CSI5126. Algorithms in bioinformatics

Multiple Sequence Alignment (MSA)

Marcel Turcotte

uOttawa

School of Electrical Engineering and Computer Science (EECS)
University of Ottawa

Version October 4, 2018

Summary

In this lecture, we consider the **generalization of the pairwise alignment problem** to multiple sequences. Although an exact formulation for the problem is easy to derive, the time/space complexity of the resulting algorithm makes it impractical. Next, we consider some **practical algorithms**. Finally, we mention the drawbacks of the **sum of pairs score**.

General objective

- **Explain in your own words** the progressive multiple sequence alignment, with sufficient details so that an actual implementation can be made.

Reading

- Bernhard Haubold and Thomas Wiehe (2006). *Introduction to computational biology: an evolutionary approach*. Birkhäuser Basel. Pages 91-100.

Reviews

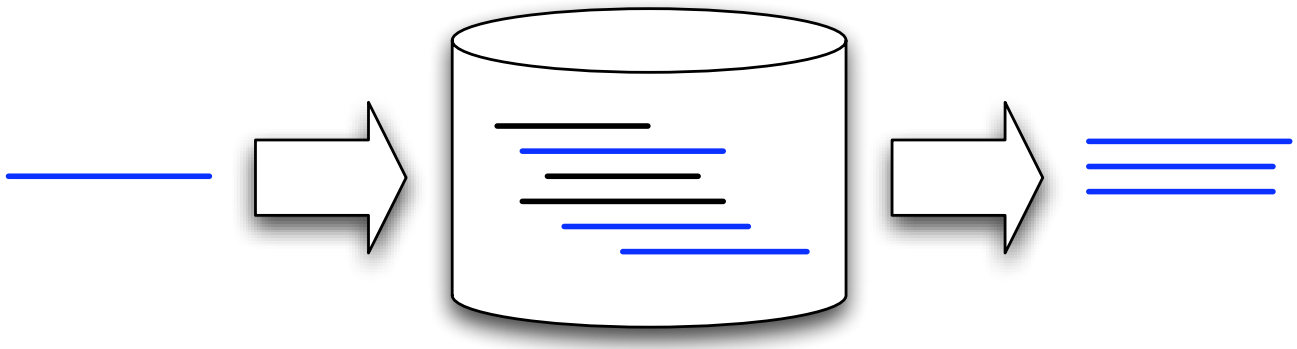
- Chowdhury, B., & Garai, G. (2017). A review on multiple sequence alignment from the perspective of genetic algorithm. *Genomics*, 109(5-6), 419–431. http://doi.org/10.1016/j.ygeno.2017.06.007
- Chatzou, M., Magis, C., Chang, J.-M., Kemena, C., Bussotti, G., Erb, I., & Notredame, C. (2016). Multiple sequence alignment modeling: methods and applications. *Briefings in Bioinformatics*, 17(6), 1009–1023. http://doi.org/10.1093/bib/bbv099
- Julie D. Thompson , Benjamin Linard, Odile Lecompte, Olivier Poch A Comprehensive Benchmark Study of Multiple Sequence Alignment Methods: Current Challenges and Future Perspectives *PLOS One*, March 31, (2011) http://dx.doi.org/10.1371/journal.pone.0018093

Reviews (continued)

- C. Kemena and C. Notredame, Upcoming challenges for multiple sequence alignment methods in the high-throughput era, *Bioinformatics*, vol. 25, no. 19, pp. 2455–2465, Sep. 2009.
- J. Pei, Multiple protein sequence alignment, *Curr Opin Struct Biol*, vol. 18, no. 3, pp. 382–386, Jun. 2008.
- C. Notredame. Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput Biol*, 3(8):e123, August 2007.
- R. C. Edgar and S. Batzoglou. Multiple sequence alignment. *Curr Opin Struct Biol*, 16(3):368–373, 2006.

Motivation: Database search problem

Find all the sequences that are similar to the **given input sequence** (statistically significant match).



Motivation

Given the following **input sequence**:

```
>d1c5fe_ 2.58.1.1.4 Cyclophilin (eukaryotic) Nematode
kdrrrvfldvtidgnlagrivmelyndiaprtcnnflmlctgmagtgkisgkplhykgst
fhrviknfmiqggdftkgdgtggesiyggmfddeefvmkhdepfvvsmankgpntngsqf
fitttpaphlnnihvvfgkvvsgqevvtkieylktnsknrpladvvilncgelv
```

We can use a pairwise sequence comparison algorithm, such as FASTA, to find **homologues**:

```
> fasta -Q -H -E 0.0001 d1c5fe_.fa astral-scopdom-atom-all-1.50.fa

 FASTA searches a protein or DNA sequence data bank
 version 3.3t06 Aug. 3, 2000
Please cite:
 W.R. Pearson & D.J. Lipman PNAS (1988) 85:2444-2448

d1c5fe_.fa: 174 aa
 >d1c5fe_ 2.58.1.1.4 Cyclophilin (eukaryotic) Nematode (Brugia malayi)
 vs  /bio/data/scopseq-1.50/astral-scopdom-atom-all-1.50.fa library
searching /bio/data/scopseq-1.50/astral-scopdom-atom-all-1.50.fa library

4231245 residues in 23790 sequences
 Expectation_n fit: rho(ln(x))= 6.7522+/-0.000373; mu= -3.1184+/- 0.019
 mean_var=71.7735+/-16.871, 0's: 0 Z-trim: 84  B-trim: 0 in 0/39
 Lambda= 0.1514
```

⇒ 9 (statistically) significant matches were found:

```
FASTA (3.36 June 2000) function [optimized, BL50 matrix (15:-5)] ktup: 2
 join: 36, opt: 24, gap-pen: -12/ -2, width:  16
 Scan time:  5.590
The best scores are:                                      opt bits E(23706)
d1c5fg_ 1.4 Cyclophilin (eukaryotic) {Nema  ( 172) 1155  261 1.8e-70
d1cyna_ 1.2 Cyclophilin (eukaryotic) {Huma  ( 178)  588  137 3.6e-33
d2rmce_ 1.3 Cyclophilin (eukaryotic) {Mous  ( 182)  555  130 5.5e-31
d1ak4b_ 1.1 Cyclophilin (eukaryotic) {Huma  ( 163)  532  125 1.6e-29
d1rmha_ 1.1 Cyclophilin (eukaryotic) {Huma  ( 164)  532  125 1.6e-29
d2rmbi_ 1.1 Cyclophilin (eukaryotic) {Huma  ( 165)  532  125 1.6e-29
d2rmbc_ 1.1 Cyclophilin (eukaryotic) {Huma  ( 165)  532  125 1.6e-29
d1awtf_ 1.1 Cyclophilin (eukaryotic) {Huma  ( 160)  375   91 3.3e-19
d1clh__ 1.5 Bacterial cyclophilin {Escheri  ( 166)  166   45 1.9e-05
```

```
>>d1c5fg_ 2.58.1.1.4 Cyclophilin (eukaryotic) {Nematode   (172 aa)
 initn: 1155 init1: 1155 opt: 1155  Z-score: 1376.0  bits: 261.1 E(): 1.8e-7
Smith-Waterman score: 1155;  100.000% identity (100.000% ungapped) in 172 a

               10        20        30        40        50        60
d1c5fe KDRRRVFLDVTIDGNLAGRIVMELYNDIAPRTCNNFLMLCTGMAGTGKISGKPLHYKGST
       ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
d1c5fg  RRRVFLDVTIDGNLAGRIVMELYNDIAPRTCNNFLMLCTGMAGTGKISGKPLHYKGST
                 10        20        30        40        50

               70        80        90       100       110       120
d1c5fe FHRVIKNFMIQGGDFTKGDGTGGESIYGGMFDDEEFVMKHDEPFVVSMANKGPNTNGSQF
     ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
d1c5fg FHRVIKNFMIQGGDFTKGDGTGGESIYGGMFDDEEFVMKHDEPFVVSMANKGPNTNGSQF
       60        70        80        90       100       110

              130       140       150       160       170
d1c5fe FITTTPAPHLNNIHVVFGKVVSGQEVVTKIEYLKTNSKNRPLADVVILNCGELV
       ::::::::::::::::::::::::::::::::::::::::::::::::::::::
d1c5fg FITTTPAPHLNNIHVVFGKVVSGQEVVTKIEYLKTNSKNRPLADVVILNCGELV
      120       130       140       150       160       170
```

```
>>d1cyna_ 2.58.1.1.2 Cyclophilin (eukaryotic) {Human (Ho  (178 aa)
 initn: 569 init1: 505 opt: 588  Z-score: 706.4  bits: 137.2 E(): 3.6e-33
Smith-Waterman score: 588;  53.254% identity (55.901% ungapped) in 169 aa o

                    10         20         30         40         50
d1c5fe  KDRRRVFLDVTIDGNLAGRIVMELYNDIAPRTCNNFLMLCTGMAGTGKISGKPLHYKG
               .:..:. :   . .:::... :..  .:.: .::. : ::   : :         ::.
d1cyna GPKVTVKVYFDLRIGDEDVGRVIFGLFGKTVPKTVDNFVALATGEKGFG--------YKN
                  10         20         30         40                  50

          60         70         80         90         100        110
d1c5fe STFHRVIKNFMIQGGDFTKGDGTGGESIYGGMFDDEEFVMKHDEPFVVSMANKGPNTNGS
       : ::::::.:::::::::::::::::::  : ::.: .::  :  ::::: : .::::
d1cyna SKFHRVIKDFMIQGGDFTRGDGTGGKSIYGERFPDENFKLKHYGPGWVSMANAGKDTNGS
               60         70         80         90         100        110

       120        130        140        150        160        170
d1c5fe QFFITTTPAPHLNNIHVVFGKVVSGQEVVTKIEYLKTNSKNRPLADVVILNCGELV
        ::::::. .  :.. :::::::. :.::: :.:  ::.:..:: ::.: .::..
d1cyna QFFITTVKTAWLDGKHVVFGKVLEGMEVVRKVESTKTDSRDKPLKDVIIADCGKIEVEKP
               120        130        140        150        160        170

d1cyna FAIAKE
```

⇒ ...

```
>>d1clh__ 2.58.1.1.5 Bacterial cyclophilin {Escherichia   (166 aa)
 initn: 178 init1:  94 opt: 166  Z-score: 208.9  bits: 45.0 E(): 1.9e-05
Smith-Waterman score: 183;  29.630% identity (34.286% ungapped) in 162 aa c

                10        20        30        40        50        60
d1c5fe KDRRRVFLDVTIDGNLAGRIVMELYNDIAPRTCNNFLMLCTGMAGTGKISGKPLHYKGST
                        :: : .:: .. :: . .::.     ....:         :...:
d1clh_    AKGDPHVLLTTSAGNIELELDKQKAPVSVQNFV----DYVNSG-------FYNNTT
                    10        20        30                  40

                70        80        90       100       110       120
d1c5fe FHRVIKNFMIQGGDFTKGDGTGGESIYGGMFDDEEFVMKHDEPFVVSMANKGPNTNGSQF
       ::::: .:::::: ::  .    ..    . .. .  ... .  ..   .   ..  :::
d1clh_ FHRVIPGFMIQGGGFT--EQMQQKKPNPPIKNEADNGLRNTRGTIAMARTADKDSATSQF
           50        60          70        80        90       100

               130           140       150           160       170
d1c5fe FITTTPAPHLNNI-----HVVFGKVVSGQEVVTKIEYLKTNS----KNRPLADVVILNCG
       ::...    :..      ..:::::::.:.:. ::  . :..    .: :   ::::.
d1clh_ FINVADNAFLDHGQRDFGYAVFGKVVKGMDVADKISQVPTHDVGPYQNVPSKPVVILSAK
           110       120       130       140       150       160


d1c5fe ELV
```

Motivation

Now **what**?

Motivation

Now **what**?

```
d2rmbi_   KGSCFHRIIPGFMCQGGDFTRHNGTGGKSIYGEKFEDENFILKHTGPGILSMANAGPNTN
d2rmbc_   KGSCFHRIIPGFMCQGGDFTRHNGTGGKSIYGEKFEDENFILKHTGPGILSMANAGPNTN
d1ak4b_   KGSCFHRIIPGFMCQGGDFTRHNGTGGKSIYGEKFEDENFILKHTGPGILSMANAGPNTN
d1rmha_   KGSCFHRIIPGFMCQGGDFTRHNGTGGKSIYGEKFEDENFILKHTGPGILSMANAGPNTN
d1awtf_   KGSCFHRIIPGFXCQGGDFTRHNGTGGKSIYGEKFEDENFILKHTGPGILSXANAGPNTN
d1cyna_   KNSKFHRVIKDFMIQGGDFTRGDGTGGKSIYGERFPDENFKLKHYGPGWVSMANAGKDTN
d2rmce_   KGSIFHRVIKDFMIQGGDFTARDGTGGMSIYGETFPDENFKLKHYGIGWVSMANAGPDTN
d1c5fg_   KGSTFHRVIKNFMIQGGDFTKGDGTGGESIYGGMFDDEEFVMKHDEPFVVSMANKGPNTN
d1clh__   NNTTFHRVIPGFMIQGGGFTEQMQQ--KKPNPPIKNEADNGLRNTRGTIAMARTADKDSA
d1efca1   ------AIDKPFLLPIEDVFSISGRG--TVVTGRVERGIIKVGEEVEIVGIKETQKSTCT
                     :   *     ..       .  .              : .          .
...
```

Computer Science's Point of View: Generalization

- An MSA (**multiple sequence alignment**) is a generalization of the pairwise sequence alignment.
- **Definition.** Given $k > 2$ strings $S = \{S_1, S_2, \ldots, S_k\}$, gaps are inserted so that 1) all the sequences have the **same length** and 2) the **distance** for the alignment is **minimized** (this can also be seen as to maximize the similarity).
- **Global** or **local** multiple alignment.

Life Science's Point of View

Conserved patterns, e.g. conserved cysteins forming **disulphide bonds**.

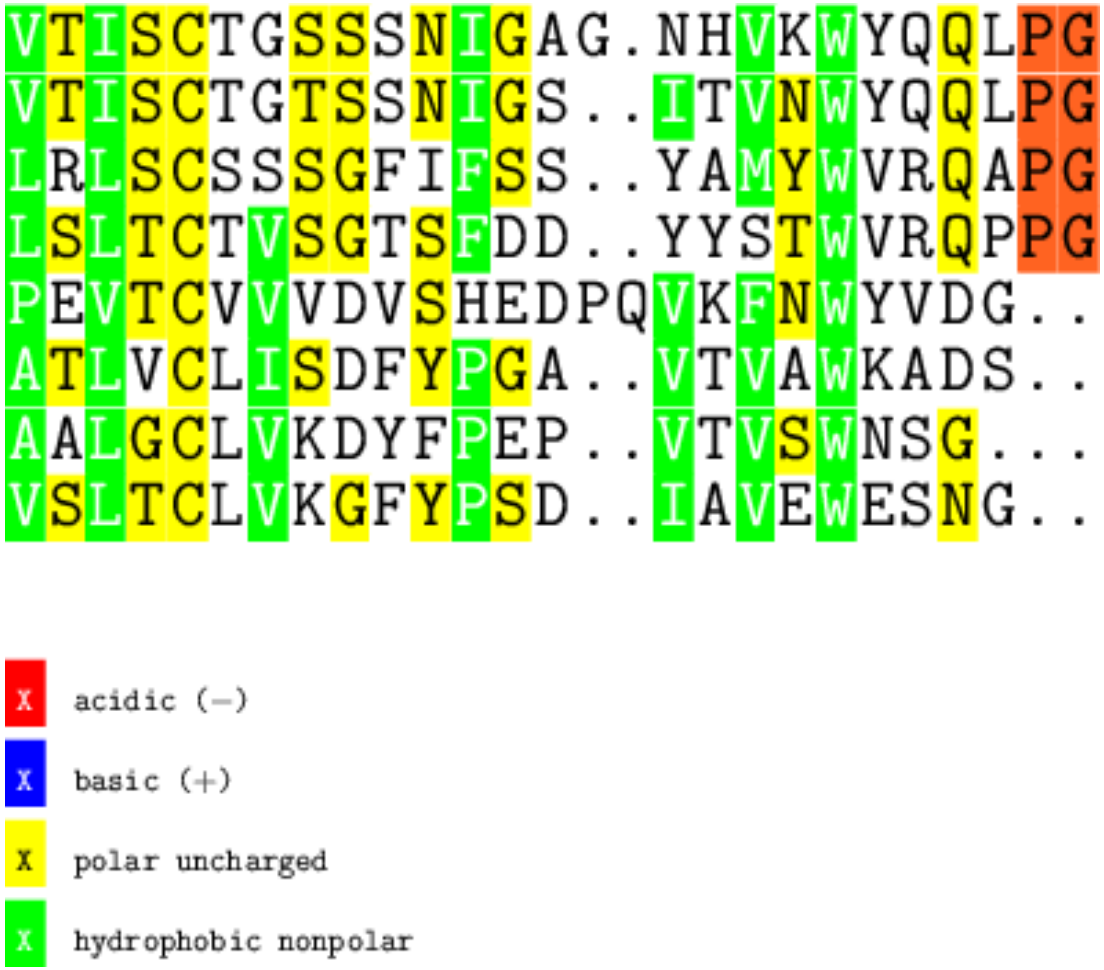
Life Science's Point of View

Conserved **Pro** and **Gl**y opposed to an insersion suggest the presence of a **loop**.

Life Science's Point of View

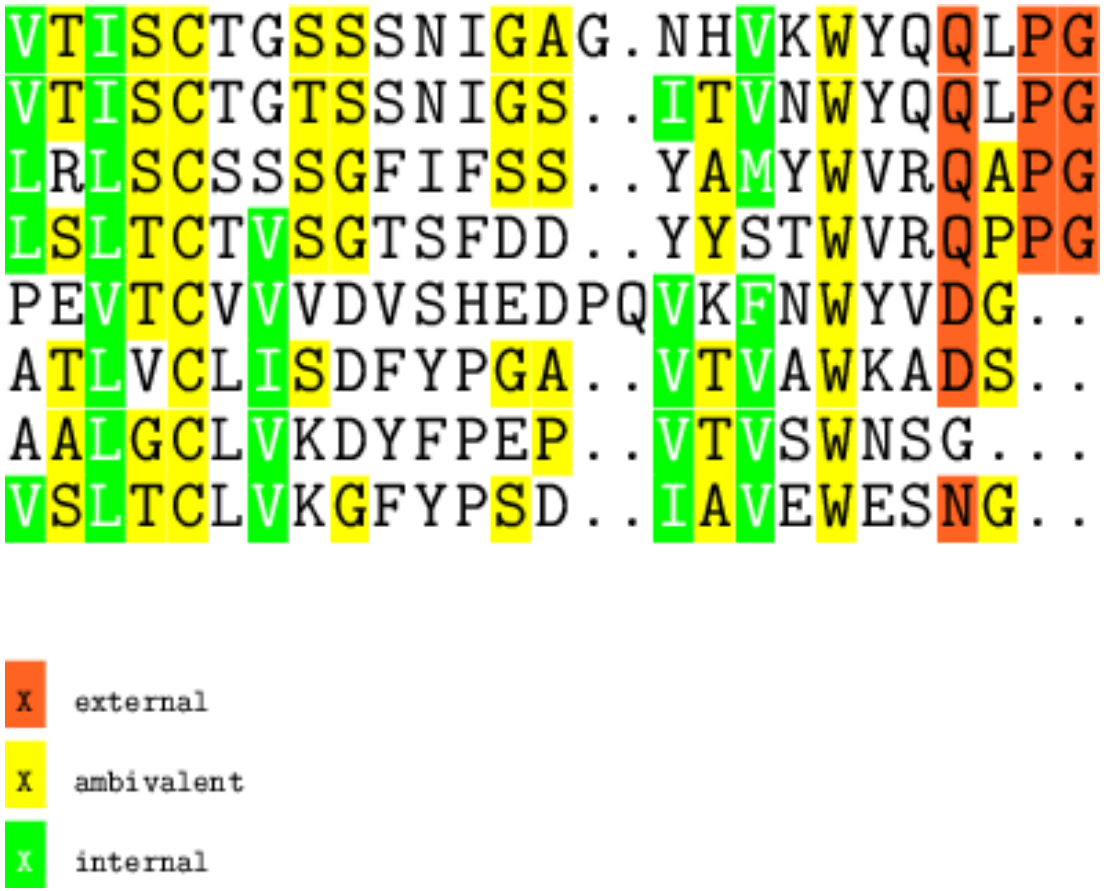
Similarity.

Life Science's Point of View



Chemical properties.

