



Computational Biology

Krzysztof Giaro
Gdańsk University of Technology
ETI Faculty



Multialignment of sequences set

conservative motifs, proteins families

Multialignment

Q5E940_BOVIN	-MPREDRATWKS	N	I	FLKII	I	Q	L	DDY	P	KCFIVG	ADNY	G	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_HUMAN	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_MOUSE	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_RAT	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_CHICK	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_RANSY	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
Q7ZUG3_BRARE	-MPREDRATWKS	N	I	YFLKII	I	Q	L	DDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_ICTPU	-MPREDRATWKS	N	I	YFLKII	I	Q	L	INDY	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	V	L	MGKNT	MMR	KAIRGHLE																		
RLAO_DROME	-MVRENKAAWAKIAQ	F	I	KVVE	E	L	D	FP	P	KCFIVG	ADNVGS	K	S	KOMO	QIRMS	L	RGK-	A	I	V	LGKNT	MMR	KAIRGHLE																		
RLAO_DICDI	-MSGAG	S	KKRKKI	L	F	I	E	KATKL	F	T	YDKMIV	A	E	ADFVG	S	SOLOKIEKS	I	RGI-	G	A	V	LGKNT	MMR	KAIRGHLE																	
Q5A1P0_DICDI	-MSGAG	S	KKRKKI	L	F	I	E	KATKL	F	T	YDKMIV	A	E	ADFVG	S	SOLOKIEKS	I	RGI-	G	A	V	LGKNT	MMR	KAIRGHLE																	
RLAO_PLAF8	-MAKLSKQOKQOM	I	YTKLSSL	I	QQYSKIL	I	L	WHDN	V	SNOMAS	VEKS	L	RGK-	A	T	I	LGKNT	R	I	T	ALKKNL	Q	ALKKNL																		
RLAO_SULAC	-MIGLAVITTTKK	I	AKKKV	I	D	S	V	AELT	E	KL	KTTHKT	I	I	ANIE	G	FPADKLHE	I	RK	K	RGK-	ADIKV	T	KKNNLFN	I	ALKNAG																
RLAO_SULTO	-MRIMAVITQERK	I	AKKKV	I	E	S	V	KELLE	E	QK	LREYHT	I	I	ANIE	G	FPADKLHD	I	RK	K	RGK-	AEIKV	K	NLFC	I	ALKNAG																
RLAO_SULSO	-MKRLALALLKQRKV	V	ASWQKIE	E	S	V	KELLE	E	QK	LREYHT	I	I	ANIE	G	FPADKLHE	I	RK	K	RGK-	ATIKV	K	NLFC	I	ALKNAG																	
RLAO_AERPE	-MSVVSILVQOMYKRE	K	P	FEWKL	I	M	R	RELE	L	ESKRHV	LF	ADLTG	G	P	I	F	V	V	R	WRK-	Y	PMV	AKR	I	IL	RAMKAAGL															
RLAO_PYRAE	-MMLATGKRYVYRT	RQ	I	PKV	I	V	S	TEATELL	I	QK	Y	YVFL	F	D	L	H	G	L	S	R	I	L	H	E	Y	R	RRY-														
RLAO_METAC	-MAEERHHTET	I	PQWPK	D	E	IEN	I	KEI	I	QSHHKV	F	GMV	G	IE	G	I	LAT	K	MOK	I	IRD	L	DY	-AVI	K	V	S	RNTL	T	RALNQLG											
RLAO_METMA	-MAEERHHTET	I	PQWPK	D	E	IEN	I	KEI	I	QSHHKV	F	GMV	G	IE	G	I	LAT	K	MOK	I	IRD	L	DY	-AVI	K	V	S	RNTL	T	RALNQLG											
RLAO_ARCFU	-MAAVQRES	I	P	FEVKYV	R	E	V	E	I	KR	I	S	SKPV	V	A	V	S	FRN	V	PA	G	Q	O	M	O	I	E	RF	RGK-	AEIKV	V	K	NLFC	I	RALDALG						
RLAO_METKA	-MAVKAKGQPSPSYE	P	PKVAEWKIE	R	E	V	E	K	E	V	E	I	KEV	K	L	E	D	Y	E	N	V	G	L	V	D	L	E	I	POLO	E	IRAKL	ERD	I	I	RVMSRN	T	LMR	I	ALEEKLD		
RLAO_METTH	-MAHVAEWKIE	R	E	V	E	K	E	V	E	I	K	E	V	E	I	K	L	H	D	I	C	G	Y	E	V	G	I	ANLAD	I	PAOLQ	M	KEQT	L	RDS	-ALIRMSKK	I	L	ISLALEKAGR			
RLAO_METTL	-MITASEEHKIAV	K	IE	E	V	N	V	KL	E	LL	KG	Q	I	V	I	L	V	D	M	M	E	V	PA	O	L	O	E	I	R	D	K	-G	F	MTL	KMSRN	T	L	I	E	RAIKEV	A
RLAO_METVA	-MIDAKSEHKIAV	K	IE	E	V	N	V	KL	E	LL	KG	Q	I	V	I	L	V	D	M	M	E	V	PA	O	L	O	E	I	R	D	K	-G	F	MTL	KMSRN	T	L	I	E	RAIKEV	A
RLAO_METJA	-METKVKAHV	V	A	E	V	E	V	E	E	V	E	E	V	E	V	E	I	K	L	H	D	I	C	G	Y	E	V	G	I	ANLAD	I	PAOLQ	M	KEQT	L	RDS	-ALIRMSKK	I	L	ISLALEKAGR	
RLAO_PYRAB	-MAHVAEWKIE	R	E	V	E	E	V	E	E	V	E	E	V	E	V	E	I	K	L	H	D	I	C	G	Y	E	V	G	I	ANLAD	I	PAOLQ	M	KEQT	L	RDS	-ALIRMSKK	I	L	ISLALEKAGR	
RLAO_PYRHO	-MAHVAEWKIE	R	E	V	E	E	V	E	E	V	E	E	V	E	V	E	I	K	L	H	D	I	C	G	Y	E	V	G	I	ANLAD	I	PAOLQ	M	KEQT	L	RDS	-ALIRMSKK	I	L	ISLALEKAGR	

