i-1,j} + g_e \end{cases} \end{cases}$$

# Our contribution

Our contribution...

# Aligning alignments

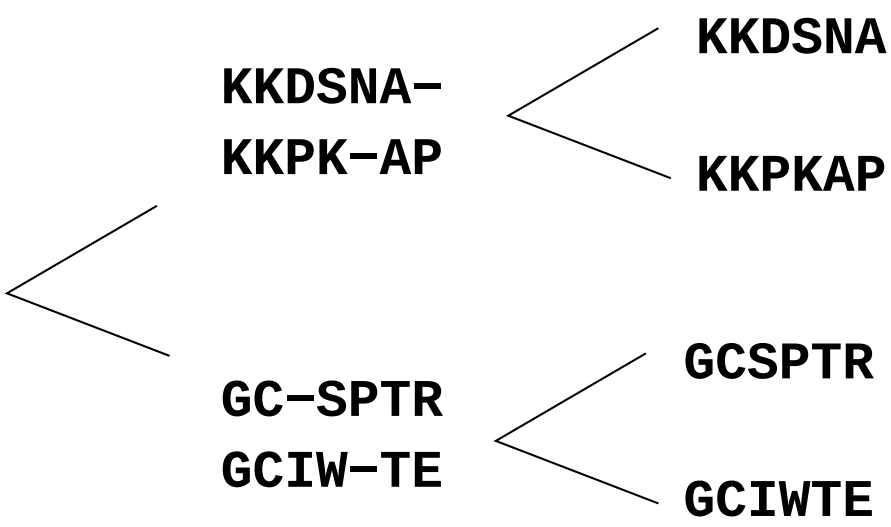
## Aligning alignments

Given two multiple alignments $A_v$ and $A_h$, find an optimal alignment of $A_v$ and $A_h$ for the sum-of-pairs objective function for a given substitution matrix $w$ and gap model $g(n)$ (John Kececioglu 1998).

## Kececioglu 2004

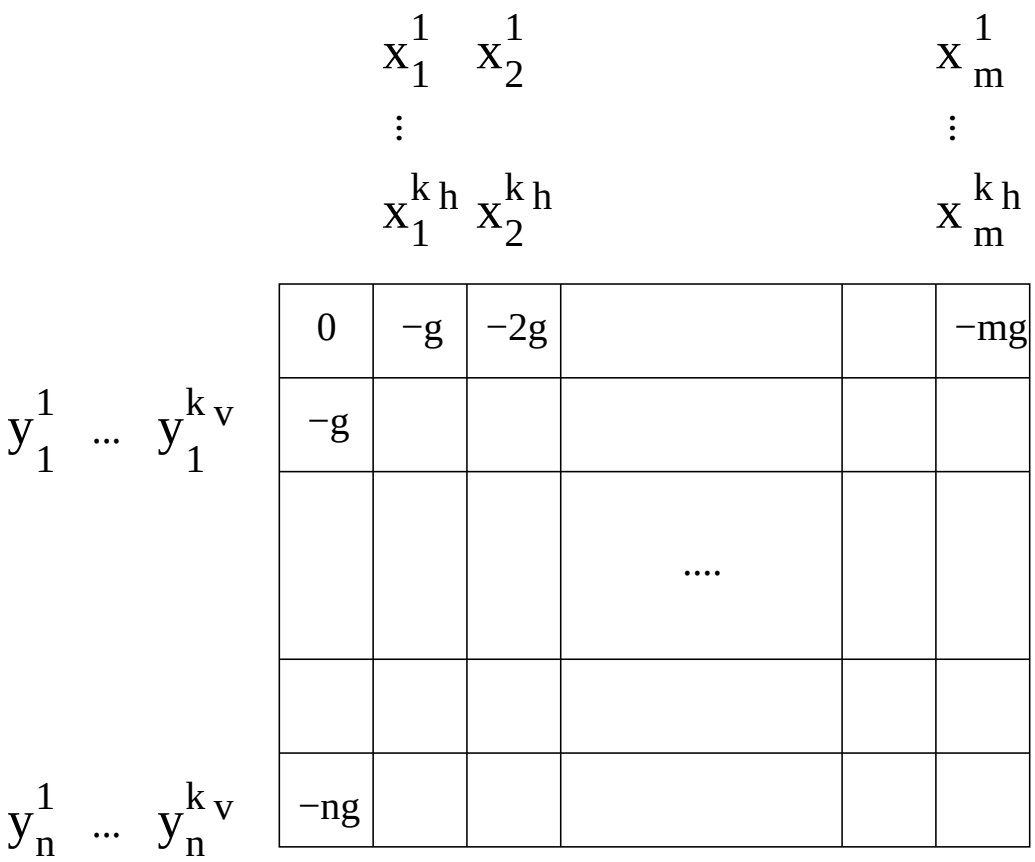
Kececioglu in [KeSt, 2004] used shapes (ordered partition of the $k + l$ sequences that represents the final gap-configuration of a multiple alignment), provided some formulae but still it is difficult to implement.