Life Science's Point of View



Patterns of conservation/substitution can indicate a preference for **solvent exposure**.

Life Science's Point of View

Secondary structure elements?

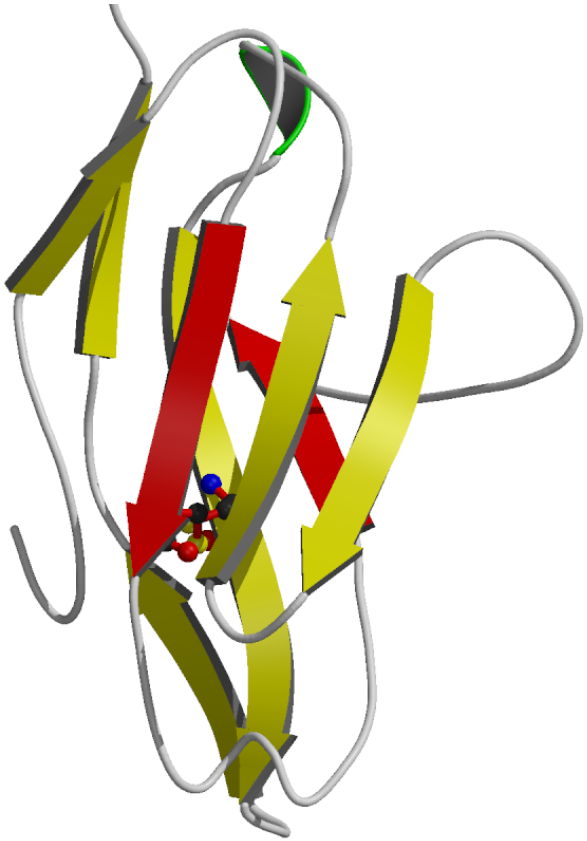
Life Science's Point of View

Gaps are good indicators of **loop** regions.

Life Science's Point of View

History (phylogeny)

Life Science's Point of View



Life Science's Point of View

"**Multiple alignments are among the most useful objects in bioinformatics**" [Wallace 2005]

- Phylogenetic trees inference
- Identifying functional residues
- Structure prediction
- etc.

Pairwise vs **Multiple Sequence** Alignment

- **Pairwise**: the question is "**are the two sequences related?**"

Pairwise vs **Multiple Sequence** Alignment

- **Pairwise**: the question is "**are the two sequences related?**"
- **Multiple**: the sequences are **assumed to be related** from the start.

Multiple Sequence Alignment

Rectangular table such that:

- Rows are (related, homologous) sequences

All three criteria might not be simultaneously met, especially for sequences that are not closely related.

Multiple Sequence Alignment

Rectangular table such that:

- Rows are (related, homologous) sequences
- Residues in a given column (site):

All three criteria might not be simultaneously met, especially for sequences that are not closely related.

Multiple Sequence Alignment

Rectangular table such that:

- Rows are (related, homologous) sequences
- Residues in a given column (site):
  1. **Evolved** from a position in some ancestral sequence (homologous)

All three criteria might not be simultaneously met, especially for sequences that are not closely related.

Multiple Sequence Alignment

Rectangular table such that:

- Rows are (related, homologous) sequences
- Residues in a given column (site):
  1. **Evolved** from a position in some ancestral sequence (homologous)
  2. Can be **superimposed** in three-dimension in a structural alignment

All three criteria might not be simultaneously met, especially for sequences that are not closely related.

Multiple Sequence Alignment

Rectangular table such that:

- Rows are (related, homologous) sequences
- Residues in a given column (site):
  1. **Evolved** from a position in some ancestral sequence (homologous)
  2. Can be **superimposed** in three-dimension in a structural alignment
  3. Have the same **functional** role

All three criteria might not be simultaneously met, especially for sequences that are not closely related.

Objective function: sum-of-pairs

Given a **multiple alignment** $\mathcal{M}$ of k sequences and n columns.

$$\mathrm{sp}(\mathcal{M}) = \sum_{c=1}^{n} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} s(m_{ci}, m_{cj})$$

where $s(a, b)$ is a substitution matrix such as PAM250 or BLOSUM62.

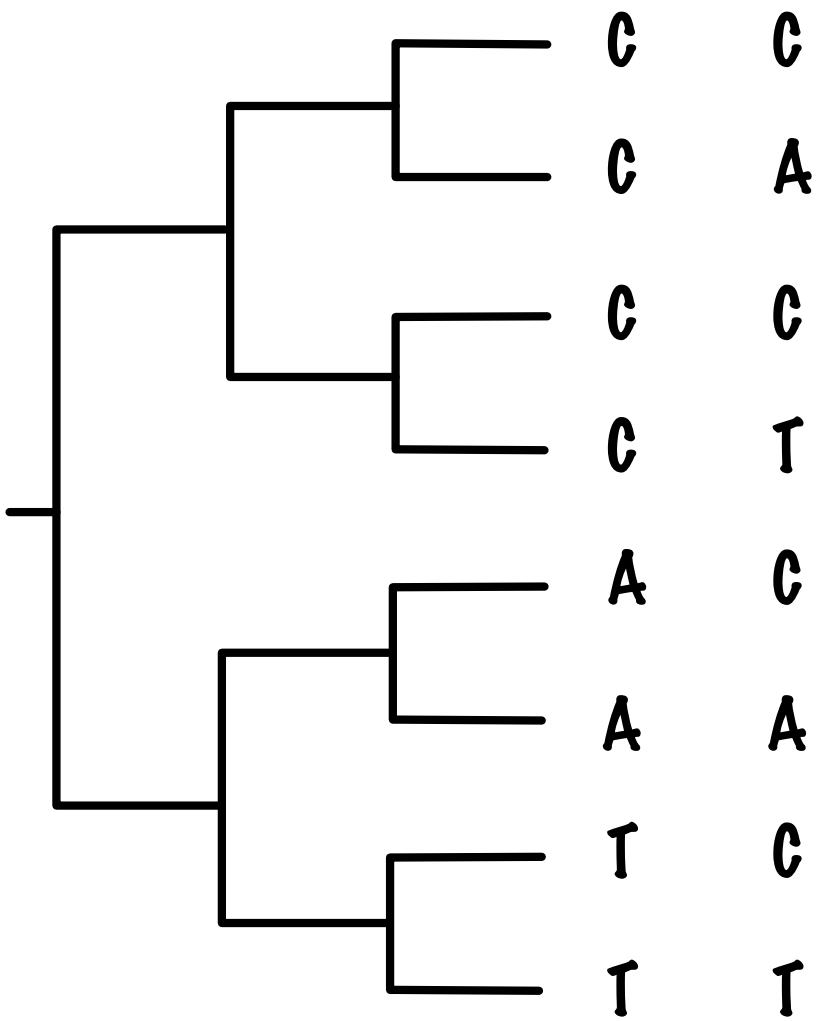
```
                    1   c           n
1     HBA_HUMAN     ...VGA--HAGEY...
      HBB_HUMAN     ...V----NVDEV...
i->   MYG_PHYCA     ...VEA--DVAGH...
 j->  GLB2HCHITP    ...VKG------D...
      LGB2LUPLU     ...FNA--NIPKH...
k     GLB1GLYDI     ...IAGADNGAGV...
```

Problem: Compute the (global) alignment that maximizes the sum-of-pairs (SP) score.

Remarks

- Unlike pairwise aligment, the sum-of-pairs score used by the MSA methods has **no theoretical fundation**, no interpretation in terms of an underlying evolutionary model.

Sum of pairs



Remarks

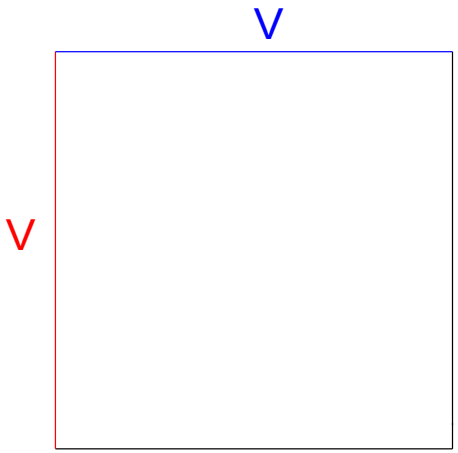
I. M. Wallace, G. Blackshields, and D. G. Higgins. Multiple sequence alignments. *Curr Opin Struct Biol*, 15(3):261–6, Jun 2005.

- "Assembling a suitable MSA is not, however, a trivial task, and **none of the existing methods have yet managed to deliver biologically perfect MSAs.**"
- "**Manually refined alignments** continue to be **superior** to purely automated methods;"
- "The **wealth of available methods** and their increasingly **similar accuracies** makes it harder than ever to objectively choose one over the others."

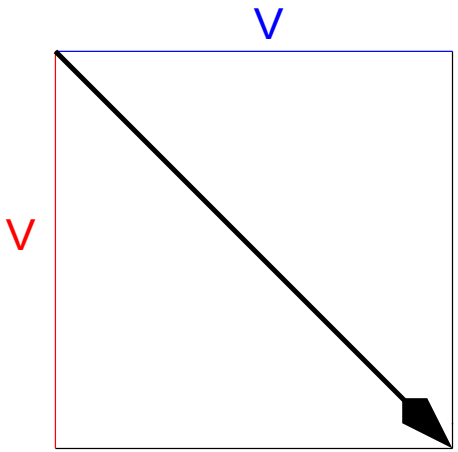
2, 3, k, go!

- Optimal alignment of **2 sequences**
- Optimal alignment of **3 sequences**
- Optimal alignment of k **sequences**