Idea: symbols descended from common antecessor letter should occupy the same column

Let be given sequences $u_1, \dots, u_r \in \Sigma^*$. Their **multialignment** are

sequences $u_1^*, \dots, u_r^* \in \Sigma^{**}$ fulfilling:

- ▶ for $i=1, \dots, r$ deletion of spaces from u_i^* results in u_i ,
- ▶ all the lengths $|u_i^*|$ are the same,
- ▶ there are no columns with spaces only: $\forall_{i \leq |u_i^*|} \exists_j u_j^*[i] \neq -$

Multialignments – not only simple generalization

... One or two homologous sequences whisper ... a full multiple alignment shouts out loud.

Lesk

- ▶ Detection of distant relationship between proteins or genes.

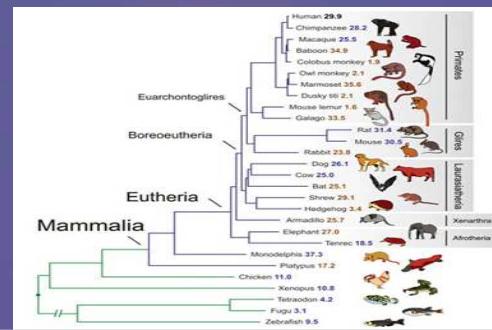
Evolutionary or functionally related molecular strings can differ significantly throughout much of the string and yet preserve the same three-dimensional structure(s), or ... motifs, domains, or the same active sites...

Example. Hemoglobins of mammals and insects

- ▶ about 140 amino acids,
- ▶ diverged ~600 mln. years ago,
- ▶ about 100 letters' substitution in each evolutionary line.

Multialignments – not only simple generalization

- Ability of describing the whole families of proteins or genes, functionally related, with a common history or similar spatial configuration. Evaluation of new candidates for family members.
- Detection of characteristic important patterns in DNA, structural motifs in protein sequence or fragments crucial for their function.
- Source of information about evolutionary history of genes, proteins or their species.



Multialignment quality function

Try to treat multialignment like 2-sequence alignment.

For multialignment u_1^*, \dots, u_r^* and **column score function** $\rho: \Sigma^r \rightarrow R$ the **multialignment score** is the scores' sum for all columns:

$$\rho^*(u_1^*, \dots, u_r^*) = \sum_{i=1, \dots, |u_1^*|} \rho(u_1^*[i], \dots, u_r^*[i])$$

- ρ should give „good” values for columns with similar letters,
- multialignment for $u_1, \dots, u_r \in \Sigma^*$ is **optimal** if its score is the smallest possible (or the greatest – in similarity-like approach).