# Why do we need to align alignments ?




progressive alignment

# The problem



# introducing the $\lambda$ function

## To simplify the expression of formulae

we introduce the function $\lambda$ which is an application $\Sigma^4 \rightarrow \mathbb{R}$, induced by the gap model :

$$\lambda \begin{pmatrix} a^r_{c-1} & a^r_c \\ a^s_{c-1} & a^s_c \end{pmatrix} = \begin{cases} 0 & \text{if} \quad c-1=0 \\ 1 & \text{if} \quad a^r_c \neq - \quad \text{and} \quad a^s_c \neq - \\ g_{op} & \text{if} \quad a^r_c = - \quad \text{or} \quad a^s_c = - \end{cases}$$

for a linear gap model

# $\lambda$ for affine gap model

## For affine gap model

For the affine gap model, the previous edit operation and especially insertions will influence the cost of the penalty.
$\forall c \in \{1, \ldots, q\}$ :

$$\lambda \begin{pmatrix} a^r_{c-1} & a^r_c \\ a^s_{c-1} & a^s_c \end{pmatrix} = \begin{cases} 0 & \text{if } c-1=0 \\ 1 & \text{if } a^r_c \neq - \text{ and } \quad a^s_c \neq - \\ g_{op} & \text{if } (a^r_c = - \text{ and } a^r_{c-1} \neq -) \\ & \text{or if } (a^s_c = - \text{ et } a^s_{c-1} \neq -) \\ g_{ext} & \text{if } (a^r_c = - \text{ and } a^r_{c-1} = -) \\ & \text{or if } (a^s_c = - \text{ et } a^s_{c-1} = -) \end{cases}$$

# Sum of pairs

## expression of sop

$$sop^c(A_c^S) = \sum_{r=1}^{k-1} \sum_{s=r+1}^{k} \delta_{r,s} \times w(a_c^r, a_c^s) \times \lambda \begin{pmatrix} a_{c-1}^r & a_c^r \\ a_{c-1}^s & a_c^s \end{pmatrix}$$

# DP for alignments

Please refer to article in LNCS book

# General form

A formula is composed of 4 factors :

- $\alpha$ : former edit operation $D_{i-1,j-1}$
- $\beta$ : sum-of-pairs score of column $j$ of $A_h$

$$\sum_{r=1}^{k_h-1} \sum_{s=r+1}^{k_h} w(x_j^r, x_j^s) \times \lambda \begin{pmatrix} x_{j-1}^r & x_j^r \\ x_{j-1}^s & x_j^s \end{pmatrix}$$

- $\gamma$ : sum-of-pairs score of the column $i$ of $A_v$

$$\sum_{r=1}^{k_v-1} \sum_{s=r+1}^{k_v} w(y_i^r, y_i^s) \times \lambda \begin{pmatrix} y_{i-1}^r & y_i^r \\ y_{i-1}^s & y_i^s \end{pmatrix}$$

- $\delta$ : interaction of column $j$ of $A_h$ with column $i$ of $A_v$

$$\sum_{r=1}^{k_v-1} \sum_{s=r+1}^{k_h} f$$

# Complexity

## Kececioglu 2004

Given two alignments of $k$ sequences of length $n$, the complexity of [KeSt, 2004] is $\mathcal{O}((3+\sqrt{2})^k(n-k)^2k^{3/2})$.

## Richer, Derrien, Hao 2007

Given two alignments of $k$ sequences of length $n$ the complexity of our algorithm is $\mathcal{O}(n^2k^2)$ (exaclty : $18n^2k^2$).

## comparison

with $k = 10$ and $n = 100$, Kececioglu'algorithm requires $7.2 \times 10^{11}$ computations while we only need $1.8 \times 10^7$.

# Experimentations and results

## Softwares used

- Clustal 1.83
- Mafft 5.86
- Muscle 3.6
- Probcons 1.11
- TCoffee 4.96
- Malinba 1.0 (Multiple Affine or LINear Block Alignment)

## Benchmark

we used BaliBase and reported the SPS (sum-of-pairs) and CS (column score) of the alignments obtained by each software.

# Results

## rank

we can rank the softwares as follows using the average SPS and CS scores :

$$CLUSTAL < MALINBA, T - COFFEE < MUSCLE < MAFFT < PROBCONS$$

# Conclusion

- aligning alignment expressed within Dynamic Programming framework
- easy to implement
- first results are good but require more efforts to get an improved quality (post-optimization treatments, secondary structure information)

# Bibliography

Needleman S.B., Wunsch C.D..
*A general method applicable to the search for similarities in the amino acid sequence of two proteins*.
Journal of Molecular Biology, 3(48):334–453, 1970.

John Kececioglu and Dean Starrett
*Aligning alignments exactly*
Proceedings of RECOMB 04, March 27-31, San Diego, CA, USA, 2004