Optimal alignment of **2 Sequences**



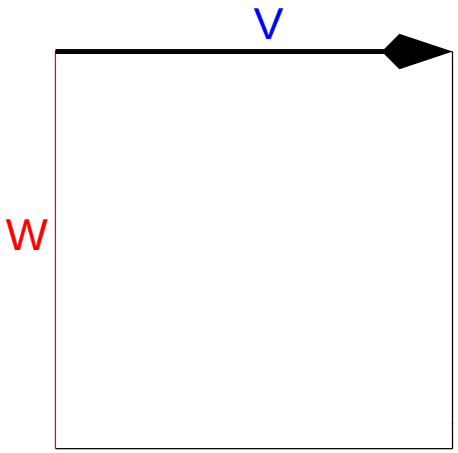
Optimal alignment of **2 Sequences**



V

V

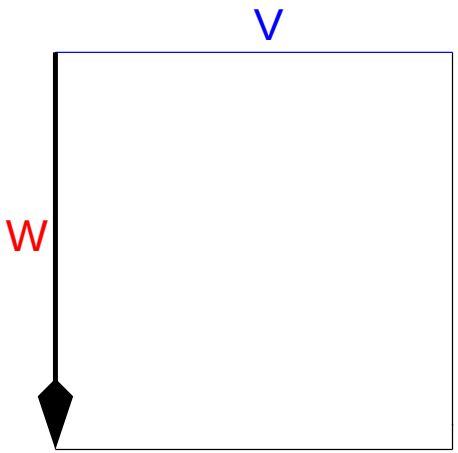
Optimal alignment of **2 Sequences**



V

–

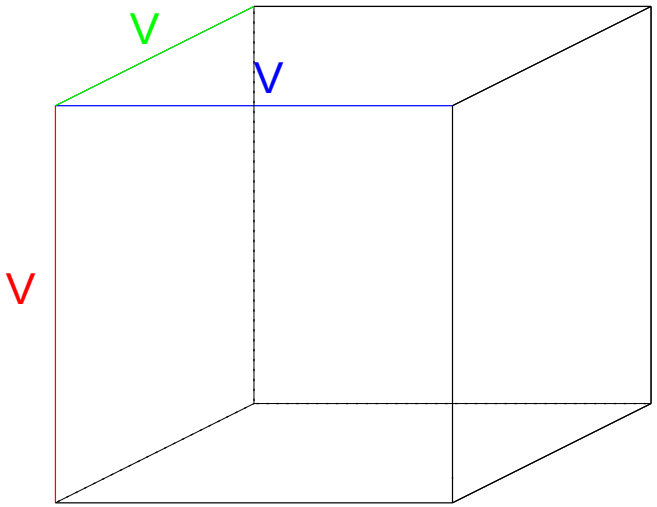
Optimal alignment of **2 Sequences**



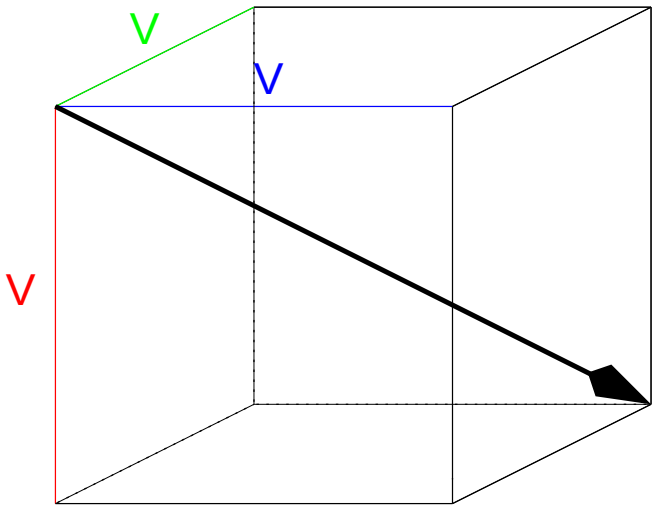
–

W

Optimal alignment of **3 Sequences**



Optimal alignment of **3 Sequences**

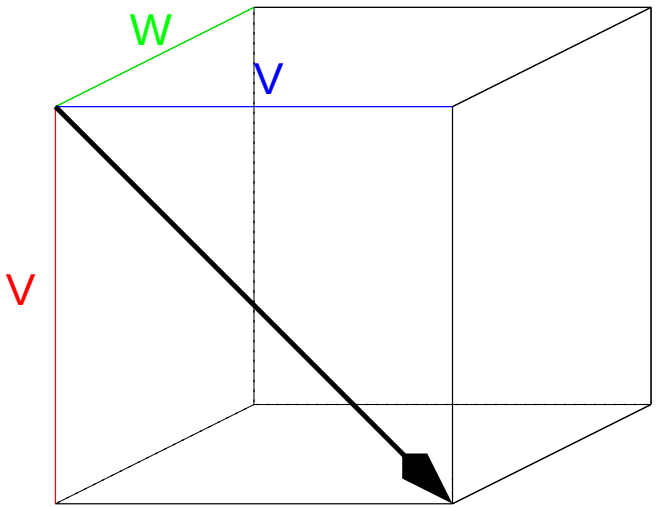


V

V

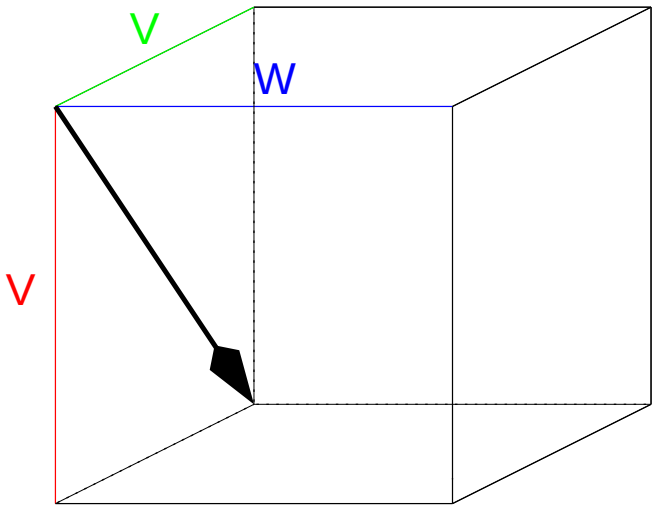
V

Optimal alignment of **3 Sequences**



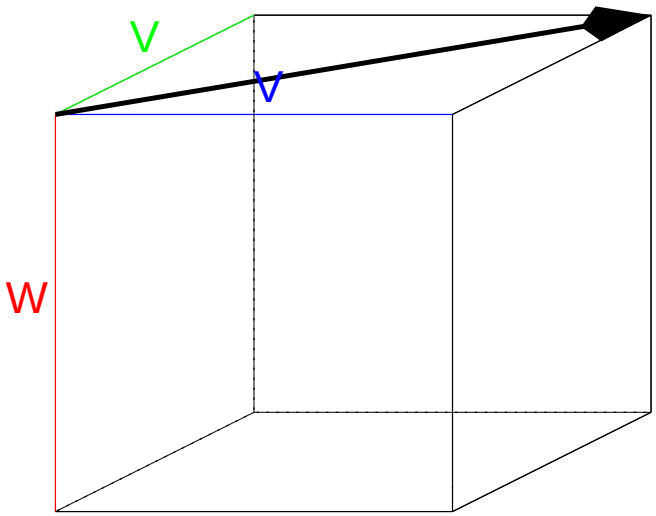

```
V
V
-
```

Optimal alignment of **3 Sequences**



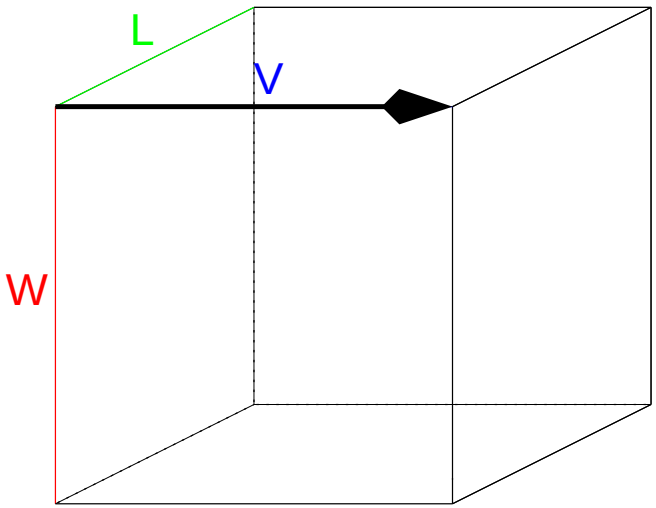

```
-
V
V
```

Optimal alignment of **3 Sequences**




```
V
-
V
```

Optimal alignment of **3 Sequences**

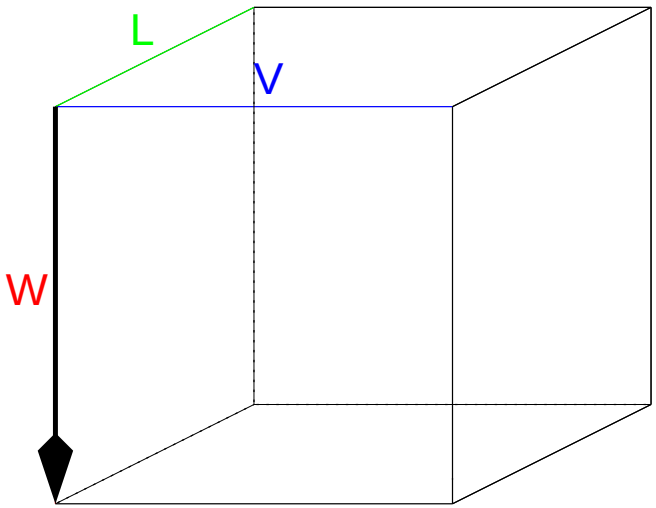


V

\-

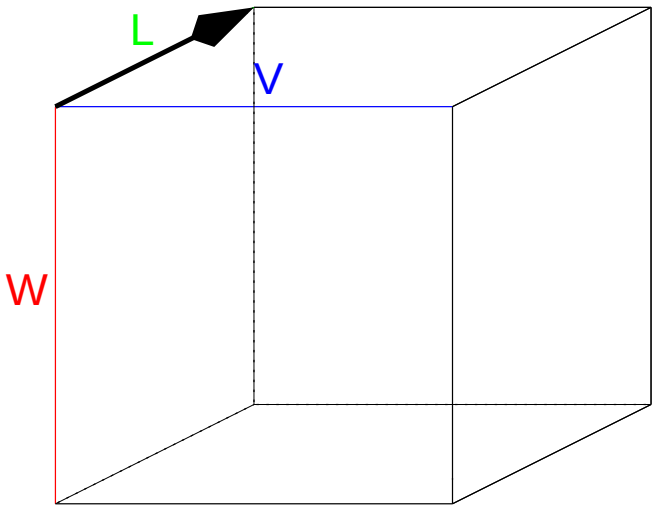
\-

Optimal alignment of **3 Sequences**



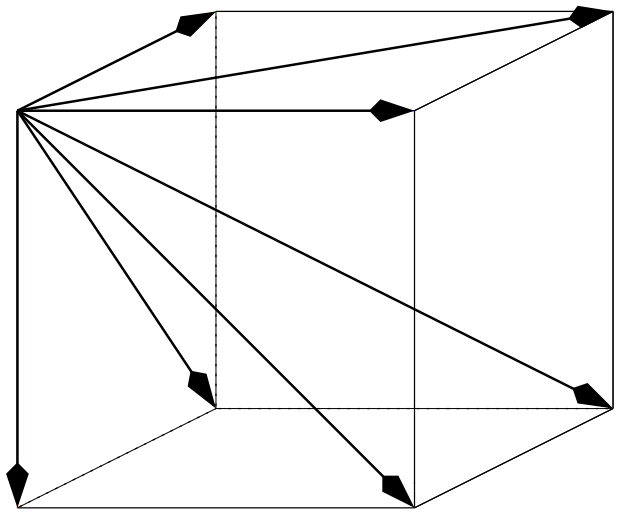

```
-
W
-
```

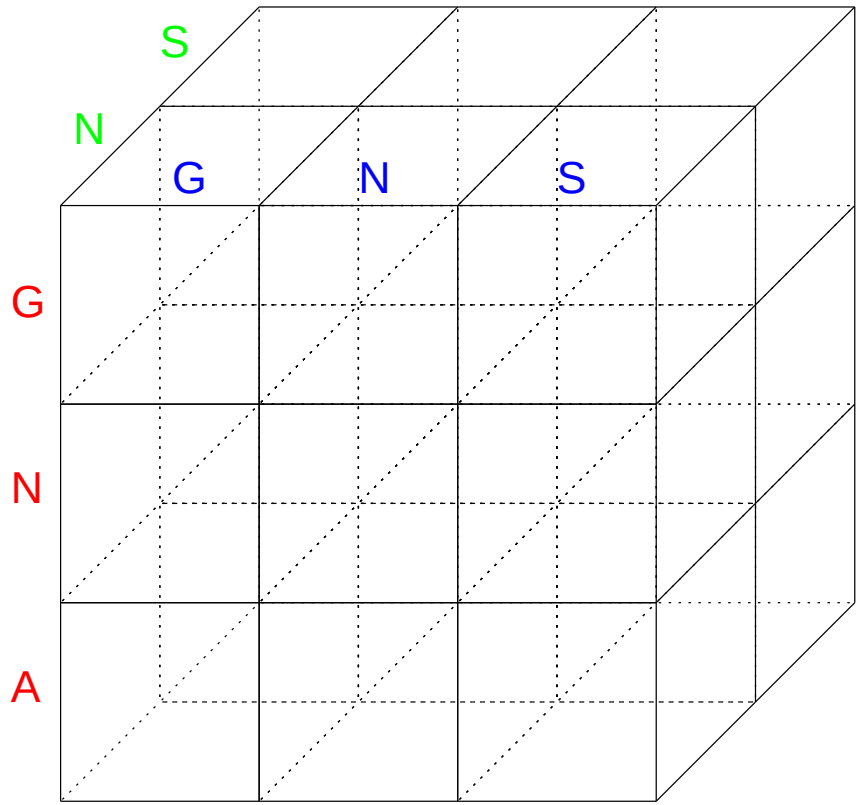
Optimal alignment of **3 Sequences**

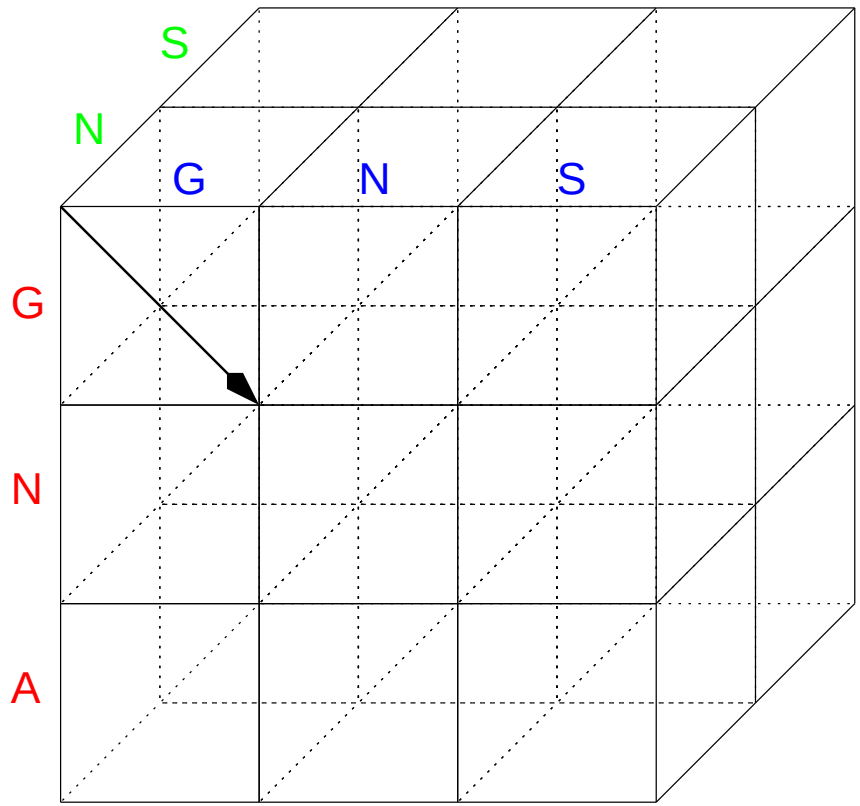



```
-
-
L
```

Optimal alignment of **3 Sequences**



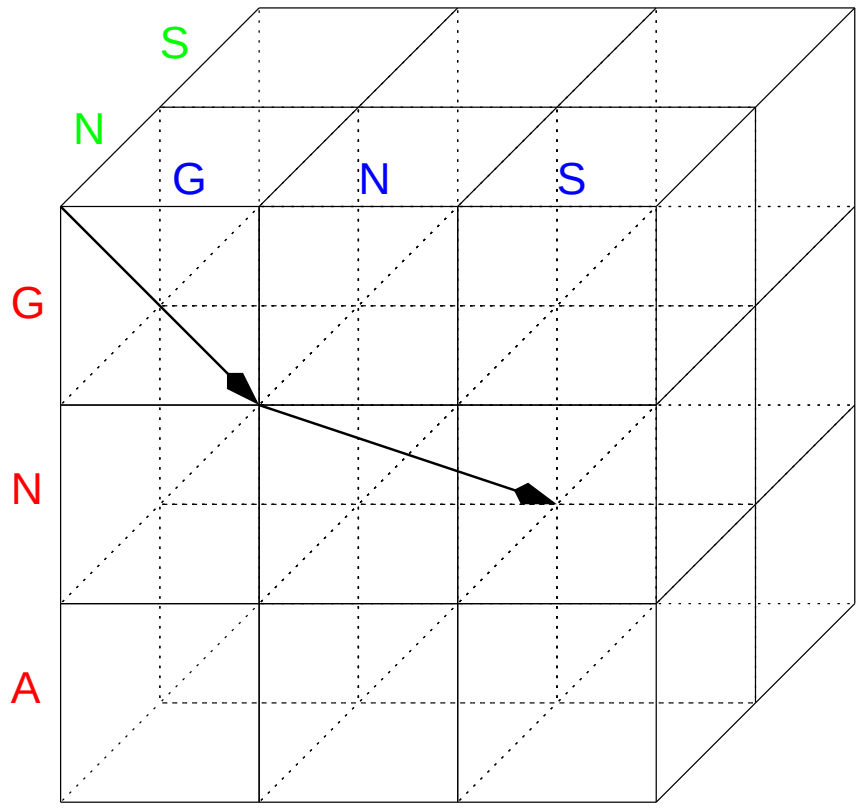

S
N
G
N
S
G
N
A



G

G

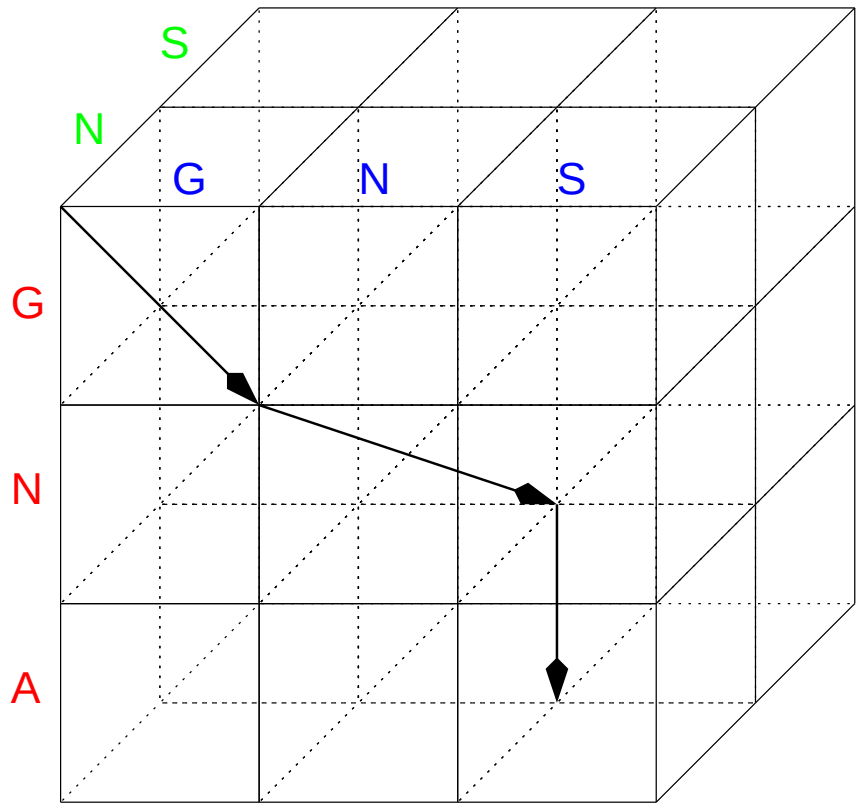
–



GN

GN

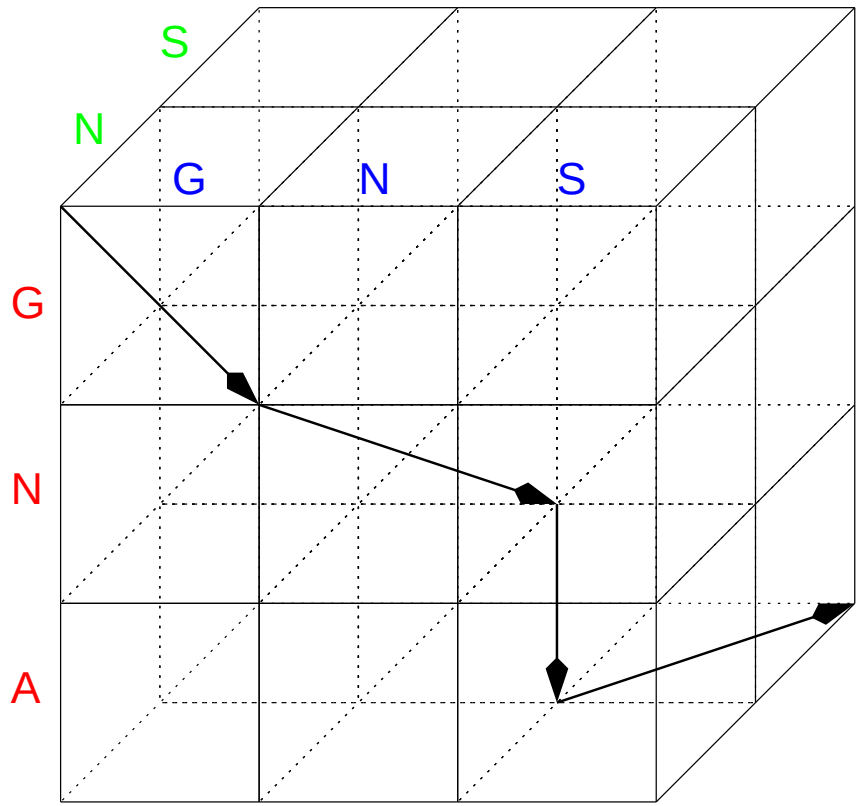
-N



GN-

GNA

-N-



GN-S

GNA-

-N-S

Exact alignment of **3 Sequences**

Given: x, y and z three strings.

$V(i,j,k)$ is the optimal SP score to align $x[1..i]$, $y[1..j]$ and $z[1..k]$ is given by:

$$V(i,j,k) = \max \begin{cases} V(i-1,j-1,k-1) + s(x_i, y_j, z_k), \\ V(i-1,j-1,k) - s(x_i, y_j, -), \\ V(i,j-1,k-1) - s(-, y_j, z_k), \\ V(i-1,j,k-1) - s(x_i, -, z_k), \\ V(i-1,j,k) - s(x_i, -, -), \\ V(i,j-1,k) - s(-, y_j, -), \\ V(i,j,k-1) - s(-, -, z_k). \end{cases}$$

$\Rightarrow$ For non-boundary cells only.

Exact alignment of **3 Sequences**

At the boundaries:

$$\begin{aligned} V(0,0,0) &= 0, \\ V(i,j,0) &= V(x_i, y_j) - (i+j) \times d, \\ V(i,0,k) &= V(x_i, z_k) - (i+k) \times d, \\ V(0,j,k) &= V(y_j, z_k) - (j+k) \times d. \end{aligned}$$

Exact alignment of k Sequences

Given: x^1, x^2 and x^k, k sequences.

The optimum SP alignment for k sequences, $V(i_1, i_2, \ldots, i_k)$, to align $x^1[1..i_1], x^2[1..i_2], \ldots, x^k[1..i_k]$

$$V(i_1, i_2, \ldots, i_k) = \max \begin{cases} V(i_1 - 1, i_2 - 1, \ldots, i_k - 1) + s(i_1, i_2, \ldots, i_k), \\ V(i_1, i_2 - 1, \ldots, i_k - 1) + s(-, i_2, \ldots, i_k), \\ V(i_1 - 1, i_2, \ldots, i_k - 1) + s(i_1, -, \ldots, i_k), \\ \ldots \\ V(i_1 - 1, i_2 - 1, \ldots, i_k) + s(i_1, i_2, \ldots, -), \\ \ldots \\ V(i_1, i_2, \ldots, i_k - 1) + s(-, -, \ldots, i_k), \\ \ldots \end{cases}$$

$\Rightarrow$ All the subsets (2^k) except the empty one, which corresponds to $-, -, \ldots, -$, hence, $2^k - 1$ cases.

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^{9}

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$
  - Time complexity is

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$
  - Time complexity is

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$
  - Time complexity is $O(2^k n^k)$ Say $k = 2$ (pairwise) and $n = 100$ takes 0.2 millisecond
    - For $k = 5$, $n = 100$ takes $\sim$ 26 hours

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$
  - Time complexity is $O(2^k n^k)$ Say $k = 2$ (pairwise) and $n = 100$ takes 0.2 millisecond
    - For $k = 5$, $n = 100$ takes $\sim$ 26 hours

Remarks

- Recall that peta- (P) $= 10^{15}$, tera- (T) 10^{12}, giga- (G) 10^9
- **Given:** k sequences of approximately the same length, n
  - Space complexity is $O(n^k)$ (memory cells)
    - For $n = 100$ and $k = 5$, $n^k = 10^{10}$
    - For $n = 100$ and $k = 10$, $n^k = 10^{20}$
  - Time complexity is $O(2^k n^k)$ Say $k = 2$ (pairwise) and $n = 100$ takes 0.2 millisecond
    - For $k = 5$, $n = 100$ takes $\sim$ 26 hours
    - For $k = 10$, $n = 100$ takes $\sim$ 16 million years!

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming;

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?
  - Genetic algorithms
    SAGA (Notredame and Higgins 1996)

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?
  - Genetic algorithms
    SAGA (Notredame and Higgins 1996)
  - Branch-and-bound
    MSA (Gupta et al 1995), DCA (Stoye et al 1997)

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?
  - Genetic algorithms
    SAGA (Notredame and Higgins 1996)
  - Branch-and-bound
    MSA (Gupta et al 1995), DCA (Stoye et al 1997)
- **Solve a simpler problem:**

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?
  - Genetic algorithms
    SAGA (Notredame and Higgins 1996)
  - Branch-and-bound
    MSA (Gupta et al 1995), DCA (Stoye et al 1997)
- **Solve a simpler problem:**
  - Progressive sequence alignment problem;

What's **next**?

The **exact algorithm** cannot be applied; **prohibitive** space and time complexity.

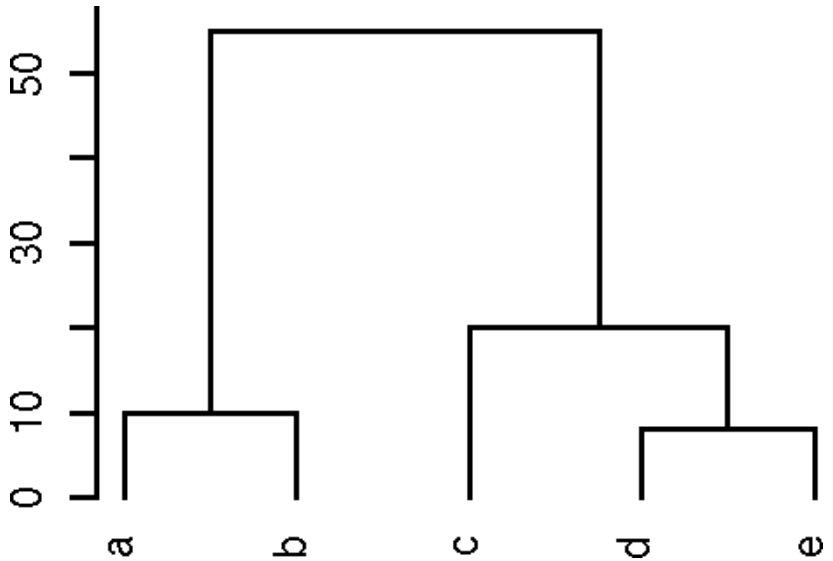
What can be done?

- **Use a different optimization technique**, something else than dynamic programming; Suggestions?
  - Genetic algorithms
    SAGA (Notredame and Higgins 1996)
  - Branch-and-bound
    MSA (Gupta et al 1995), DCA (Stoye et al 1997)
- **Solve a simpler problem:**
  - Progressive sequence alignment problem; most widely used approach.

Progressive alignment methods

Idea.

1. **Two sequences** are chosen and aligned by standard **dynamic programming** algorithm
2. A **third sequence** is chosen and aligned to the first alignment
3. **Iterate** until all sequences have been aligned



$\Rightarrow$ Most commonly used approach.

Progressive Aligments

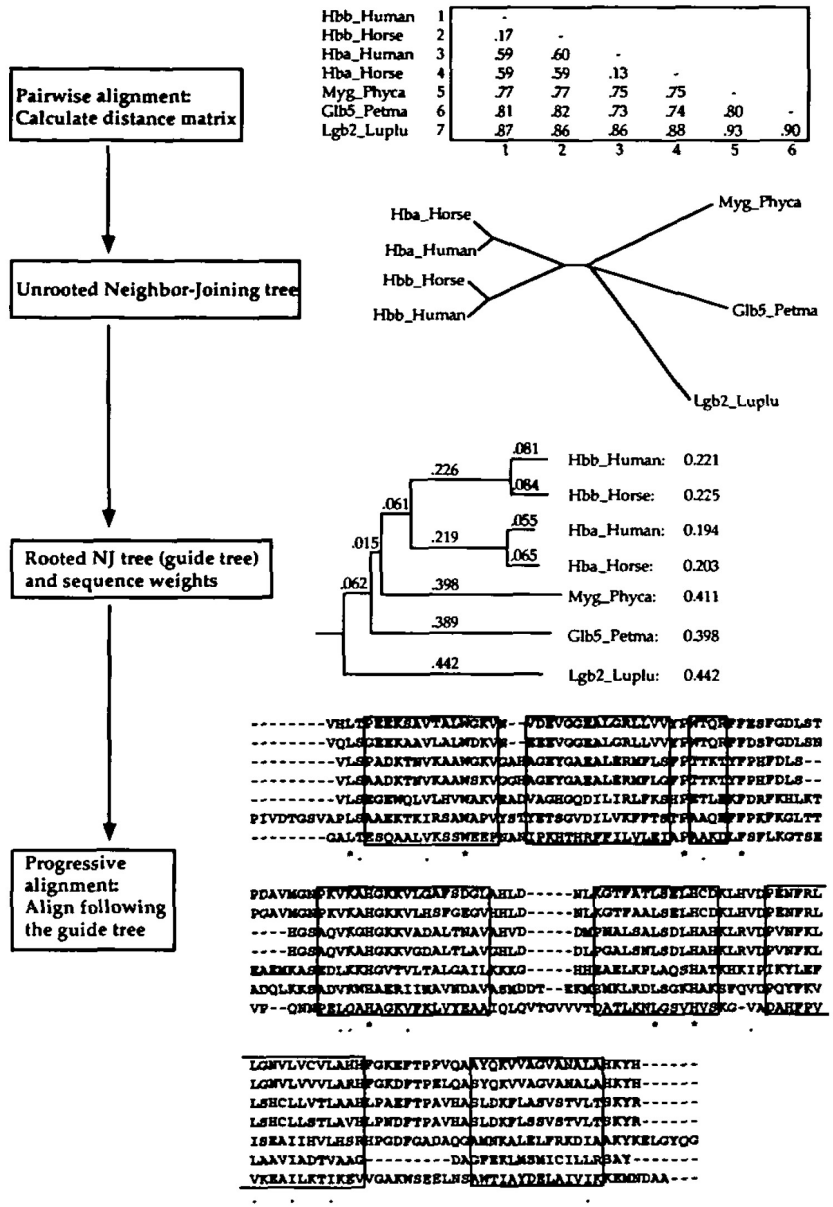
- P. Hogeweg and B. Hesper. The alignment of sets of sequences and the construction of phyletic trees: an integrated method. *J Mol Evol*, 20(2):175–186, 1984.
- D. G. Higgins and P. M. Sharp. Clustal: a package for performing multiple sequence alignment on a microcomputer. *Gene*, 73(1):237–44, Dec 1988.