Example. Column functions based on 2-letter distance or similarity.

- SP-measure:** sum of distances (or similarities) for every pair of letters in a column: $\rho_{SP}(a_1, \dots, a_r) = \sum_{1 \leq i < j \leq r} d(a_i, a_j)$.
 \Rightarrow For $u_1^*, \dots, u_r^* \in \Sigma^*$ we have $\rho_{SP}^*(u_1^*, \dots, u_r^*) = \sum_{1 \leq i < j \leq r} d(u_i^*, u_j^*)$
- Consensus error measure:** for each column find a consensus letter, that minimizes the sum of distances: $\rho_{CONS}(a_1, \dots, a_r) = \min_{c \in \Sigma} \sum_{i=1, \dots, r} d(a_i, c)$

Calculating optimal multialignment

Is dynamic programming approach possible? For words $u_1, \dots, u_r \in \Sigma^*$ and all prefixes $u_1[1 \dots i_1], \dots, u_r[1 \dots i_r]$ store in r -dimensional array their multialignment optimal score

$$D(i_1, \dots, i_r) = \rho^*(u_1[1 \dots i_1], \dots, u_r[1 \dots i_r])$$

Recursion: $D(i_1, \dots, i_r) =$

$$= \min_{\Delta_1, \dots, \Delta_r \in \{0,1\}, \Sigma \Delta > 0} D(i_1 - \Delta_1, \dots, i_r - \Delta_r) + \rho(\Delta_1 \bullet u_1[i_1], \dots, \Delta_r \bullet u_r[i_r])$$

– where $1 \bullet a = a$ and $0 \bullet a = '-'$ for $a \in \Sigma$.

Space complexity $O(|u_1| \cdot \dots \cdot |u_r|)$ and time cost $O(2^r |u_1| \cdot \dots \cdot |u_r|)$ – both exponential! Applicable for small number of sequences only.

Remind NP-hard *Longest Common Subsequence* problem: a given set of words $R = \{u_1, \dots, u_r\} \subseteq \Sigma^+$. We search for the longest possible word $w = w_1 \dots w_k \in \Sigma^*$, that $\forall_{u \in R} u = x_0 w_1 x_1 w_2 \dots w_k x_k \quad x_i \in \Sigma^*$.

Calculating optimal multialignment

Consider column score function $\rho(a_1, \dots, a_r)$:

- the same letter case: $\rho(a, \dots, a) = 1$ – for every $a \in \Sigma$
- all other possibilities : $\rho(a_1, \dots, a_r) = 0$

\Rightarrow in optimal (ρ^* -maximum) multialignment of u_1, \dots, u_r successive letters of some LCS fill different columns.

$\Rightarrow \rho^*(u_1, \dots, u_r) = \text{LCS's length.}$

.....	A	..	C	..	T
.....	A	..	C	..	T
.....	A	..	C	..	T
.....	A	..	C	..	T

\Rightarrow we have a reduction of LCS problem to optimum multialignment problem. Thus finding optimum multialignment is NP-hard.

Theorem. Optimum multialignment problem remains NP-hard even for SP- and consensus error column score functions.

In practice: only heuristics are applicable.

Multialignment profile

Abbreviated information from multialignment

- makes possible an evaluation of candidate sequences for new alignment elements,
- eventually shows how to extend the multialignment by a new one.

For multialignment $u_1^*, \dots, u_r^* \in \Sigma^{*}$

- **consensus word:** for each column i one consensus letter:

$$c_i = \operatorname{argmin}_{c \in \Sigma^*} \sum_{j=1, \dots, r} d(c, u_j^*[i])$$

Example. Words $u_1 = \text{'ABCA'}$, $u_2 = \text{'ABABA'}$, $u_3 = \text{'ACCB'}$, $u_4 = \text{'CBBC'}$, discrete metric d , alignment:

$$u_1^* = \text{ABC-A}$$

$$u_2^* = \text{ABABA}$$

$$u_3^* = \text{ACCB-}$$

$$u_4^* = \text{CB-BC}$$

$$\text{ABCBA}$$

Multialignment profile

More information contains the **profile** of multialignment u_1^* ,

$\dots, u_r^* \in \Sigma^{*}$ – for each column i we have the observed probability $p_i(c)$ of symbols $c \in \Sigma^*$ placed in this column.

