

Chapter 3

Multiple Sequence Alignments

```

      *           :           *           : : :
Q5E940_BOVIN  -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
RLA0_HUMAN   -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
RLA0_MOUSE   -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
  RLA0_RAT    -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
RLA0_CHICK   -----MPREDRATWKSNYFMKIIQLLDDYPKCFVVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
RLA0_RANSY   -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--SALE 76
Q7ZUG3_BRARE -----MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKOMQOIRMSLRGK-AVVLGMGKNTMMRKAIRGHLENN--PALE 76
RLA0_ICTPU   -----MPREDRATWKSNYFLKIIQLLNDYPKCFIVGADNVGSKOMQOIRMSLRGK-AIVLMGKNTMMRKAIRGHLENN--PALE 76
RLA0_DROME   -----MVRENKAAWKAQYFIKVVLEFDFPKCFIVGADNVGSKOMQOIRMSLRGL-AVVLGMGKNTMMRKAIRGHLENN--PQLE 76
RLA0_DICDI   -----MSCAG-SKRKRLFIEKATKLFITYDKMIVAEADVFVSSQLQKIRKSIRGI-GAVLMGKNTMIRKVIRDLADSK--PELD 75
Q54LP0_DICDI -----MSCAG-SKRKNVFIEKATKLFITYDKMIVAEADVFVSSQLQKIRKSIRGI-GAVLMGKNTMIRKVIRDLADSK--PELD 75
RLA0_PLAF8   -----MAKLSKQKKQMYIEKLSLIIQQYSKILIVHVDNVGNSQMASVRKSLRGK-ATILMGKNTIRRTALKKNLQAV--PQIE 76
RLA0_SULAC   -----MIGLAVTTTKIAKWKVDEVAELTEKLRHTKTIIANIEGFPADKLHEIRKCLRGL-ADIKVTKNNLNFNIALKNAG-----YDTK 79
RLA0_SULTO   -----MRIMAVITQERKIAKWKIEEVKELEQLREYHTIIANIEGFPADKLHDIRKMRGM-AEIKVTKNTLFGIAAKNAG-----LDVS 80
RLA0_SULSO   -----MKRLALALKQRKVASWKLVEKLELTKIKNSNTILIGNLEGFPADKLHEIRKCLRGL-ATIKVTKNTLFGIAAKNAG-----IDIE 80
RLA0_AERPE   MSVVSIVGOMYKREKIPWKTLMLELEELFSKHRVFLFADLTGTPTFVVRVRKKLWKK-YPMVAKKRIILRAMKAAGLE---LDDN 86
RLA0_PYRAE   -MMLAIGKRRYVRTROYPAKRVKIVSEATELLQKYPYVFLFDLHGLSSRILHEYRYRLRY-GVIKIKPTLFLKIAFTKVYGG---IPAE 85
RLA0_METAC   -----MAEERHTEHIPQWKKDEIENIKELIQSHKVFQGMVIEGILATKMQKIRRDLDV-AVLKVSRTNLTTERALNQLG-----ETIP 78
RLA0_METMA   -----MAEERHTEHIPQWKKDEIENIKELIQSHKVFQGMVIEGILATKIQKIRRDLDV-AVLKVSRTNLTTERALNQLG-----ESIP 78
RLA0_ARCFU   -----MAAVRGS---PPEYKVRAVEEIKRMISSEKPVVAIVSFRNVPAGOMQKIRREFRGK-AEIKVVKNTLLEALDAGL-----GDYL 75
RLA0_METKA   MAVKAKGQPPSYEPKVAEWKRREVEKELKELMDEYENVGLVDLEGIPAPQLQEIIRAKLRERDTIIRMSRNTLMRIAEEKLDER--PELE 88
RLA0_METTH   -----MAHVAEWKKKEVQELHDLIKGYEVVGIANLADIPARQLQKMRQTLRDS-ALIRMSKKTLLISLAEKAGREL--ENVV 74
RLA0_METTL   -----MITAESEHKIAPWKIEEVNKLKELLKNGQIVALVDMMEVPAVQLOEIRDKIR-GTMTLKMSRNTLIERAIKEVAEETGNDEFA 82
RLA0_METVA   -----MIDAKSEHKIAPWKIEEVNALKELLKSNVIALIDMMEVPAVQLOEIRDKIR-DQMTLKMSRNTLIKRAVEEVAEETGNDEFA 82
RLA0_METJA   -----METKVKAHVAPWKIEEVKTLKGLIKSKPVVAIVDMDDVPAPQLQEIIRDKIR-DKVKLRMSRNTLIERALKEAAEELNPKLA 81
RLA0_PYRAB   -----MAHVAEWKKKEVEELANLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGLLRVSRTNLTIELAIKKAQELGKPELE 77
RLA0_PYRHO   -----MAHVAEWKKKEVEELAKLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGLLRVSRTNLTIELAIKKAQELGKPELE 77
RLA0_PYRFU   -----MAHVAEWKKKEVEELANLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGLLRVSRTNLTIELAIKKAQELGKPELE 77
RLA0_PYRKO   -----MAHVAEWKKKEVEELANLKSYPVIALVDVAGVPAYPLSKMRDKLR-GKALLRVSRTNLTIELAIKRAQELGQPELE 76
RLA0_HALMA   MSAESERKTETIPWKKQEEVDAIVEMIESYESVGVVNIAGIPSRQLODMRRDLHGT-AELRVSRTNLTLEALDDVD-----DGLE 79
RLA0_HALVO   MSESEVRQTEVIPQWKKREVEDELVDVIESYESVGVVNIAGIPSRQLODMRRDLHGS-AAVRMSRNTLVNRAALDEVN-----DGFE 79
RLA0_HALSA   MSAEQRTTEEVPEWKRQEVDELVDLLETYDSVGVVNIAGIPSRQLODMRRDLHGQ-AAALRMSRNTLLVRALEEAG-----DGLD 79
RLA0_THEAC   -----MKEVSQKKKELVNEITQRIKASRSVAIVDTAGIRTRQIQDIRGKNRGK-INLKVIKKTLFLKALENLGD-----EKLS 72
RLA0_THEVO   -----MRKINPKKKEIVSELAODITKSKAVAVDIKGVTRQMODIRAKNRDK-VKIKVVKKTLFLKALDSIND-----EKLT 72
RLA0_PICTO   -----MTEPAQWKIDFVKNLENEINSRKAIVSIVKGLRNNFQKIRNSIRDK-ARIKVSRRARLLRLAIENIGK-----NNIV 72
ruler 1.....10.....20.....30.....40.....50.....60.....70.....80.....90