Remarks (digression)

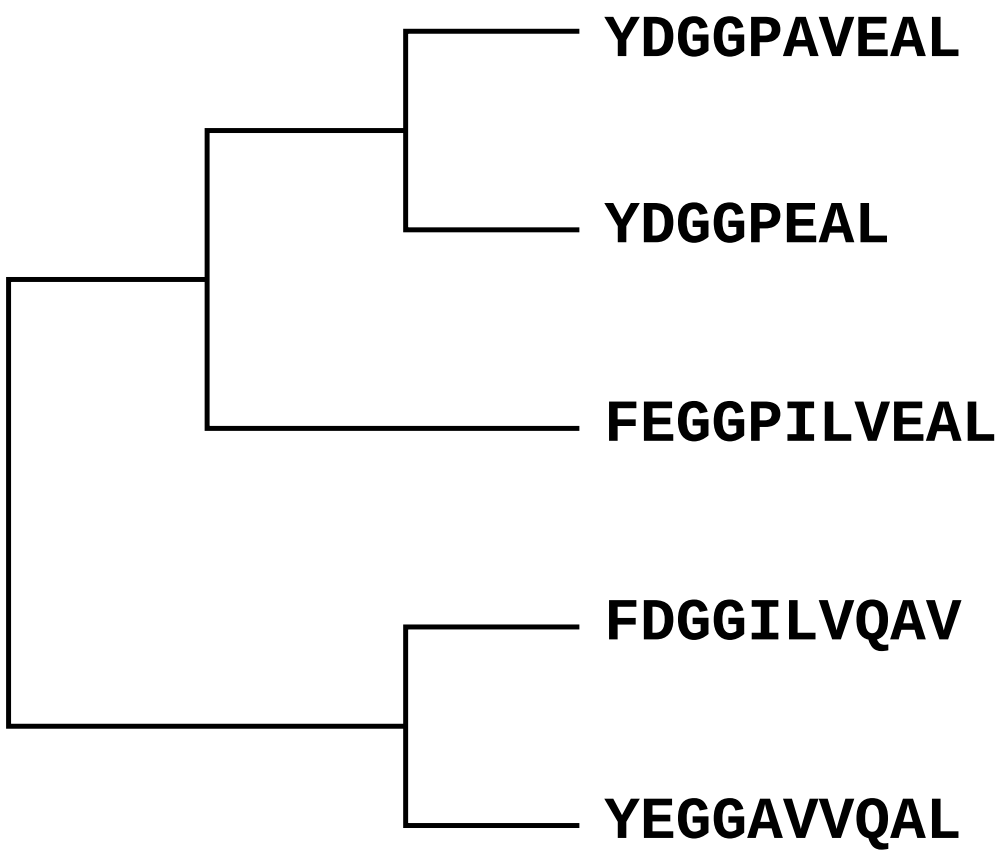
- Publications are the **currency** of academia!
- The number of citations demonstrates the **impact** of the work in the field.
- As of 2018-10-03, Des Higgins, the author of Clustal, has **125,800** citations (Scopus, 164,298 citations on Google Scholar)!

Progressive alignment

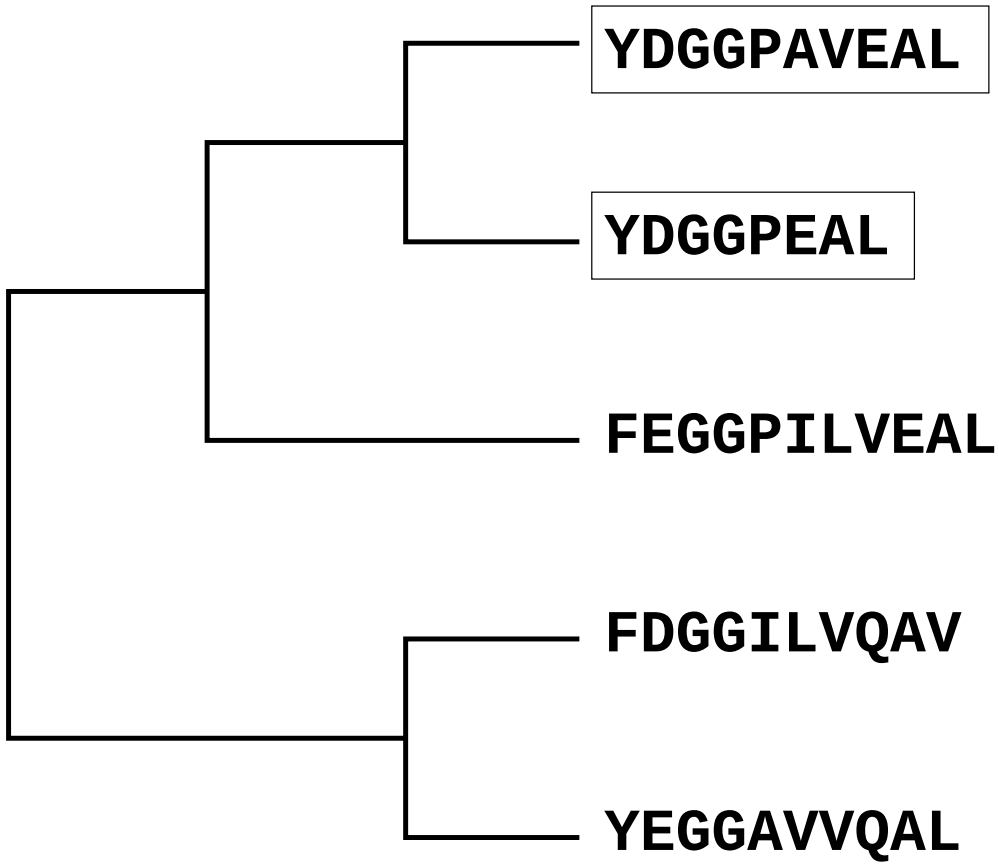
1. **Calculate** $d_{i,j}$, distance between sequences i and j, for all i and j
2. Build a **guide tree**
3. From the deepest node up to the root build all the **pairwise partial alignments** (bottom-up)


Pairwise alignment: Calculate distance matrix
Unrooted Neighbor-Joining tree
Rooted NJ tree (guide tree) and sequence weights
Progressive alignment: Align following the guide tree
Hbb_Human 1
Hbb_Horse 2 .17
Hba_Human 3 .59 .60
Hba_Horse 4 .59 .59 .13
Myg_Phyca 5 .77 .77 .75 .75
Glb5_Petma 6 .81 .82 .73 .74 .80
Lgb2_Luplu 7 .87 .86 .86 .88 .93 .90
1 2 3 4 5 6
Hba_Horse
Hba_Human
Hbb_Horse
Hbb_Human
Myg_Phyca
Glb5_Petma
Lgb2_Luplu
.081
.226
.084
.061
.055
.219
.015
.065
.062
.398
.389
.442
Hbb_Human: 0.221
Hbb_Horse: 0.225
Hba_Human: 0.194
Hba_Horse: 0.203
Myg_Phyca: 0.411
Glb5_Petma: 0.398
Lgb2_Luplu: 0.442