Example. Words $u_1 = \text{'ABCA'}$, $u_2 = \text{'ABABA'}$, $u_3 = \text{'ACCB'}$, $u_4 = \text{'CBBC'}$.

$$12345$$

$$u_1^* = \text{ABC-A}$$

$$u_2^* = \text{ABABA}$$

$$u_3^* = \text{ACCB-}$$

$$u_4^* = \text{CB-BC}$$

Profile:

	1	2	3	4	5
A	0.75	0	0.25	0	0.5
B	0	0.75	0	0.75	0
C	0.25	0.25	0.50	0	0.25
-	0	0	0.25	0.25	0.25

Profile alignment

Given profile p_1, \dots, p_L and a new word $v \in \Sigma^*$ we can define their 2-alignment similarly like alignment of two words, but this time indices of profile's columns stay instead of symbols. That alignment gives a way of extending profiled multialignment by a word v (just put whole columns or columns of spaces in place of indices in the profile line).

Example. Words $u_1 = \text{ABCA}$, $u_2 = \text{ABABA}$, $u_3 = \text{ACCB}$, $u_4 = \text{CBBC}$.

Multialignment:

$$u_1^* = \text{ABC-A}$$

Multialignment extended by v :

$$u_2^* = \text{ABABA}$$

$$v^* = \text{AAB-BC}$$

$$u_3^* = \text{ACCB-}$$

$$\text{A-BC-A}$$

$$u_4^* = \text{CB-BC}$$

$$\text{A-BABA}$$

New word $v = \text{AABBC}$ and its alignment with the profile:

$$\text{A-CCB-}$$

$$v^* = \text{AAB-BC}$$

$$\text{C-B-BC}$$

$$1-2345$$

Profile alignment

Problem. How to define and find efficiently the „best” alignment of word and profile?

Having similarity function for letters $s: \Sigma' \times \Sigma \rightarrow R$ we „extend” it for letter-column pairs ($a \in \Sigma'$ and p_i) taking letters similarities weighted with their probabilities:

$$s(a, p_i) = \sum_{b \in \Sigma} s(a, b) p_i(b)$$

Now the greatest total similarity alignment of word and profile can be found like maximum similarity alignment of two sequences.

Example. $v = \text{AABBC}$, profile:

	1	2	3	4	5
A	0.75	0	0.25	0	0.5
B	0	0.75	0	0.75	0
C	0.25	0.25	0.50	0	0.25
-	0	0	0.25	0.25	0.25

and their alignment: AAB-BC
 $1-2345$

Alignment similarity:

$$[0.75s(\text{A}, \text{A}) + 0.25s(\text{A}, \text{C})] +$$

$$s(\text{A}, -) +$$

$$[0.75s(\text{B}, \text{B}) + 0.25s(\text{B}, \text{C})] +$$

$$[0.25s(-, \text{A}) + 0.5s(-, \text{C}) + 0.25s(-, -)] +$$

$$[0.75s(\text{B}, \text{B}) + 0.25s(\text{B}, -)] +$$

$$[0.5s(\text{C}, \text{A}) + 0.25s(\text{C}, \text{C}) +$$

$$+ 0.25s(\text{C}, -)]$$

Profile alignment

Analogously for two multialignments and their profiles p_1, \dots, p_L and q_1, \dots, q_M we can:

- ▶ define a profile-profile alignment with columns indices and spaces in both rows
 - ▶ define mean similarity of their columns based on similarity function of letters from Σ' :
- $$s(p_i, q_j) = \sum_{a \in \Sigma'} \sum_{b \in \Sigma'} s(a, b) p_i(a) q_j(b)$$
- ▶ use this measure for efficient search for the „best” alignment of these profiles,
 - ▶ merge two multialignments into one by aligning their profiles first.

Profile alignment

Example. Words $u_1 = \text{TAG}$, $u_2 = \text{CAT}$, $u_3 = \text{TG}$ **ALIGNMENT:**

$v_1 = \text{TC}$, $v_2 = \text{AGG}$, $v_3 = \text{ATC}$.