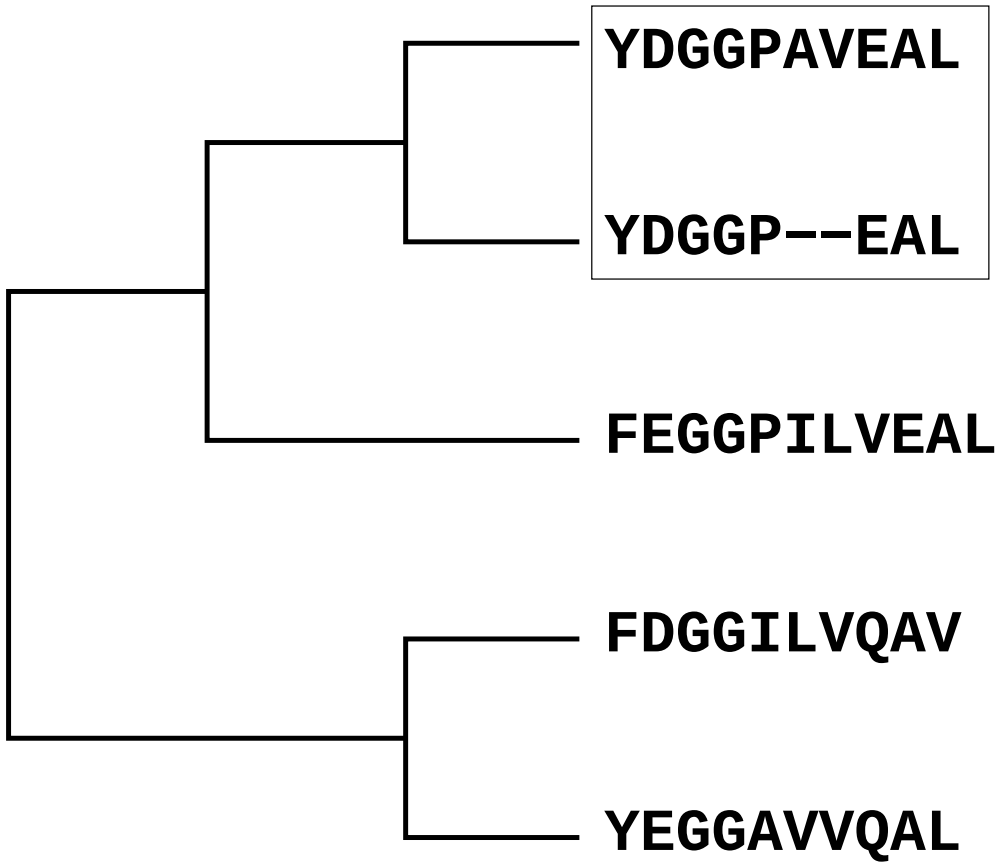
CLUSTALW



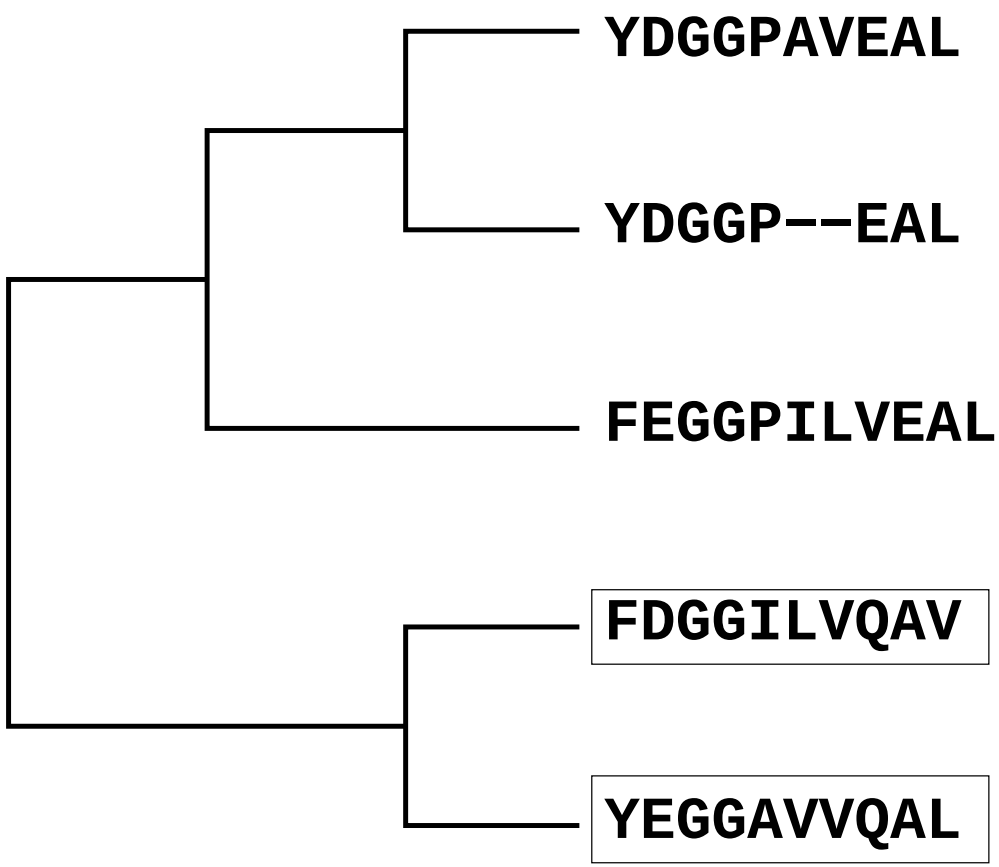
CLUSTALW



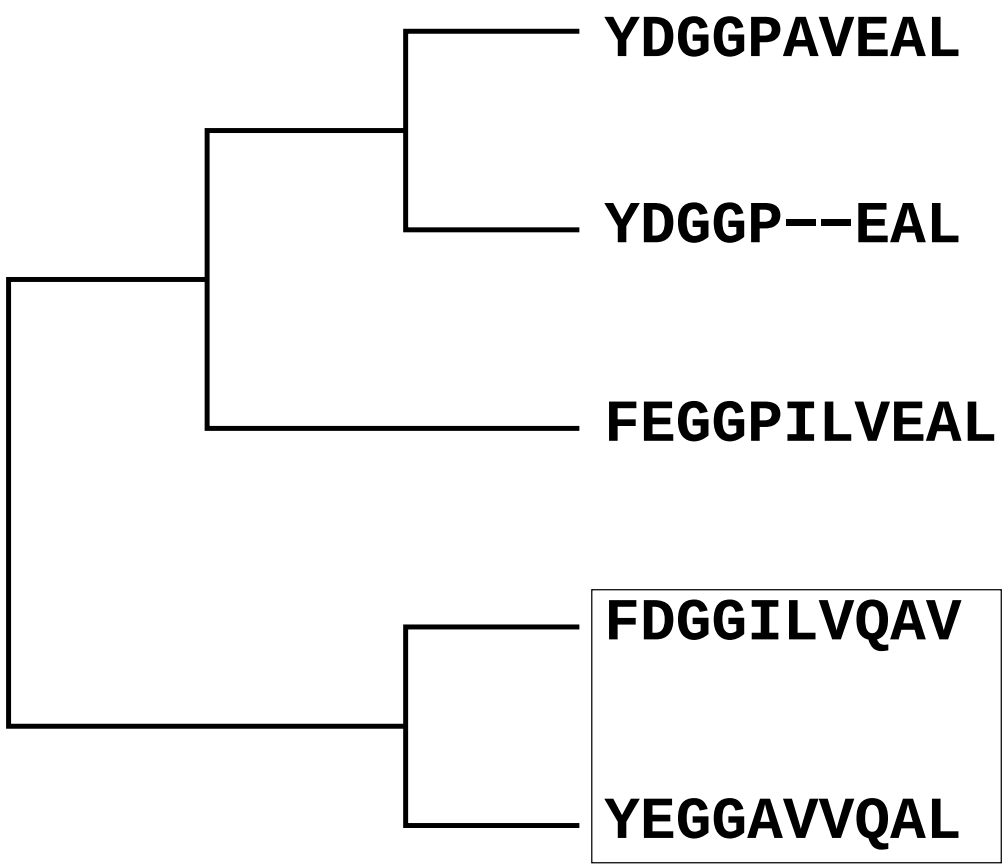
CLUSTALW



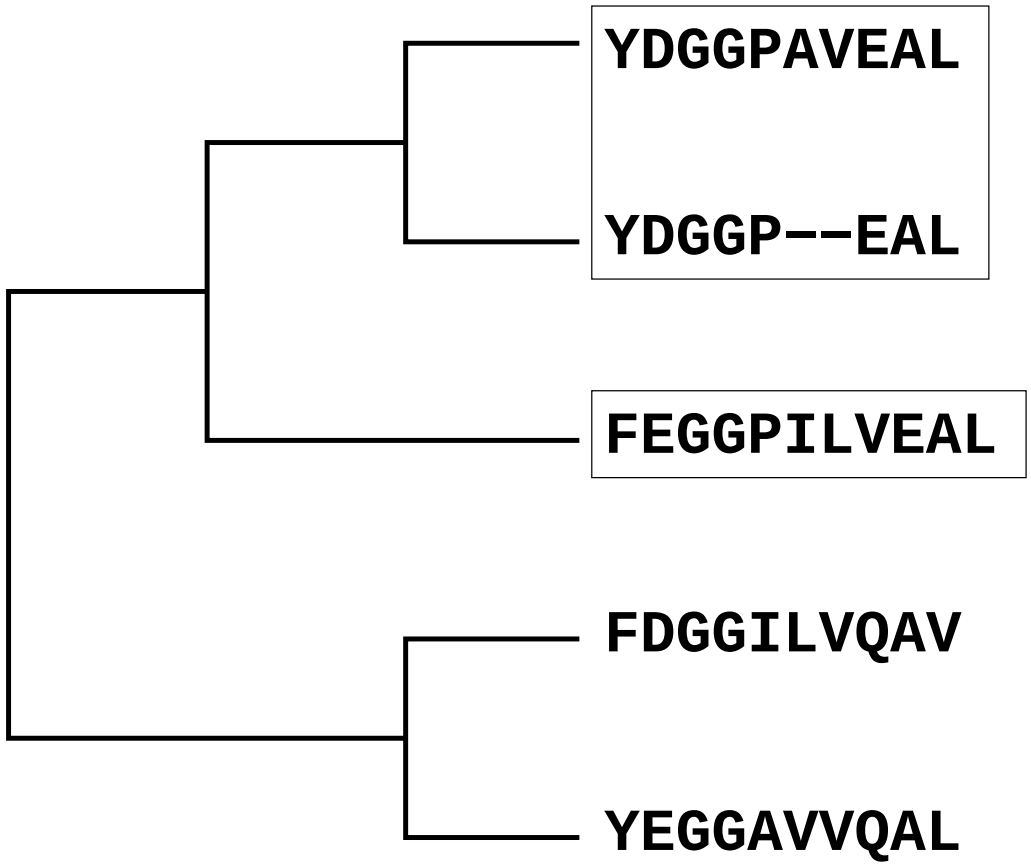
CLUSTALW



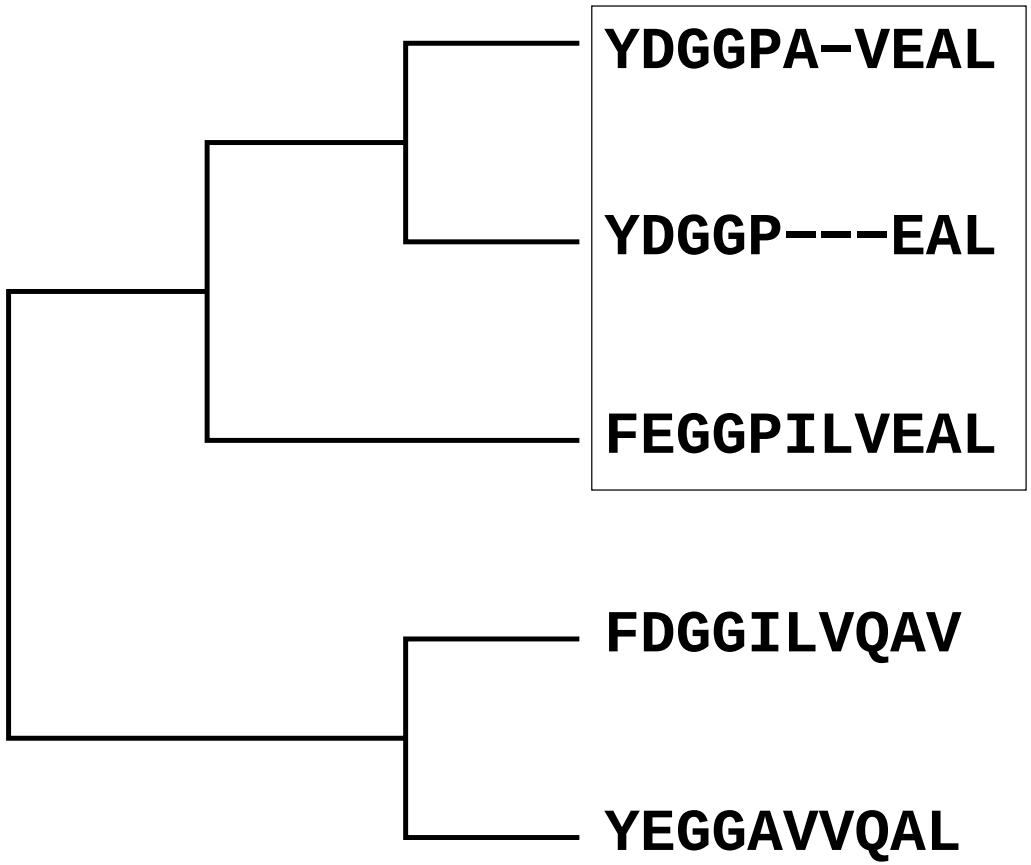
CLUSTALW



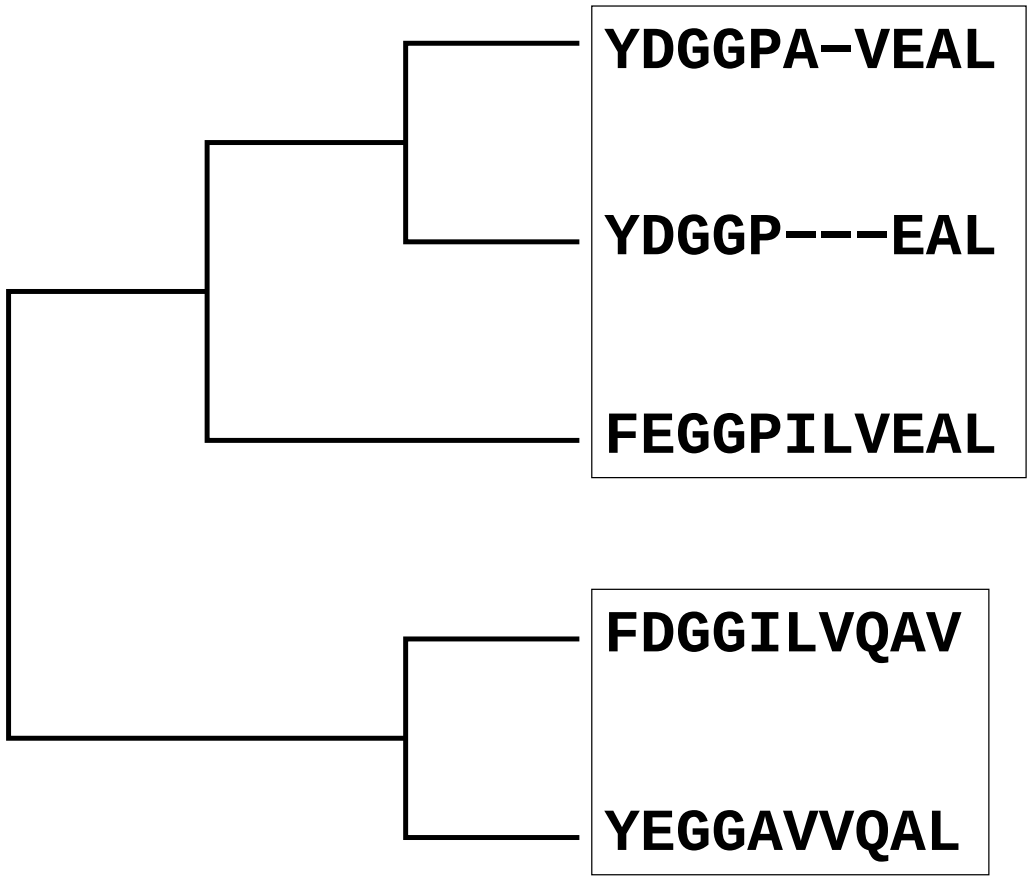
CLUSTALW



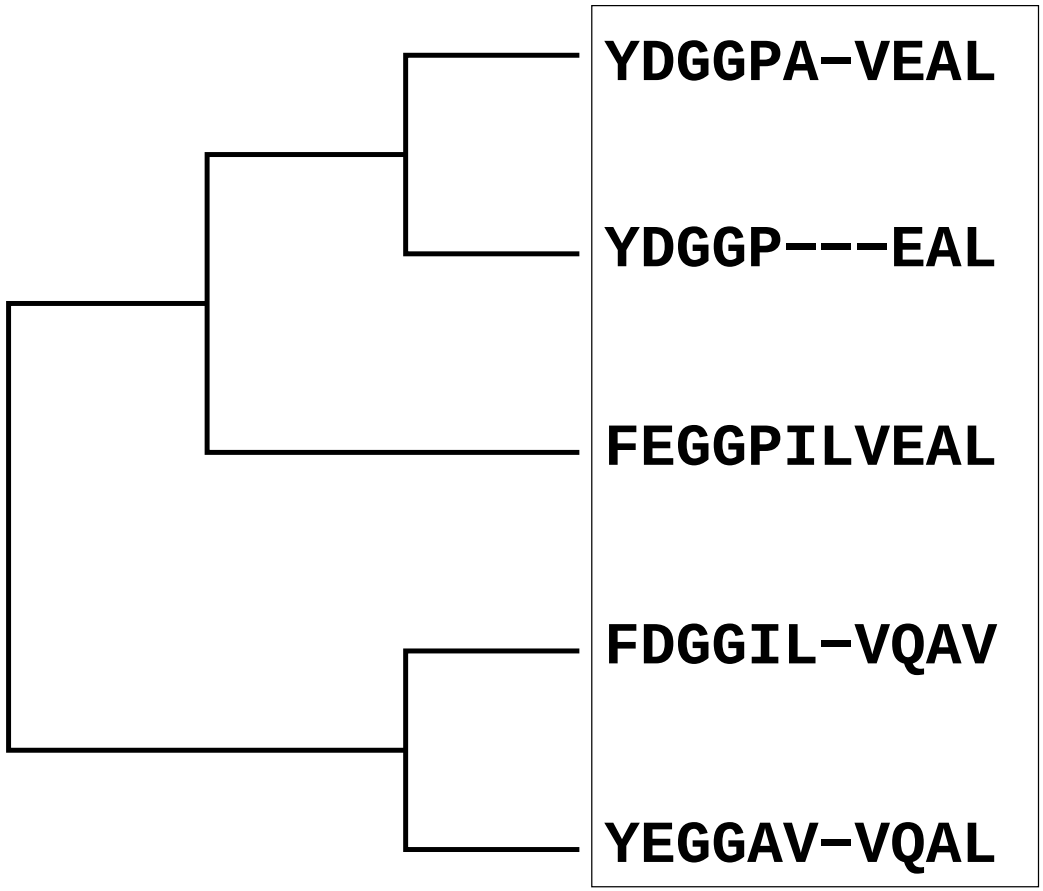
CLUSTALW



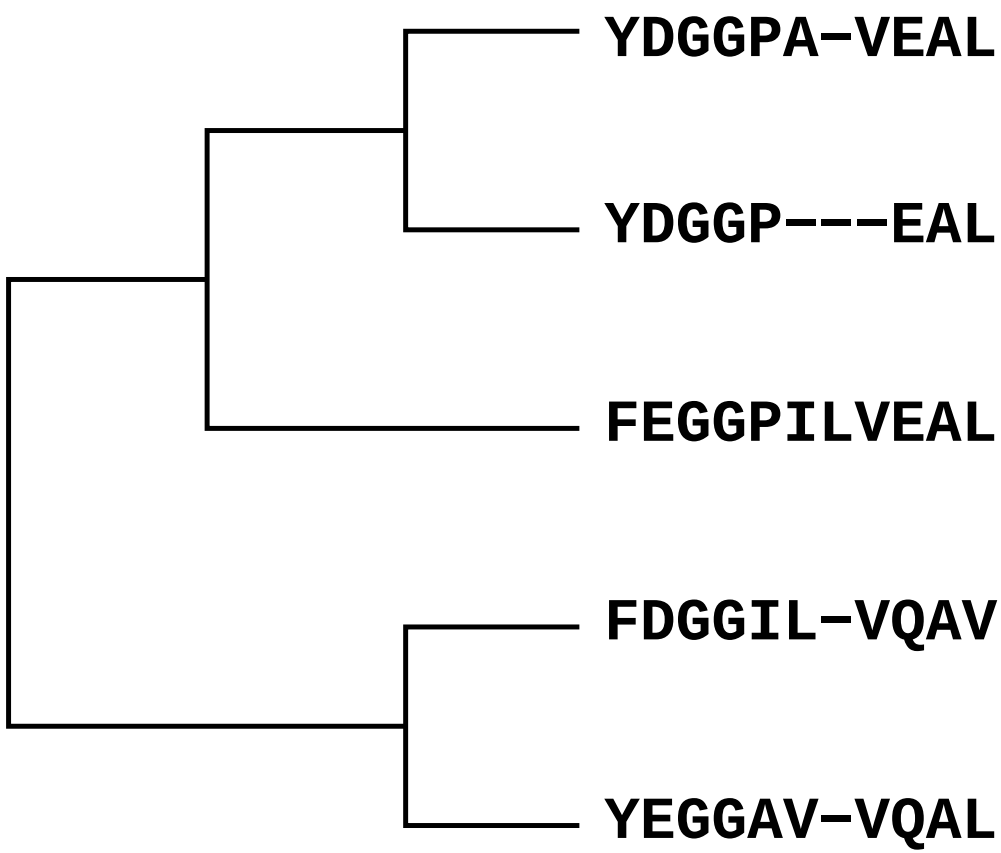
CLUSTALW



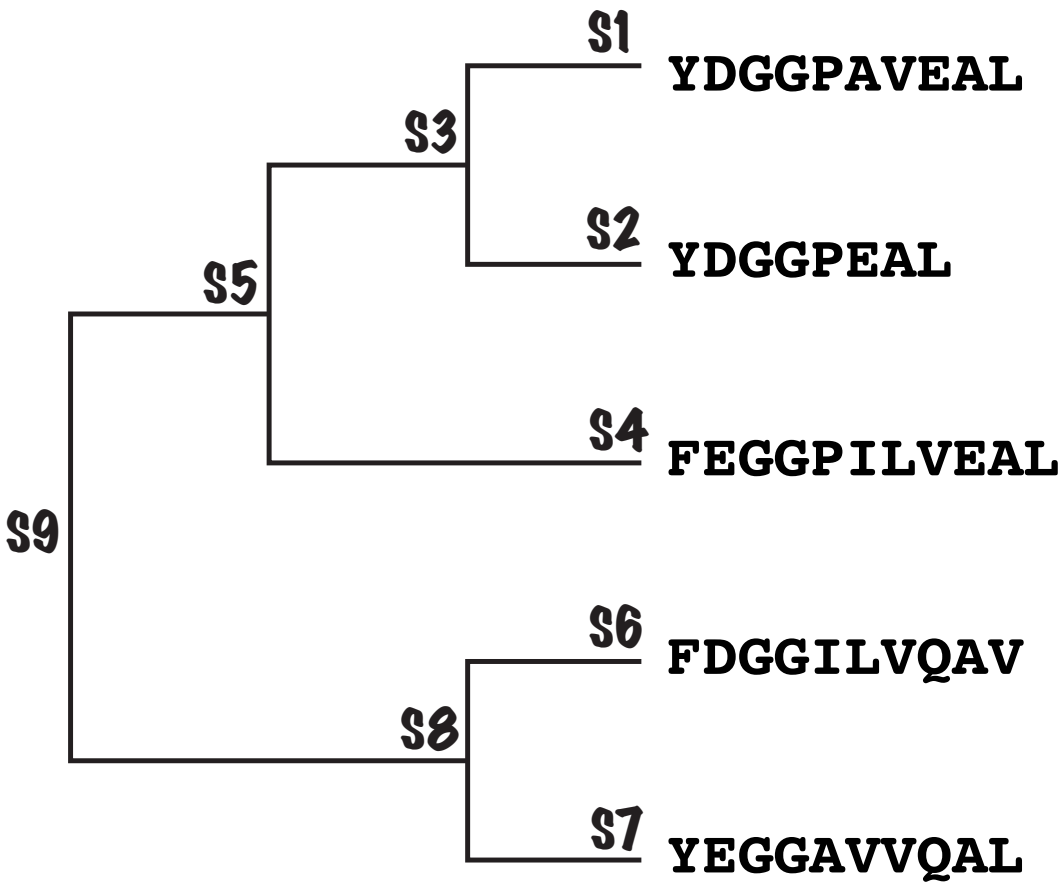
CLUSTALW



CLUSTALW

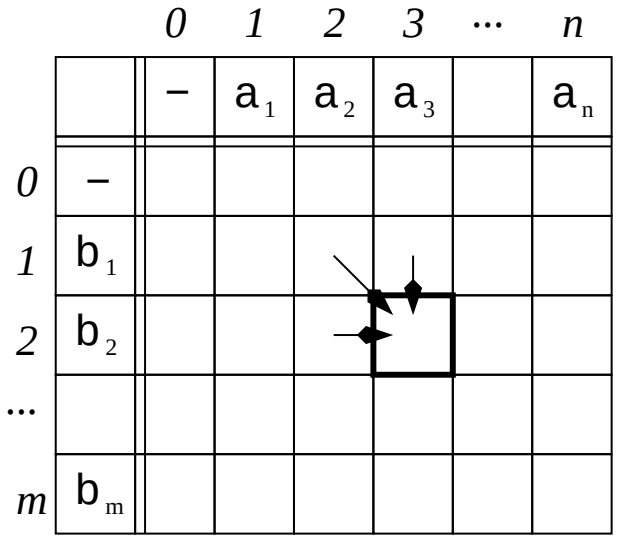


Progressive sequence alignment: take 2



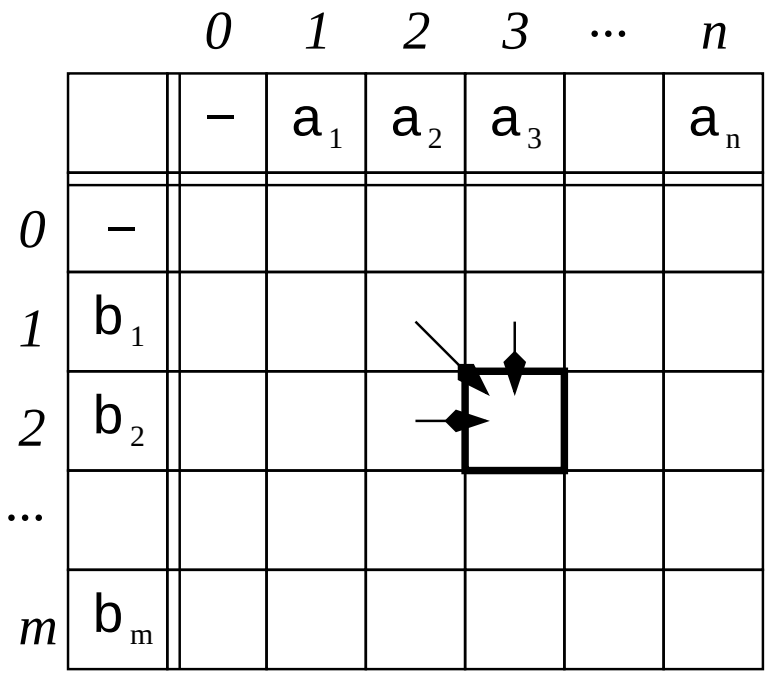
Source code available on the course Web site, as well as the Appendix.

Sequence vs Sequence, Sequence vs MSA, MSA vs MSA



- $a_1, a_2 \ldots a_n$ represents a sequence or an alignment
- When it represents a sequence, the a_i are the symbols of the sequence.
- When it represents an alignment, the a_i are the columns of the alignment.
- Similarly, for $b_1, b_2 \ldots b_m$.

S_1 vs S_2



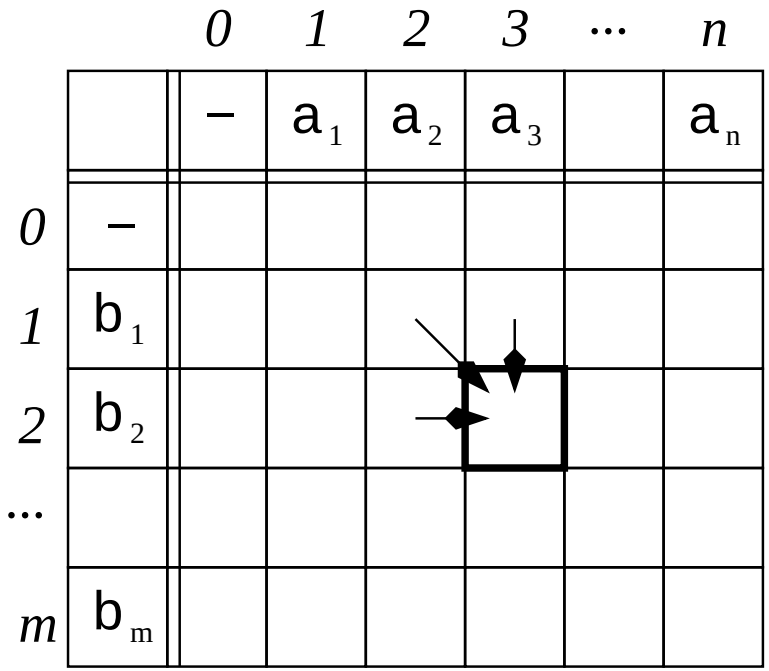
$a = S_2 =$ YDGGPEAL

$b = S_1 =$ YDGGPAVEAL

		Y	D	G	G	P	E	A	L
	[0]	[-6]	[-12]	[-18]	[-24]	[-30]	[-36]	[-42]	[-48]
Y	[-6]	[10]	[4]	[-2]	[-8]	[-14]	[-20]	[-26]	[-32]
D	[-12]	[4]	[14]	[8]	[2]	[-4]	[-10]	[-16]	[-22]
G	[-18]	[-2]	[8]	[19]	[13]	[7]	[1]	[-5]	[-11]
G	[-24]	[-8]	[2]	[13]	[24]	[18]	[12]	[6]	[0]
P	[-30]	[-14]	[-4]	[7]	[18]	[30]	[24]	[18]	[12]
A	[-36]	[-20]	[-10]	[1]	[12]	[24]	[30]	[26]	[20]
V	[-42]	[-26]	[-16]	[-5]	[6]	[18]	[24]	[30]	[28]
E	[-48]	[-32]	[-22]	[-11]	[0]	[12]	[22]	[24]	[27]
A	[-54]	[-38]	[-28]	[-17]	[-6]	[6]	[16]	[24]	[22]
L	[-60]	[-44]	[-34]	[-23]	[-12]	[0]	[10]	[18]	[30]

YDGGP--EAL
YDGGPAVEAL

S_3 vs S_4



$a = S_4 =$ FEGGPILVEAL

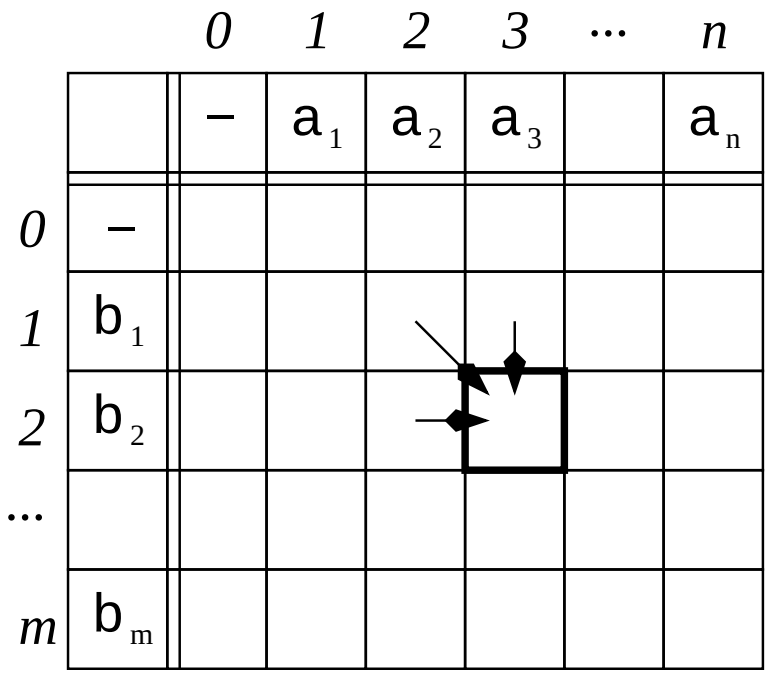
$b = S_3 =$ YDGGPAVEAL

YDGGP--EAL

```
                F     E     G     G     P     I     L     V     E     A     L
    [   0][ -12][ -24][ -36][ -48][ -60][ -72][ -84][ -96][-108][-120][-132]
Y Y [  -2][  24][  12][   0][ -12][ -24][ -36][ -48][ -60][ -72][ -84][ -96]
D D [ -10][  16][  34][  22][  10][  -2][ -14][ -26][ -38][ -50][ -62][ -74]
G G [ -17][   9][  27][  49][  37][  25][  13][   1][ -11][ -23][ -35][ -47]
G G [ -24][   2][  20][  42][  64][  52][  40][  28][  16][   4][  -8][ -20]
P P [ -30][  -4][  14][  36][  58][  82][  70][  58][  46][  34][  22][  10]
A - [ -42][ -16][   2][  24][  46][  70][  69][  57][  46][  34][  24][  12]
V - [ -54][ -28][ -10][  12][  34][  58][  62][  59][  49][  37][  25][  14]
E E [ -62][ -36][ -16][   4][  26][  50][  58][  60][  59][  61][  49][  37]
A A [ -72][ -46][ -26][  -6][  16][  40][  50][  56][  62][  61][  67][  55]
L L [ -78][ -52][ -32][ -12][  10][  34][  50][  68][  66][  62][  63][  85]

FEGGPILVEAL
YDGGP---EAL
YDGGPA-VEAL
```

S_5 vs S_6



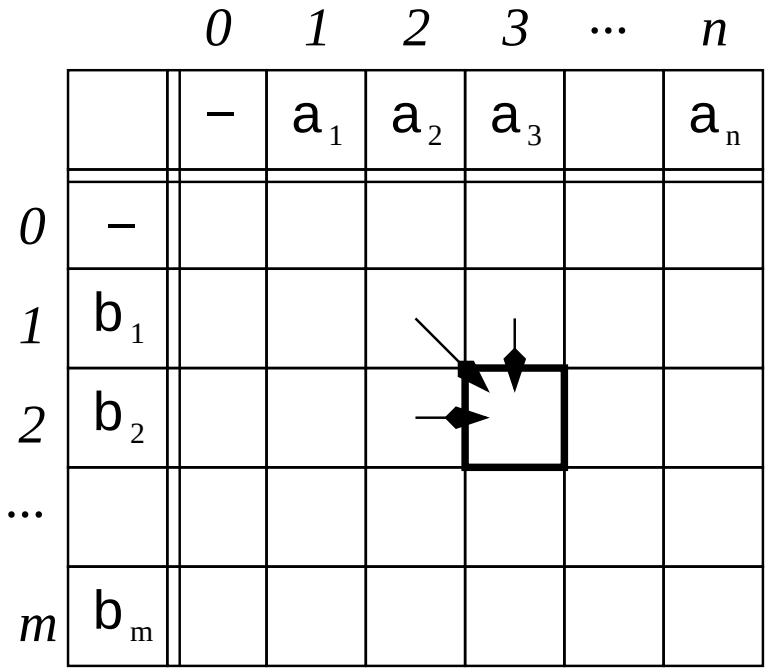
$a = S_6 = \texttt{YEGGAVVQAL}$

$b = S_5 = \texttt{FDGGILVQAV}$

```
            Y     E     G     G     A     V     V     Q     A     L
  [  0][ -6][-12][-18][-24][-30][-36][-42][-48][-54][-60]
F [ -6][  7][  1][ -5][-11][-17][-23][-29][-35][-41][-47]
D [-12][  1][ 10][  4][ -2][ -8][-14][-20][-26][-32][-38]
G [-18][ -5][  4][ 15][  9][  3][ -3][ -9][-15][-21][-27]
G [-24][-11][ -2][  9][ 20][ 14][  8][  2][ -4][-10][-16]
I [-30][-17][ -8][  3][ 14][ 19][ 18][ 12][  6][  0][ -6]
L [-36][-23][-14][ -3][  8][ 13][ 21][ 20][ 14][  8][  6]
V [-42][-29][-20][ -9][  2][  8][ 17][ 25][ 19][ 14][ 10]
Q [-48][-35][-26][-15][ -4][  2][ 11][ 19][ 29][ 23][ 17]
A [-54][-41][-32][-21][-10][ -2][  5][ 13][ 23][ 31][ 25]
V [-60][-47][-38][-27][-16][ -8][  2][  9][ 17][ 25][ 33]
```

FDGGILVQAV
YEGGAVVQAL

S_7 vs S_8



$a = S_8 =$ FDGGILVQAV
YEGGAVVQAL
$b = S_7 =$ FEGGPILVEAL
YDGGP---EAL
YDGGPA-VEAL

```
                  F     D     G     G     I     L     V     Q     A     V
                  Y     E     G     G     A     V     V     Q     A     L
      [   0][ -29][ -62][ -93][-124][-161][-195][-227][-259][-293][-327]
Y Y F [ -12][  81][  48][  17][ -14][ -51][ -85][-117][-149][-183][-217]
D D E [ -38][  55][ 115][  84][  53][  16][ -18][ -50][ -82][-116][-150]
G G G [ -59][  34][  94][ 165][ 134][  97][  63][  31][  -1][ -35][ -69]
G G G [ -80][  13][  73][ 144][ 215][ 178][ 144][ 112][  80][  46][  12]
P P P [ -98][  -5][  55][ 126][ 197][ 229][ 195][ 163][ 134][ 106][  72]
A - I [-135][ -42][  18][  89][ 160][ 192][ 210][ 182][ 150][ 116][  87]
- - L [-159][ -66][  -6][  65][ 136][ 168][ 186][ 182][ 150][ 116][  90]
V - V [-191][ -98][ -38][  33][ 104][ 136][ 162][ 186][ 158][ 132][ 110]
E E E [-215][-122][ -62][   9][  80][ 112][ 138][ 166][ 214][ 180][ 146]
A A A [-245][-152][ -92][ -21][  50][  88][ 114][ 148][ 184][ 234][ 200]
L L L [-263][-170][-110][ -39][  32][  70][ 132][ 148][ 166][ 216][ 278]

FDGGIL-VQAV
YEGGAV-VQAL
FEGGPILVEAL
YDGGP---EAL
YDGGPA-VEAL
```

Fine-tuning

- **Weighting scheme** compensates for large families
- **Close sequences** are aligned with **BLOSUM80**, whilst **distant ones** are aligned with **BLOSUM50**
- **Gap opening** is a function of the amino acid found at that position, and reduced if the position is embedded into a region of 5 or more hydrophilic residues
- **Gap penalty** increases if no gap is found at column or nearby
- Etc.

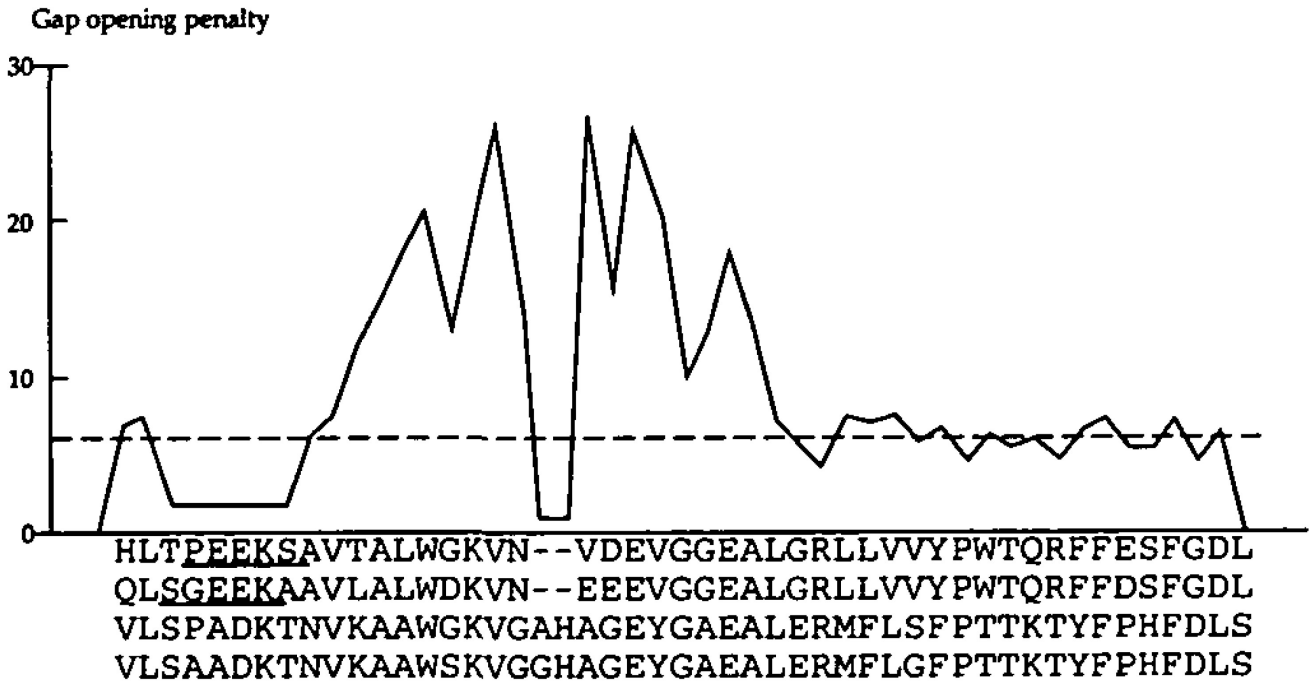


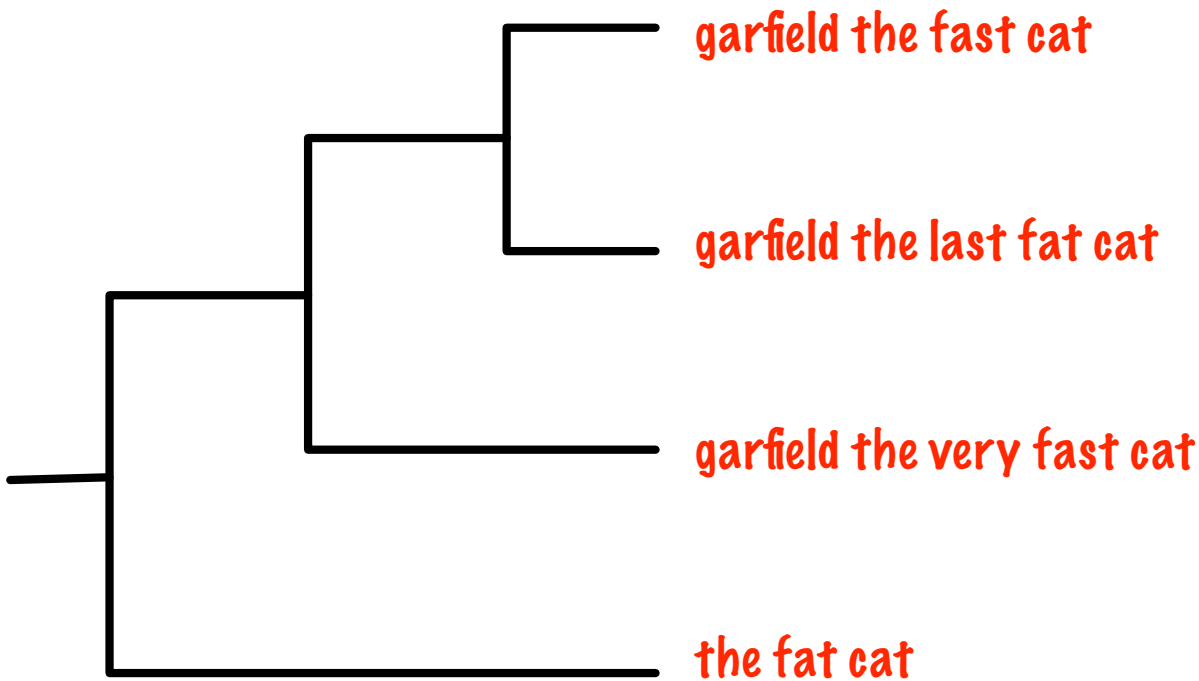
Figure 3. The variation in local gap opening penalty is plotted for a section of alignment. The inital gap opening penalty is indicated by a dotted line. Two hydrophilic stretches are underlined. The lowest penalties correspond to the ends of the alignment, the hydrophilic stretches and the two positions with gaps. The highest values are within 8 residues of the two gap positions. The rest of the variation is caused by the residue specific gap penalties (12).

Progressive alignment methods: limitations

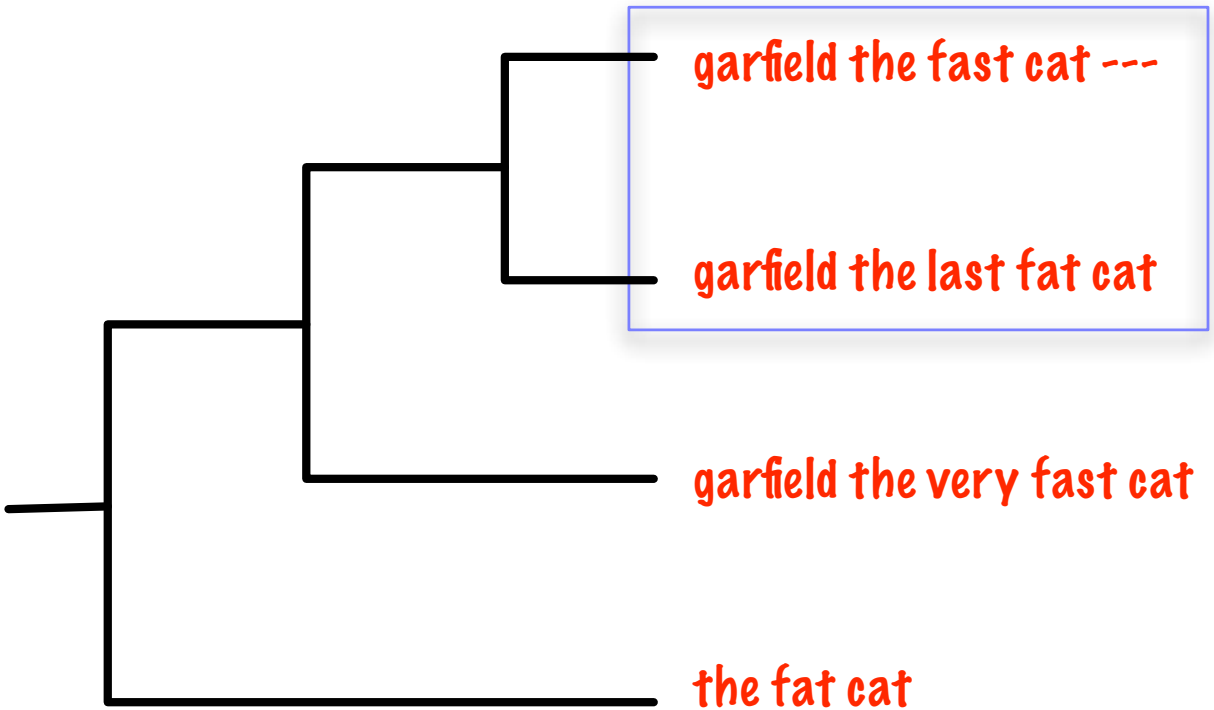
Progressive alignment methods: limitations

- Progressive alignment methods are **heuristics** (greedy algorithm)
- No attempt is made to optimize a **global score**
- Produce **reasonable** alignments
- **Fast**

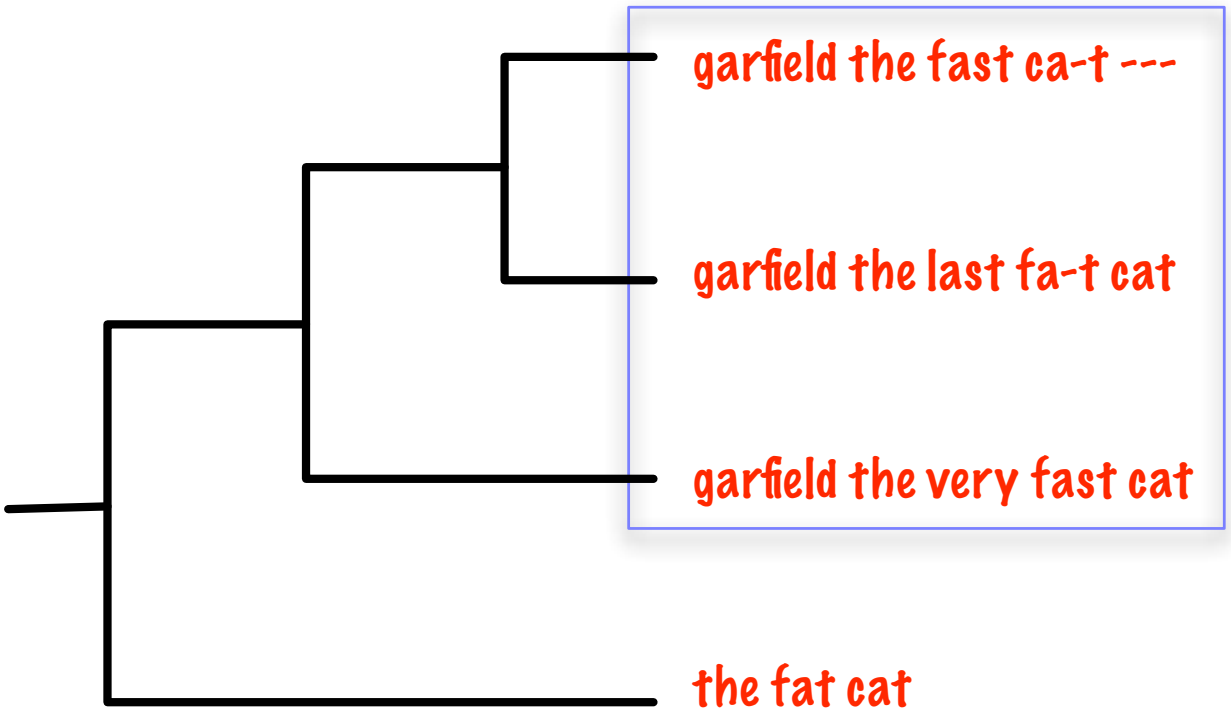
Progressive alignment methods: limitations



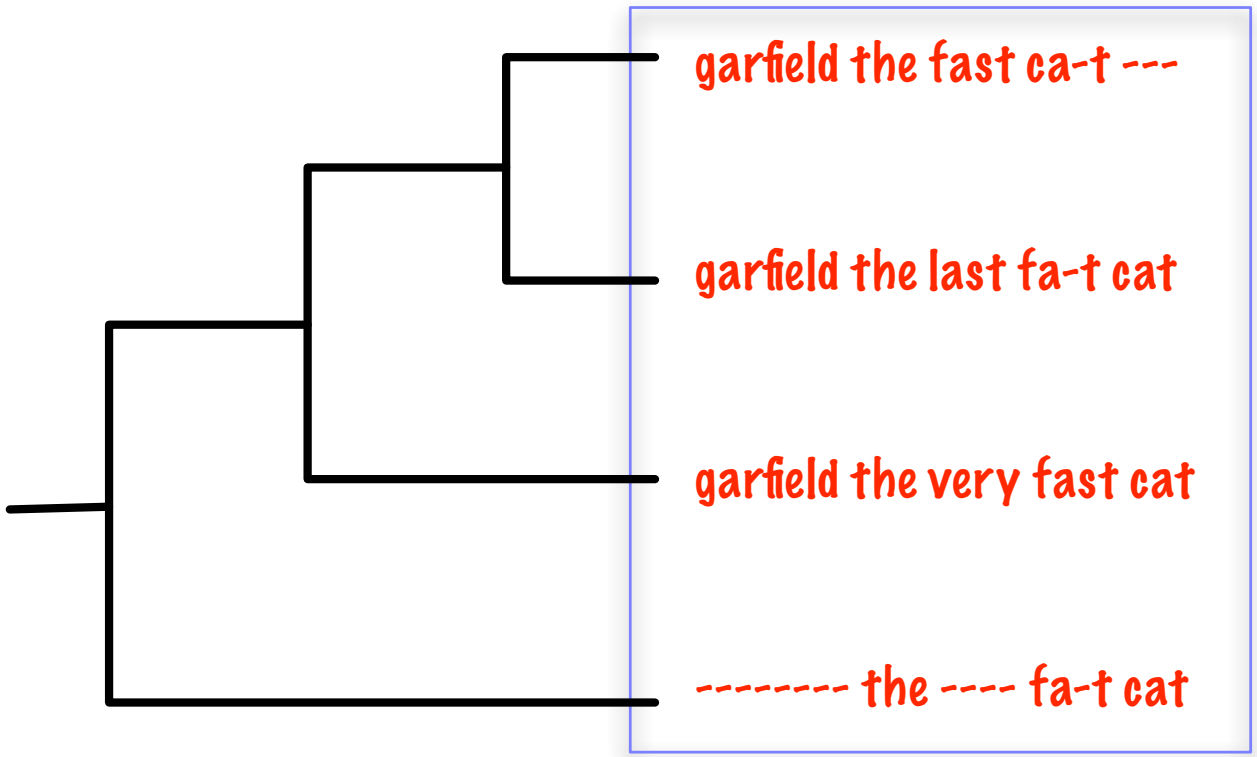
Progressive alignment methods: limitations