$u_1^* = \text{TAG}$

$v_1^* = \text{TC-}$

1-23

$u_2^* = \text{CAT}$

$v_2^* = \text{AGG}$

12-3

$u_3^* = \text{T-G}$

$v_3^* = \text{ATC}$

PROFILE I:

	1	2	3
A	0	0.66	0
G	0	0	0.66
T	0.66	0	0.33
C	0.33	0	0
-	0	0.33	0

PROFILE II:

	1	2	3
A	0.66	0	0
G	0	0.33	0.33
T	0.33	0.33	0
C	0	0.33	0.33
-	0	0	0.33

Alignment similarity

$$s(p_i, q_j) = \sum_{a \in \Sigma} \sum_{b \in \Sigma} s(a, b) p_i(a) q_j(b)$$

$$\begin{aligned} I: & s(T, T) * 0.66 * 0.33 + \\ & s(C, T) * 0.33 * 0.33 + \\ & s(T, A) * 0.66 * 0.66 \end{aligned}$$

Profile alignment

Example. Words $u_1 = \text{TAG}$, $u_2 = \text{CAT}$, $u_3 = \text{TG}$ **ALIGNMENT:**
 $v_1 = \text{TC}$, $v_2 = \text{AGG}$, $v_3 = \text{ATC}$.

$$u_1^* = \text{TAG}$$

$$u_2^* = \text{CAT}$$

$$u_3^* = \text{T-G}$$

PROFILE I:

	1	2	3
A	0	0.66	0
G	0	0	0.66
T	0.66	0	0.33
C	0.33	0	0
-	0	0.33	0

$$v_1^* = \text{TC-}$$

$$v_2^* = \text{AGG}$$

$$v_3^* = \text{ATC}$$

1-23

12-3

PROFILE II:

	1	2	3
A	0.66	0	0
G	0	0.33	0.33
T	0.33	0.33	0
C	0	0.33	0.33
-	0	0	0.33

Alignment similarity

$$s(p_i q_j) = \sum_{a \in \Sigma} \sum_{b \in \Sigma} s(a, b) p_i(a) q_j(b)$$

$$\text{II: } s(-, C) * 1 * 0.33 + \\ s(-, G) * 1 * 0.33 + \\ s(-, T) * 1 * 0.33$$

Profile alignment

Example. Words $u_1 = \text{TAG}$, $u_2 = \text{CAT}$, $u_3 = \text{TG}$ **ALIGNMENT:**

$$v_1 = \text{TC}, v_2 = \text{AGG}, v_3 = \text{ATC}.$$

$$u_1^* = \text{TAG}$$

$$u_2^* = \text{CAT}$$

$$u_3^* = \text{T-G}$$

PROFILE I:

	1	2	3
A	0	0.66	0
G	0	0	0.66
T	0.66	0	0.33
C	0.33	0	0
-	0	0.33	0

$$v_1^* = \text{TC-}$$

$$v_2^* = \text{AGG}$$

$$v_3^* = \text{ATC}$$

1-23

12-3

PROFILE II:

	1	2	3
A	0.66	0	0
G	0	0.33	0.33
T	0.33	0.33	0
C	0	0.33	0.33
-	0	0	0.33

Alignment similarity

$$s(p_i q_j) = \sum_{a \in \Sigma} \sum_{b \in \Sigma} s(a, b) p_i(a) q_j(b)$$

$$\text{III: } s(A, -) * 0.66 * 1 + \\ s(-, -) * 0.33 * 1$$

Profile alignment

Example. Words $u_1 = \text{'TAG'}$, $u_2 = \text{'CAT'}$, $u_3 = \text{'TG'}$ **ALIGNMENT:**
 $v_1 = \text{'TC'}$, $v_2 = \text{'AGG'}$, $v_3 = \text{'ATC'}$.

$$u_1^* = \text{TAG}$$

$$u_2^* = \text{CAT}$$

$$u_3^* = \text{T-G}$$

PROFILE I:

	1	2	3
A	0	0.66	0
G	0	0	0.66
T	0.66	0	0.33
C	0.33	0	0
-	0	0.33	0

$$v_1^* = \text{TC-}$$

1-23

$$v_2^* = \text{AGG}$$

12-3

$$v_3^* = \text{ATC}$$

PROFILE II:

	1	2	3
A	0.66	0	0
G	0	0.33	0.33
T	0.33	0.33	0
C	0	0.33	0.33
-	0	0	0.33

$$\text{IV: } s(G, -) * 0.66 * 0.33 + \\ s(G, G) * 0.66 * 0.33 + \\ s(G, C) * 0.66 * 0.33 + \\ s(T, -) * 0.33 * 0.33 + \\ s(T, G) * 0.33 * 0.33 + \\ s(T, C) * 0.33 * 0.33$$