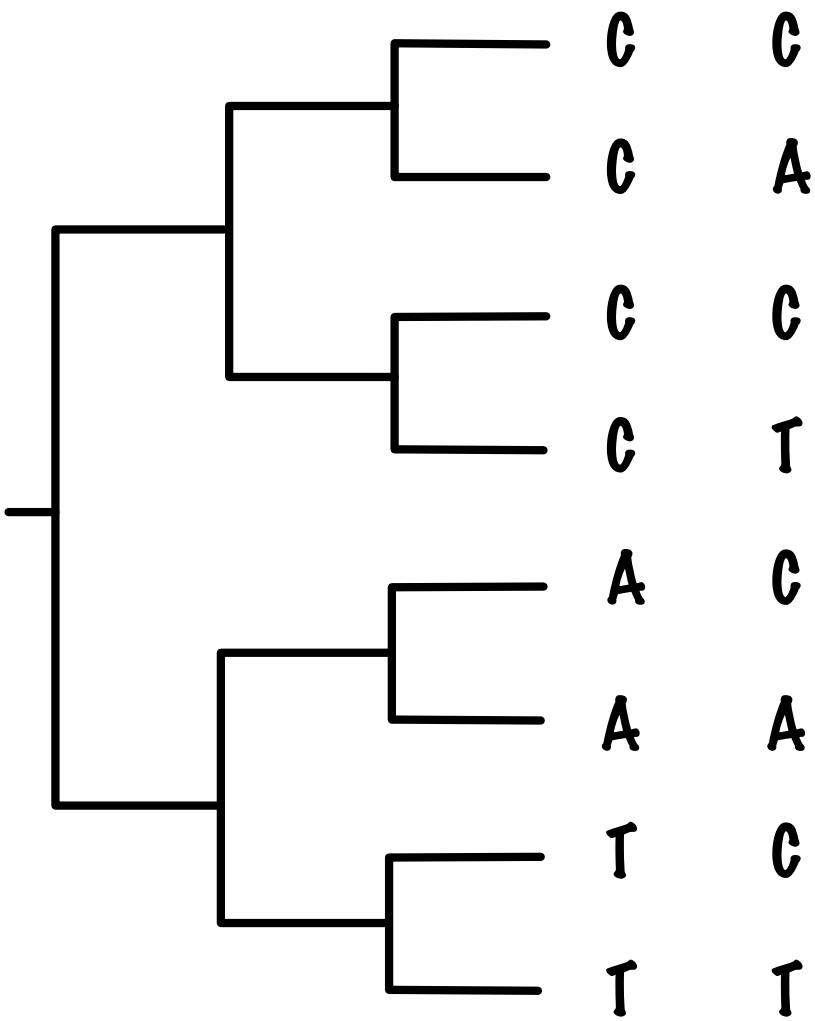
Progressive alignment methods: limitations



Progressive alignment methods: limitations



Sum of pairs



Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences
3. if the score of the alignment **improves**, use that alignment in place of the original one

Can be added to any base method.

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences
3. if the score of the alignment **improves**, use that alignment in place of the original one
4. repeat from 1 until **no improvement is observed** or the maximum number of iterations has been reached

Can be added to any base method.

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences
3. if the score of the alignment **improves**, use that alignment in place of the original one
4. repeat from 1 until **no improvement is observed** or the maximum number of iterations has been reached

Can be added to any base method.

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences
3. if the score of the alignment **improves**, use that alignment in place of the original one
4. repeat from 1 until **no improvement is observed** or the maximum number of iterations has been reached

Can be added to any base method.

Effective; +6% improvement ClustalW/HOMSTRAD.

Itarative Progressive Aligments

Progressive alignment methods cannot re-evaluate a decision that was made in the early stages of the algorithm!

Iterative methods:

1. **take out** of the alignment a sequence that has been aligned at a previous step
2. **re-align** that sequence against the remainging aligned sequences
3. if the score of the alignment **improves**, use that alignment in place of the original one
4. repeat from 1 until **no improvement is observed** or the maximum number of iterations has been reached

Can be added to any base method.

Effective; +6% improvement ClustalW/HOMSTRAD.

Most modern algorithms use iteration; ProbCons and Muscle do.

Itarative Progressive Aligments

- I. M. Wallace, O. O'Sullivan, and D. G. Higgins. Evaluation of iterative alignment algorithms for multiple alignment. *Bioinformatics*, 21(8):1408–14, Apr 2005.
- Evaluates 3 schemes for the iterations with 5 algorithms (ProbCons, Muscle, T-Coffee, ClustalW and Mafft (FFT-NSI).
- Modest improvements on HOM184, 0.18 (ProbCons) – 4.10 (Mafft)%
- Larger improvements on HOM37, 0 (ProbCons) – 13.56 (Mafft)%

Gold standards: evaluating the accuracy of MSAs

MSAs are compared to reference alignments to determine an accuracy score.

The reference alignments are generally created from protein **structures** and/or **manually** curated.

- **BAliBase** (first large data-set, human intervention high)
- **HOMSTRAD**, **OXBENCH**, **PREFAB**, **SABmark**
- **IRMbase** (simulated sequence data)

Gold standards: evaluating the accuracy of MSAs

The accuracy is often measured as the **fraction of columns that are identical**, in both test and reference alignments.

(See **aln_compare** by Notredame et al 2000)

Recent Methods

Table 1. Summary of the Methods Described in the Review

Method	Score	Templates	Validation Values		Server
			PreFab	HOMSTRAD	
ClustalW [14]	Matrix	—	61.80 [12]	—	http://www.ebi.ac.uk/clustalw/
Kalign	Matrix	—	63.00 [18]	—	http://msa.cgb.ki.se/
MUSCLE [6]	Matrix	—	68.00 [16]	45.0 [9]	http://www.drive5.com/muscle/
T-Coffee [10]	Consistency	—	69.97 [12]	44.0 [9]	http://www.tcoffee.org/
ProbCons [7]	Consistency	—	70.54 [12]	—	http://probcons.stanford.edu/
MAFFT [8]	Consistency	—	72.20 [12]	—	http://align.genome.jp/mafft/
M-Coffee [12]	Consistency	—	72.91 [12]	—	http://www.tcoffee.org/
MUMMALS [16]	Consistency	—	73.10 [16]	—	http://prodata.swmed.edu/mummals/
DbClustal [24]	Profiles	—	—	—	http://bips.u-strasbg.fr/PipeAlign/
PRALINE [9]	Matrix	Profiles	—	50.2 [9]	http://zeus.cs.vu.nl/programs/pralinewww/
PROMALS [16]	Consistency	Profiles	79.00 [16]	—	http://prodata.swmed.edu/promals/
SPEM [28]	Matrix	Profiles	77.00 [28]	—	http://sparks.informatics.iupui.edu/Softwares-Services_files/spem.htm
Expresso [13]	Consistency	Structures	—	71.9 [11][a]	http://www.tcoffee.org/
T-Lara [29]	Consistency	Structures	—	—	https://www.mi.fu-berlin.de/w/LiSA/

Progressive Aligments and **Scoring Schemes**

- Matrix-based: ClustalW, MUSCLE, Kalign;
- Consistency-based: Dialign, T-Coffee, PCMA, ProbCons, MUMMALS, MAFFT. (Which PAM, you said?)

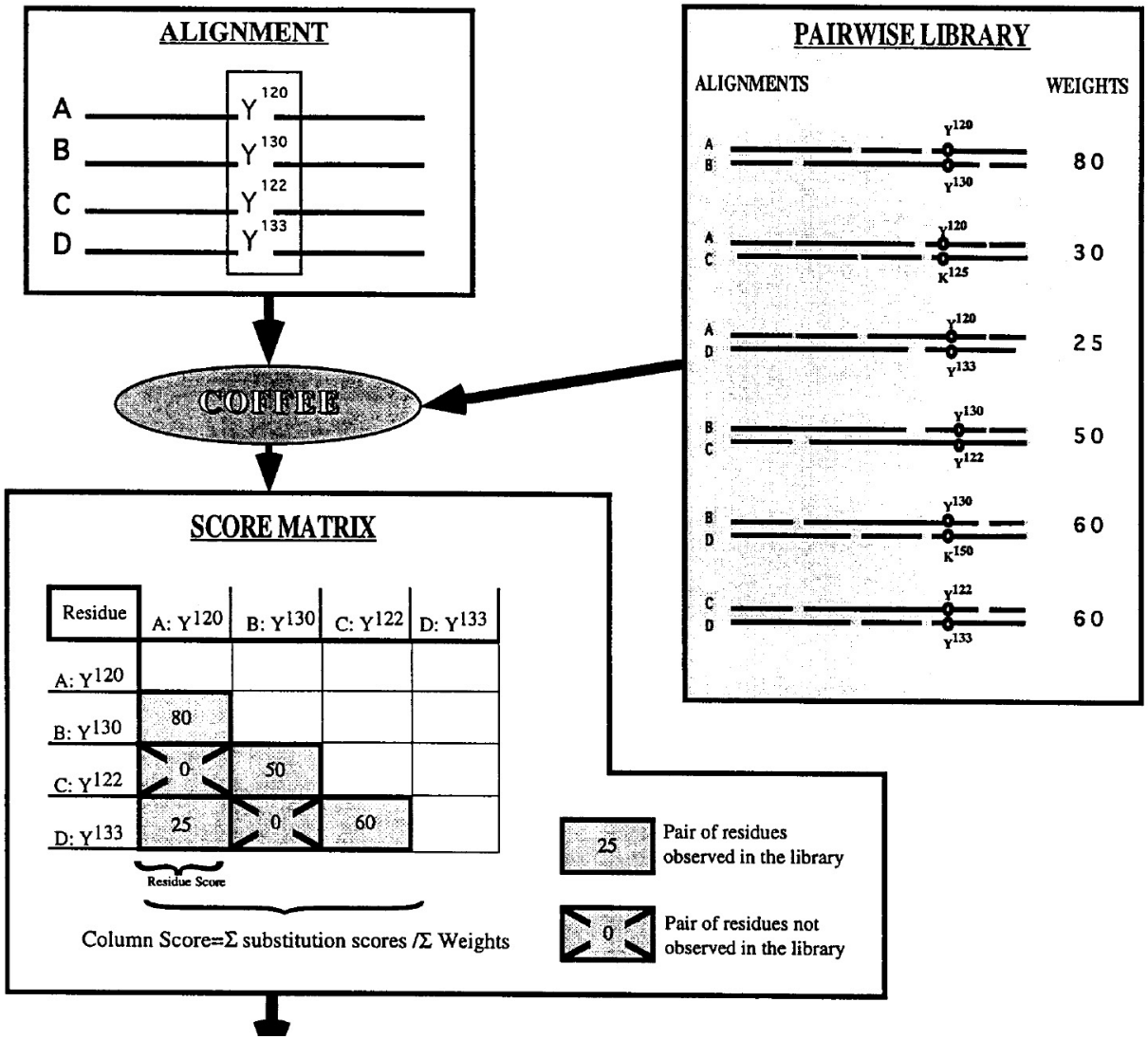
Studies suggest that consistency-based scoring schemes produce more accurate aligments than matrix-based schemes but are k-times slower.

Matrix-based scoring functions

The scoring scheme consists of a substitution matrix (such as PAM or BLOSUM); $s(a, b)$.

Consistency-based scoring functions

- Consistency-based methods are trying to find a multiple sequence alignment that has the **highest level of similarity** with a collection of **pairwise alignments**.
- Here, the **sum-of-pair** score is replaced by an objective function that measures the **consistency** of the alignment with respect to an "all-against-all" collection of pairwise alignments (called library).


ALIGNMENT
A
B
C
D
Y120
Y130
Y122
Y133
PAIRWISE LIBRARY
ALIGNMENTS
WEIGHTS
80
30
25
50
60
60
COFFEE
SCORE MATRIX
Residue
A: Y120
B: Y130
C: Y122
D: Y133
80
0
50
25
0
60
Residue Score
Column Score=Σ substitution scores /Σ Weights
25
Pair of residues observed in the library
0
Pair of residues not observed in the library

Consistency-based aligments: COFFEE objective function

COFFEE = Consistency based Objective Function For alignmEnt Evaluation!

$$\text{COFFEE} = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} W_{i,j} \times \text{SCORE}(A_{i,j})}{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} W_{i,j} \times \text{LENGTH}(A_{i,j})}$$

- where $A_{i,j}$ is the pair of sequences S_i and S_j extracted from the multiple alignment.
- $\text{SCORE}(A_{i,j})$ is the number of pair of residues that are aligned in $A_{i,j}$ **AND** in the library.
- $\text{LENGTH}(A_{i,j})$ is the length of the alignment.
- $W_{i,j}$ is the weight assigned to the pair S_i and S_j.
- When all the $W_{i,j}$ are 1, the score ranges from 0 to 1, otherwise, the score is normalized to be in that range.

Consistency-based aligments

- **SAGA-COFFEE** is one of the first consistency-based method
- Consistency-based generally require **large** amounts of (time, memory) **resources**, which typically limits their application to set 100 or less sequences.
- **MAFFT** and **MUSCLE** scale to larger sets, still with good accuracies.

ProbCons

- C. B. Do, M. S. P. Mahabhashyam, M. Brudno, and S. Batzoglou. Probcons: Probabilistic consistency-based multiple sequence alignment. *Genome Res*, 15(2):330–40, Feb 2005.
- Defines a novel objective function, probabilistic consistency.
- On several benchmarks, as well as independent publications, ProbCons has been shown to be (one of) the best method for producing MSAs.

When everything else fails

When everything else fails

- Using additional sources of information
- Combining approaches (meta-methods)

Using additional sources of information

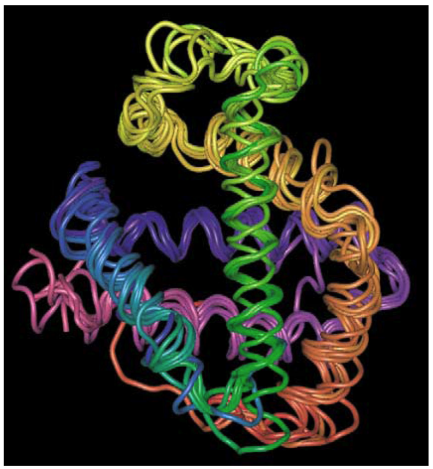
- Secondary structure
- Three-dimensional structure
- Profiles

3D-Coffee

```
hba_horse    ---------VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHF-DLS-
hba_human    ---------VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF-DLS-
hbb_horse    --------VQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDLSN
hbb_human    --------VHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDLST
glb5_petma   PIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLTT
myg_phyca    ---------VLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLKT
lgb2_luplu   --------GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE
                       *:  :   :   *  . .         :  .:   * :   *  :   .

hba_horse    ----HGSAQVKAHGKKVGDALTLAVGHLDD-----LPGALSNLSDLHAHKLRVDPVNFKL
hba_human    ----HGSAQVKGHGKKVADALTNAVAHVDD-----MPNALSALSDLHAHKLRVDPVNFKL
hbb_horse    PGAVMGNPKVKAHGKKVLHSFGEGVHHLDN-----LKGTFAALSELHCDKLHVDPENFRL
hbb_human    PDAVMGNPKVKAHGKKVLGAFSDGLAHLDN-----LKGTFATLSELHCDKLHVDPENFRL
glb5_petma   ADQLKKSADVRWHAERIINAVNDAVASMDDT--EKMSMKLRDLSGKHAKSFQVDPQYFKV
myg_phyca    EAEMKASEDLKKHGVTVLTALGAILKKKGH------HEAELKPAQSHATKHKIPIKYLEF
lgb2_luplu   VP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-VADAHFPV
                   . .:: *.  :   .                      .  *  .  :    : .

hba_horse    LSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSKYR------
hba_human    LSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------
hbb_horse    LGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH------
hbb_human    LGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------
glb5_petma   LAAVIADTVAAG---------DAGFEKLMSMICILLRSAY-------
myg_phyca    ISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
lgb2_luplu   VKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA---
             :   :  .:      .     ...       .   :     . . ..
```



Current Opinion in Structural Biology

Profiles

- **PRALINE** uses PSI-BLAST to collect homologous sequences for each of the input sequences.

  "by including up to 100 close homologues in the alignment, the accuracy of most methods increased noticeably. (...) the improvement was almost as good as including structural information."
- The profiles are used in place of the individual sequences in a progressive alignment.
- **Position-specific distribution**, instead of an identical global distribution.
- SPEM also predicts and uses secondary structure information.

Meta-methods: M-Coffee

Combines the results of MUSCLE, MAFFT, POA, Dialign-T, T-Coffee, ClustalW, PCMA and ProbCons.

Meta-methods: M-Coffee

Combines the results of MUSCLE, MAFFT, POA, Dialign-T, T-Coffee, ClustalW, PCMA and ProbCons.

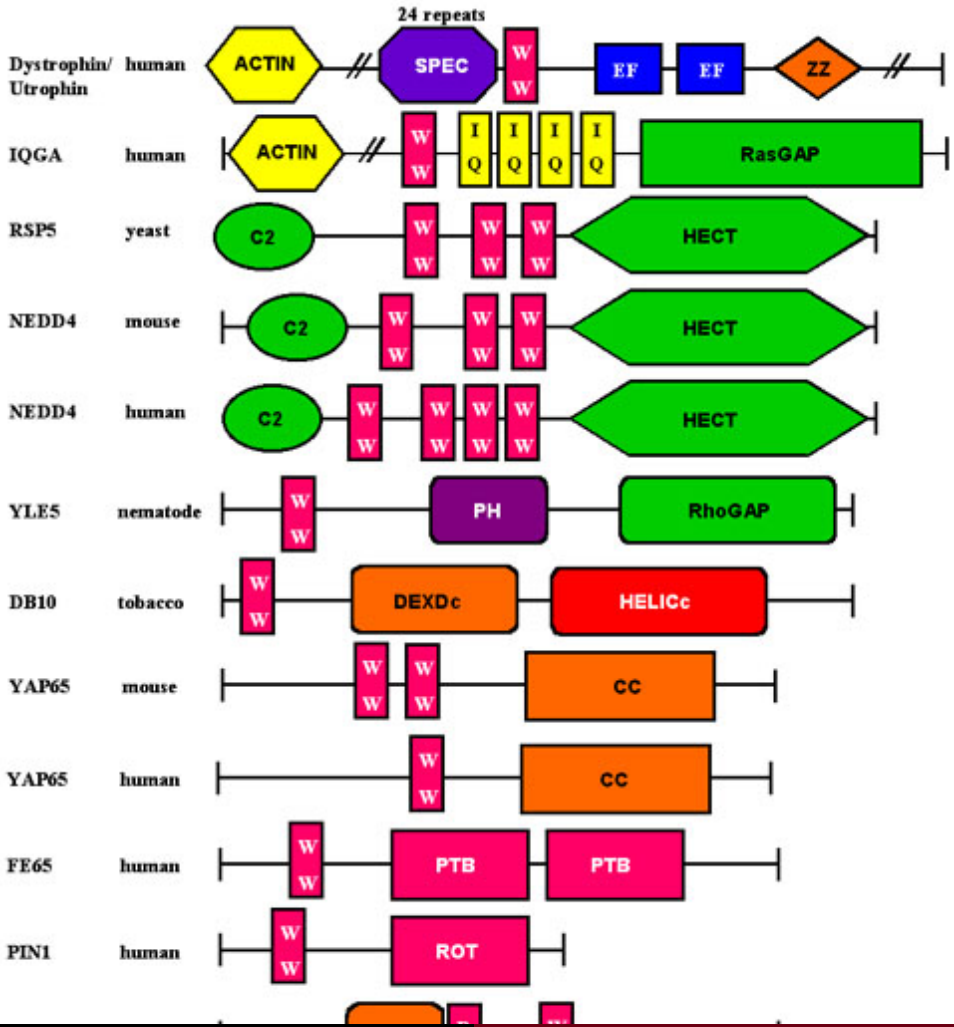
1. Makes your life easier!
2. Improved accuracy
3. Gives you local estimations of the reliability of your alignment

Extensions for **specific problems**

ALIGN-M, DIALIGN, POA and SATHMO are methods for handling sequence families consisting of co-linear conserved regions interspersed with variable regions.

Local multiple sequence alignments

- **Input:** proteins with diverse domain organizations.
- **Output:** an alignment of the homologous regions.



Local multiple sequence alignments

- No thoroughly tested methods exist for the local multiple sequence alignment problem.
- ABA is graphical tool meant to assist the process.
- ProDA (proda.stanford.edu) is an experimental approach.

Biological **accuracy**

- According to benchmark tests, MAFFT, MUSCLE, PROBCONS and T-COFFEE deliver the most realistic alignments
- Most modern algorithms produce more accurate alignment than CLUSTALW

Edgar and Batzoglou's **recommendations**

Context	**Methods**
$2-100$ sequences, $\lvert S_i\rvert < 10,000$, w/ 3D	3DCoffee
$2-100$ sequences, $\lvert S_i\rvert < 10,000$	PROBCONS, T-COFFEE, MAFF
$100-500$ sequences, $\lvert S_i\rvert < 10,000$	MAFFT, MUSCLE
> 500 sequences, $\lvert S_i\rvert < 10,000$	MAFFT, MUSCLE with specific c
$2-100$ sequences, including variables regions	DIALIGN
$2-100$ sequences, repeated or shuffled domains	ProDA
$2-100$ sequences, $\lvert S_i\rvert > 20,000$	CLUSTALW

Conclusions

1. ProbCons is the best individual method [Wallace 2005]
2. M-Coffee is the best meta-method [Notredame 2007]
3. "...the best methods have become indistinguishable, except when considering remote homologs (less than 25% identity)." PLoS Computational Biology 3(8):e123 August 2007
4. In the end, manual editing might (will) be needed
5. S. Griffiths-Jones and A. Bateman. The use of structure information to increase alignment accuracy does not aid homologue detection with profile HMMS *Bioinformatics*, 18(9):1243–1249, 2002.

Future developments

- Hydrophobicity dependent gap penalties helps (4 %) [Kececioglu]
- Input set dependent parameter selection
- Better use of phylogenetic information
- Hopefully, incorporating models of sequence evolution

References

1. C. Notredame. Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput Biol*, 3(8):e123, August 2007.
2. R. C. Edgar and S. Batzoglou. Multiple sequence alignment. *Curr Opin Struct Biol*, 16(3):368–373, 2006.
3. I. M. Wallace, G. Blackshields, and D. G. Higgins. Multiple sequence alignments. *Curr Opin Struct Biol*, 15(3):261–6, Jun 2005.

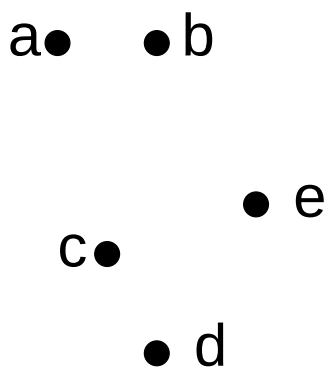
Appendix: Building the guide tree using UPGMA

UPGMA = Unweighted Pair Group Method using Arithmetic averages*.

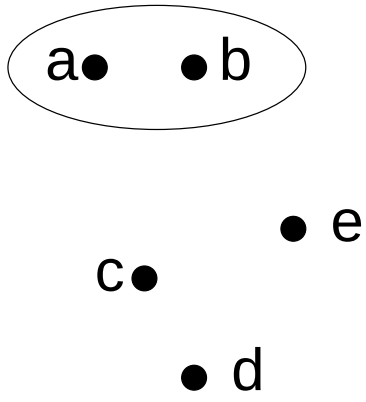
{ Initialization } Assign each sequence i to its own cluster C_i Define one leaf of T for each sequence, place it at height zero. { Iterations } Find the pair of clusters i and j which minimizes d_{ij}. Define a new cluster $C_k = C_i \bigcup C_j$. Calculate d_{kl} for all l. Create the parent node k of i and j at height $d_{ij}/2$ in T. Add k to the current list of clusters and remove i and j. { Termination } Stop when the list of clusters contains only one entry.

$\Rightarrow$ Simple and intuitive.

*See Durbin et al. p. 166

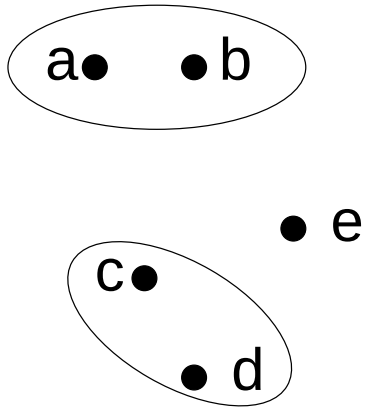


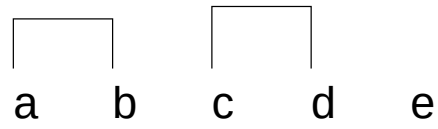
	a	b	c	d	e
a	0	10	21	32	25
b		0	21	30	18
c			0	11	16
d				0	18
e					0



	a	b	c	d	e
	{a,b}		c	d	e
{a,b}		0	21	31	21.5
c			0	11	16
d				0	18
e					0

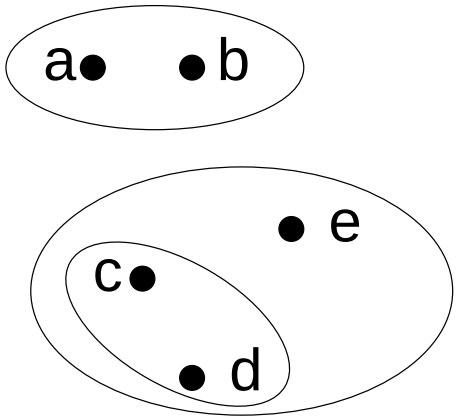
$$d_{\{ab\},e} = (d_{ae} + d_{be})/2 = (25 + 18)/2$$

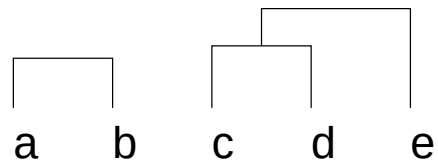




	{a,b}	{c,d}	e
{a,b}	0	26	21.5
{c,d}		0	17
e			0

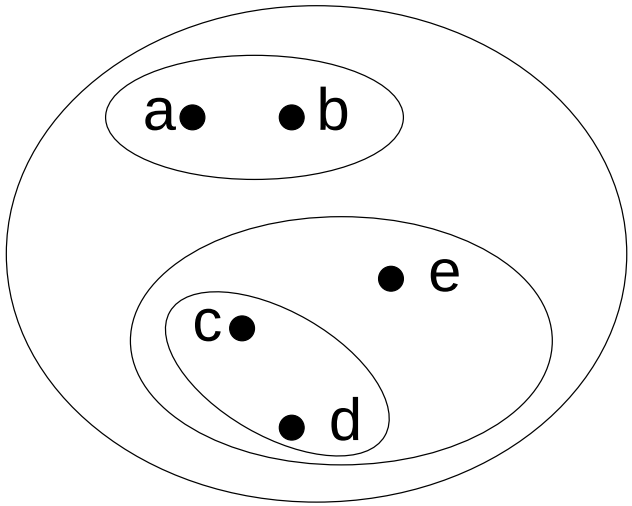
$$d_{\{ab\},\{cd\}} = (d_{ac} + d_{ad} + d_{bc} + d_{bd})/4 = (21 + 32 + 21 + 30)/4$$

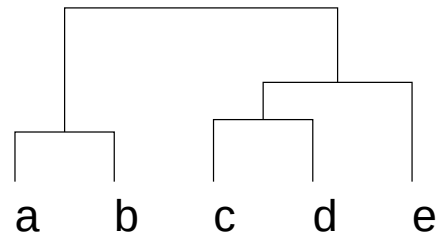




	{a,b}	{c,d,e}
{a,b}	0	24.5
{c,d,e}		0

$$d_{\{ab\},\{cde\}} = (d_{ac} + d_{ad} + d_{ae} + d_{bc} + d_{bd} + d_{bd})/6$$





	{a,b,c,d,e}
{a,b,c,d,e}	0

UPGMA: Distance measure†

Average distance (produces clusters with same variance):

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$$

Complete linkage (produces compact clusters):

$$d_{ij} = \max_{p \in C_i, q \in C_j} d_{pq}$$

Single linkage (picks up elongated/irregular clusters):

$$d_{ij} = \min_{p \in C_i, q \in C_j} d_{pq}$$

† In statistics it is often called hierarchical clustering

Appendix: Java class for the MSA problem

```
public class Alignment {

    private String[] s1, s2;
    private int row, col;
    private int[][] t;
    private char[][] pointers;
    private static final int D = -6; // deletion

    Alignment(String[] s1, String[] s2) {

        this.s1 = s1;
        this.s2 = s2;

        row = s1[0].length()+1; // +1 for the initial conditions
        col = s2[0].length()+1;

        t = new int[row][col];
        pointers = new char[row][col];
    } // ...
```

```
public int fillGlobal() {
    for (int i=0; i<row; i++) { t[i][0] = initRow(i); pointers[i][0] = 'D'; }
    for (int j=1; j<col; j++) { t[0][j] = initCol(j); pointers[0][j] = 'I'; }
    for (int i=1; i<row; i++) {
        for (int j=1; j<col; j++) {
            int del = t[i-1][j] + scoreDel( i );
            int ins = t[i][j-1] + scoreIns( j );
            int diag = t[i-1][j-1] + scoreDiag( i, j );
            pointers[i][j] = 'D';
            int max = del;
            if ( ins > max ) {
                max = ins; pointers[i][j] = 'I';
            }
            if ( diag > max ) {
                max = diag; pointers[i][j] = 'M';
            }
            t[i][j] = max;
        }
    }
    return t[row-1][col-1];
} // ...
```

```
private int scoreDiag(int i, int j) {

    int sum = 0;

    for (int k=0; k<(s1.length-1); k++)
        for (int l=k+1; l<s1.length; l++) {
            char a = s1[k].charAt(i-1); // DP has an extra row
            char b = s1[l].charAt(i-1); // DP has an extra column
            if (a != '-' && b != '-')
                sum += PAM250.score(a,b);
            else if (a != '-' || b != '-')
                sum += D;
            // a == '-' && b == '-' is scored 0
        }
    for (int k=0; k<(s2.length-1); k++)
        for (int l=k+1; l<s2.length; l++) {
            char a = s2[k].charAt(j-1);
            char b = s2[l].charAt(j-1);
            if (a != '-' && b != '-')
                sum += PAM250.score(a,b);
            else if (a != '-' || b != '-')
                sum += D;
        }
    for (int k=0; k<s1.length; k++)
        for (int l=0; l<s2.length; l++) {
            char a = s1[k].charAt(i-1);
            char b = s2[l].charAt(j-1);
            if (a != '-' && b != '-')
                sum += PAM250.score(a,b);
            else if (a != '-' || b != '-')
                sum += D;
        }
    return sum;
} // ...
```

```
private int scoreDel(int i) {
    int sum = 0;
    for (int k=0; k<(s1.length-1); k++)
        for (int l=k+1; l<s1.length; l++) {
            char a = s1[k].charAt(i-1); // DP has an extra row
            char b = s1[l].charAt(i-1); // DP has an extra column
            if (a != '-' && b != '-')
                sum += PAM250.score(a,b);
            else if (a != '-' || b != '-')
                sum += D;
            // a == '-' && b == '-' is scored 0
        }
    for (int k=0; k<s1.length; k++) {
        char a = s1[k].charAt(i-1);
        if (a != '-')
            sum += s2.length * D;
    }
    return sum;
} // ...
```

```
private int scoreIns(int j) {
    int sum = 0;
    for (int k=0; k<(s2.length-1); k++)
        for (int l=k+1; l<s2.length; l++) {
            char a = s2[k].charAt(j-1);
            char b = s2[l].charAt(j-1);
            if (a != '-' && b != '-')
                sum += PAM250.score(a,b);
            else if (a != '-' || b != '-')
                sum += D;
        }
    for (int l=0; l<s2.length; l++) {
        char b = s2[l].charAt(j-1);
        if (b != '-')
            sum += s1.length * D;
    }
    return sum;
} // ...
```

```
private int initRow(int i) {
    int sum = 0;
    for (int p=1; p<=i; p++) {
        for (int k=0; k<(s1.length-1); k++)
            for (int l=k+1; l<s1.length; l++) {
                char a = s1[k].charAt(p-1);
                char b = s1[l].charAt(p-1);
                if (a != '-' && b != '-')
                    sum += PAM250.score(a,b);
                else if (a != '-' || b != '-')
                    sum += D;
            }
        for (int k=0; k<s1.length; k++) {
            char a = s1[k].charAt(p-1);
            if (a != '-')
                sum += s2.length * D;
        }
    }
    return sum;
}  // ...
```

```
private int initCol(int j) {
    int sum = 0;
    for (int p=1; p<=j; p++) {
        for (int k=0; k<(s2.length-1); k++)
            for (int l=k+1; l<s2.length; l++) {
                char a = s2[k].charAt(p-1);
                char b = s2[l].charAt(p-1);
                if (a != '-' && b != '-')
                    sum += PAM250.score(a,b);
                else if (a != '-' || b != '-')
                    sum += D;
            }
        for (int l=0; l<s2.length; l++) {
            char b = s2[l].charAt(p-1);
            if (b != '-')
                sum += s1.length * D;
        }
    }
    return sum;
}  // ...
```

```
public static int sumOfPair( String[] msa ) {
    int sum = 0, row= msa.length, col= msa[0].length();
    for (int p=0; p<col; p++)
        for (int k=0; k<(row-1); k++)
            for (int l=k+1; l<row; l++) {
                char a = msa[k].charAt(p);
                char b = msa[l].charAt(p);
                if (a != '-' && b != '-')
                    sum += PAM250.score(a,b);
                else if (a != '-' || b != '-')
                    sum += D;
                // a == '-' && b == '-' is scored 0
            }
    return sum;
} // ...
```

```
    public static void main(String[] args) {

        String[] s1 = { "YDGGPAVEAL" };
        String[] s2 = { "YDGGPEAL" };

        Alignment msa1 = new Alignment(s1, s2);

        msa1.fillGlobal(); msa1.display(); msa1.displayPointers();

        String[] s3 = { "YDGGPAVEAL",
                        "YDGGP--EAL" };
        String[] s4 = { "FEGGPILVEAL" };

        Alignment msa2 = new Alignment(s3, s4);

        msa2.fillGlobal(); msa2.display(); msa2.displayPointers();

        String[] s5 = { "FDGGILVQAV" };
        String[] s6 = { "YEGGAVVQAL" };

        Alignment msa3 = new Alignment(s5, s6);

        msa3.fillGlobal(); msa3.display(); msa3.displayPointers();

        String[] s7 = { "YDGGPA-VEAL",
                        "YDGGP---EAL",
                        "FEGGPILVEAL" };
        String[] s8 = { "FDGGILVQAV",
                        "YEGGAVVQAL" };

        Alignment msa4 = new Alignment(s7, s8);

        msa4.fillGlobal(); msa4.display(); msa4.displayPointers();
    }
}
```

```
public class PAM250 {

 private static int[][] matrix
 ={
   { 2,-2, 0, 0,-2, 0, 0, 1,-1,-1,-2,-1,-1,-3, 1, 1, 1,-6,-3, 0, 0, 0, 0,-8 },
   {-2, 6, 0,-1,-4, 1,-1,-3, 2,-2,-3, 3, 0,-4, 0, 0,-1, 2,-4,-2,-1, 0,-1,-8 },
   { 0, 0, 2, 2,-4, 1, 1, 0, 2,-2,-3, 1,-2,-3, 0, 1, 0,-4,-2,-2, 2, 1, 0,-8 },
   { 0,-1, 2, 4,-5, 2, 3, 1, 1,-2,-4, 0,-3,-6,-1, 0, 0,-7,-4,-2, 3, 3,-1,-8 },
   {-2,-4,-4,-5,12,-5,-5,-3,-3,-2,-6,-5,-5,-4,-3, 0,-2,-8, 0,-2,-4,-5,-3,-8 },
   { 0, 1, 1, 2,-5, 4, 2,-1, 3,-2,-2, 1,-1,-5, 0,-1,-1,-5,-4,-2, 1, 3,-1,-8 },
   { 0,-1, 1, 3,-5, 2, 4, 0, 1,-2,-3, 0,-2,-5,-1, 0, 0,-7,-4,-2, 3, 3,-1,-8 },
   { 1,-3, 0, 1,-3,-1, 0, 5,-2,-3,-4,-2,-3,-5, 0, 1, 0,-7,-5,-1, 0, 0,-1,-8 },
   {-1, 2, 2, 1,-3, 3, 1,-2, 6,-2,-2, 0,-2,-2, 0,-1,-1,-3, 0,-2, 1, 2,-1,-8 },
   {-1,-2,-2,-2,-2,-2,-2,-3,-2, 5, 2,-2, 2, 1,-2,-1, 0,-5,-1, 4,-2,-2,-1,-8 },
   {-2,-3,-3,-4,-6,-2,-3,-4,-2, 2, 6,-3, 4, 2,-3,-3,-2,-2,-1, 2,-3,-3,-1,-8 },
   {-1, 3, 1, 0,-5, 1, 0,-2, 0,-2,-3, 5, 0,-5,-1, 0, 0,-3,-4,-2, 1, 0,-1,-8 },
   {-1, 0,-2,-3,-5,-1,-2,-3,-2, 2, 4, 0, 6, 0,-2,-2,-1,-4,-2, 2,-2,-2,-1,-8 },
   {-3,-4,-3,-6,-4,-5,-5,-5,-2, 1, 2,-5, 0, 9,-5,-3,-3, 0, 7,-1,-4,-5,-2,-8 },
   { 1, 0, 0,-1,-3, 0,-1, 0, 0,-2,-3,-1,-2,-5, 6, 1, 0,-6,-5,-1,-1, 0,-1,-8 },
   { 1, 0, 1, 0, 0,-1, 0, 1,-1,-1,-3, 0,-2,-3, 1, 2, 1,-2,-3,-1, 0, 0, 0,-8 },
   { 1,-1, 0, 0,-2,-1, 0, 0,-1, 0,-2, 0,-1,-3, 0, 1, 3,-5,-3, 0, 0,-1, 0,-8 },
   {-6, 2,-4,-7,-8,-5,-7,-7,-3,-5,-2,-3,-4, 0,-6,-2,-5,17, 0,-6,-5,-6,-4,-8 },
   {-3,-4,-2,-4, 0,-4,-4,-5, 0,-1,-1,-4,-2, 7,-5,-3,-3, 0,10,-2,-3,-4,-2,-8 },
   { 0,-2,-2,-2,-2,-2,-2,-1,-2, 4, 2,-2, 2,-1,-1,-1, 0,-6,-2, 4,-2,-2,-1,-8 },
   { 0,-1, 2, 3,-4, 1, 3, 0, 1,-2,-3, 1,-2,-4,-1, 0, 0,-5,-3,-2, 3, 2,-1,-8 },
   { 0, 0, 1, 3,-5, 3, 3, 0, 2,-2,-3, 0,-2,-5, 0, 0,-1,-6,-4,-2, 2, 3,-1,-8 },
   { 0,-1, 0,-1,-3,-1,-1,-1,-1,-1,-1,-1,-1,-2,-1, 0, 0,-4,-2,-1,-1,-1,-1,-8 },
   {-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8,-8, 1 }
  };
```

```
 private static int toIndex( char a ) {
     String charIndex = "ARNDCQEGHILKMFPSTWYVBZX";
     int index;
     index = charIndex.indexOf( a );
     if ( index == -1 )
         index = charIndex.length();
     return index;
 }

 public static int score( char a, char b ) {
     return matrix[ toIndex( a ) ][ toIndex( b ) ];
 }

}
```

References

Pensez-y!

L'impression de ces notes n'est probablement pas nécessaire!