Alignment similarity

$$s(p_i, q_j) = \sum_{a \in \Sigma} \sum_{b \in \Sigma} s(a, b) p_i(a) q_j(b)$$

Profile alignment

Example. Words $u_1 = \text{'TAG'}$, $u_2 = \text{'CAT'}$, $u_3 = \text{'TG'}$ **ALIGNMENT:**

$$v_1 = \text{'TC'}$$

$$u_1^* = \text{TAG}$$

1-23

$$u_2^* = \text{CAT}$$

12-3

$$u_3^* = \text{T-G}$$

PROFILE I:

	1	2	3
A	0	0.66	0
G	0	0	0.66
T	0.66	0	0.33
C	0.33	0	0
-	0	0.33	0

$$v_1^* = \text{TC-}$$

1-23

$$v_2^* = \text{AGG}$$

12-3

$$v_3^* = \text{ATC}$$

PROFILE II:

	1	2	3
A	0.66	0	0
G	0	0.33	0.33
T	0.33	0	0
C	0	0.33	0.33
-	0	0	0.33

Alignment similarity

$$s(p_i, q_j) = \sum_{a \in \Sigma} \sum_{b \in \Sigma} s(a, b) p_i(a) q_j(b)$$

$$u_1^* = \text{T-AG}$$

33 0

$$u_2^* = \text{C-AT}$$

33 0.33

$$u_3^* = \text{T--G}$$

0.33

$$v_1^* = \text{TC--}$$

33

$$v_2^* = \text{AG-G}$$

0.33

$$v_3^* = \text{AT-C}$$

0.33

Progressive multialigning methods

Heuristic algorithms that construct efficiently „good” multialignments of given sequences based on similarity function $s: \Sigma^* \rightarrow R$. Each step input sequences are merged into disjoint clusters, and sequences in one cluster are already multialigned.

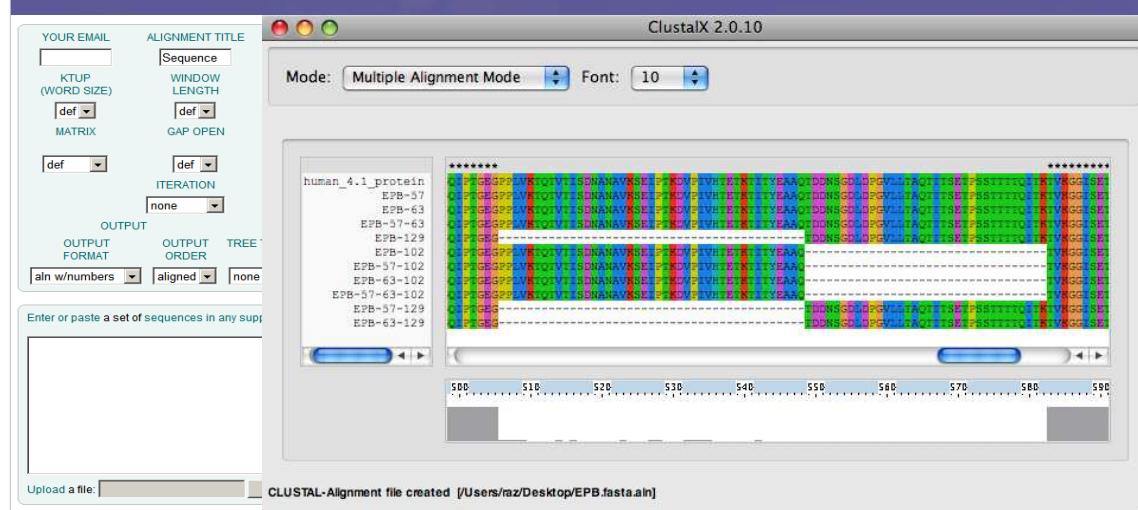
1. In the beginning, each word is a 1-element cluster.
2. At each stage choose two different clusters to merge into one new, combining their multialignments by their profiles alignment.

various strategies possible: choose clusters having two most similar sequences, with max. mean sequences similarity, with max. profiles similarity ...

3. Stop when one cluster with all sequences is obtained.
... or use the iterative improvement: delete one sequence from multialignment, try to align it better with the rest multialignment, continue ...

CLUSTAL: the most popular multialignment tool

CLUSTAL uses heuristic approach based on progressive aligning.

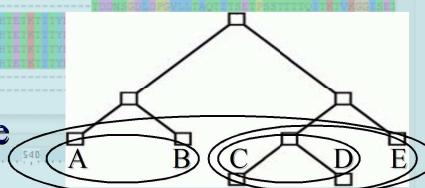


CLUSTAL: the most popular multialignment tool

CLUSTAL uses heuristic approach based on progressive aligning.

Guide tree method:

1. Count similarity score for each pair of input sequences.
2. Find distances between every two sequences (using similarities S e.g. *Feng & Doolittle* distance: $d = -100 \ln [(S - S_{\text{rand}})/(S_{\text{ident}} - S_{\text{rand}})]$ or other evolutionary distance).
3. Use distances and some agglomerative (hierarchical) clustering to build a rooted tree with sequences in leaves.
4. Create progressive multialignment by merging clusters in order appropriate to guide tree vertices processed in bottom-up manner.



CLUSTAL: the most popular multialignment tool

CLUSTAL uses heuristic approach based on progressive aligning.

Other CLUSTAL improvements:

- additional sequences weights deprecate closely related strings contribution into profiles,
- similarity function matrix is chosen automatically according to observed similarity of aligned sequences,
- during progressive alignment gap penalty decreases when merged alignments already have gaps,
- if the score of progressive alignment is low, the guide tree may be corrected on the fly.

CLUSTAL-Alignment file created [/Users/raz/Desktop/EPB.fasta.aln]

PROSITE – protein families database

PROSITE collects biologically significant sequential patterns obtained from amino acid sequences multialignments.



- functional amino acid patterns,
- protein domains,
- protein families characterized by conservative motifs.

Ability of detecting patterns in given amino acid sequences.

Two kinds of records describing patterns:

- profiles,
- regular expressions of special format.



PROSITE – protein families database

PROSITE pattern notation:

- **-** – separator between the pattern's elements,
- **V** – any letter, one letter amino acid code,
- **x** – any amino acid,
- **[...]** – one amino acid from bracket,
- **{...}** – one amino acid, but not from bracket,
- **e(i)** – for element *e* and number *i*: repetition of *e* exactly *i* times,
- **e(i,j)** – repetition of *e* exactly *k* times, where $k \geq i$ and $k \leq j$.



Example. Pattern of some RNA-binding proteins' family:

[RK]-G-{EDRKHPCG}-[AGSCI]-[FY]-[LIV]-x-[FYM]

Fragment of multialignment:

- SRSLKMR**GQAFVIFKEVSSAT**
- KLTGRP**RGVAFVRYNKREEAQ**
- VGCSVH**KGEAEVOYVNERNAR**

PROSITE – protein families database

Example. PROSITE pattern description.

The screenshot shows the PROSITE pattern description for PDOC00168. The pattern is defined as: **[GDN] - x(2) - [LIVF] - x(3) - (VH) - (M) - [LVMFCA] - x(2) - [LVMFCA] - (L - [STAIVQDN] - x(2) - [LVMFSA] - x(5) - [GCN] - x - [LVMFY])**. The page also includes a note about the H-protein of the glycine cleavage system (GCS) and a link to a signature pattern for the lipoyl-binding site.

PROSITE – protein families database

The screenshot shows the search results for PS00189. It lists several entries, including ACOC_PSEPU/29-58, ACOC_RALEH/34-63, ADEC_LACS1/482-511, APT_GRAFK/138-167, ARGO_ECO57/162-191, ARGO_ECOL6/162-191, ARGO_ECOLI/162-191, ARGO_SHIFL/162-191, CHTP1_BOVIN/268-297, CHTP1_DANRE/257-286, CHTP1_HUMAN/268-297, CHTP1_MOUSE/268-297, CHTP1_RAT/268-297, CHTP1_XENLA/268-297, CII17_XENTR/450-479, COPC_ARATH/154-183, CPX1_BACME/296-325, CSPG4_MOUSE/126-155, FBP1L_HUMAN/528-557, FBP1L_MOUSE/528-557, FBP1L_RAT/528-557, FNBP1_XENLA/533-562, GATA_RHIEC/42-71, GCSH1_ARATH/81-110, GCSH1_DICDI/66-95, GCSH1_GULAC/55-84, GCSH1_PHEIN/47-76, GCSH2_AQLAC/59-80, GCSH2_PSEAE/49-78, and GCSH2_PSEPU/49-78. The detailed view of the first entry shows the consensus pattern and sequences from various organisms.