```

Multiple alignment algorithms

Definition. A multiple alignment of sequences X^1, \dots, X^n is a series of gapped sequences $\tilde{X}^1, \dots, \tilde{X}^n$ such that

- (i) \tilde{X}^i is an extension of X^i obtained by insertions of spaces;
- (ii) $|\tilde{X}^1| = |\tilde{X}^2| = \dots = |\tilde{X}^n|$.

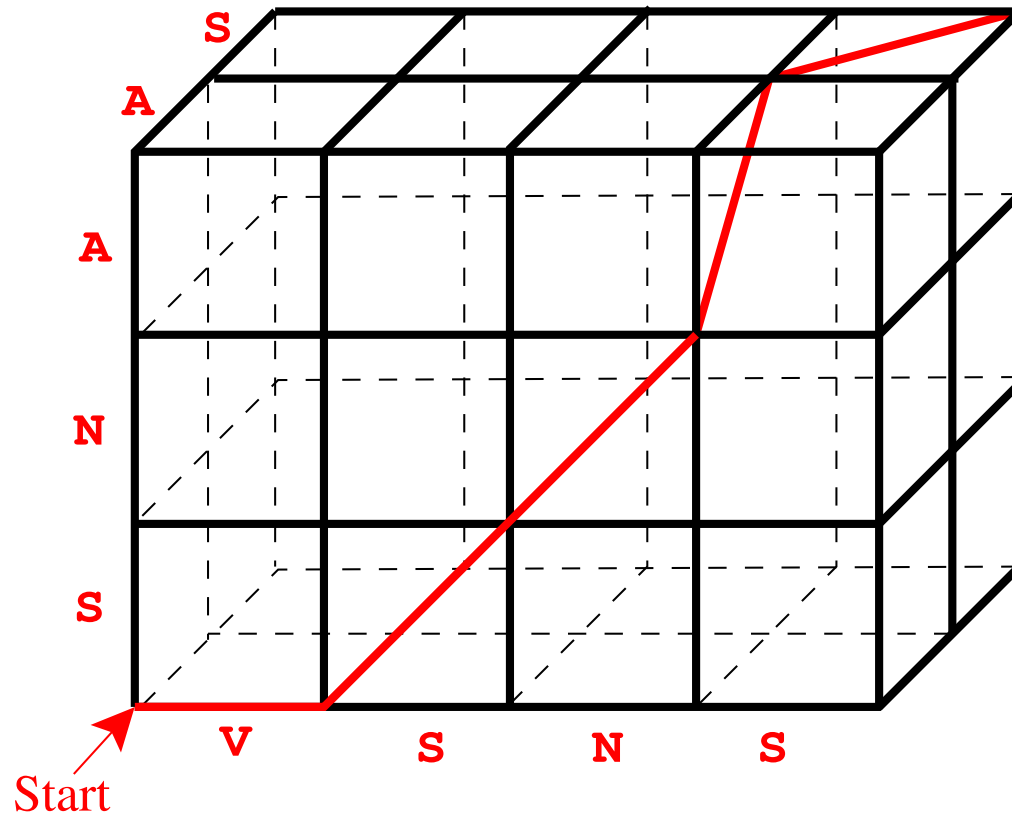
Why are we interested in multiple alignments?

- A multiple alignment **carries more information** than a pairwise one, as a protein can be matched against a family of proteins instead of only against another one.
- Multiple similarity of (protein) sequences suggests
 - a **common structure**,
 - a **common function**,
 - a **common evolutionary source**.

The alignment hyper-cube

Best multiple alignment of r sequences:

Best path through r -dimensional hyper-cube D .



V S N _ S
 _ S N A _
 _ _ _ A S

1 1 1 0 1
 0 1 1 1 0
 0 0 0 1 1

→ ϵ_1
 ↑ ϵ_2
 ↗ ϵ_3

Alignment path for three example sequences.

Dynamic Programming Solution

- Best multiple alignment of r sequences:
Best path through r -**dimensional hyper-cube**.
- Define $S(j_1, j_2, \dots, j_r)$ as the best score for aligning the prefixes of lengths j_1, j_2, \dots, j_r of the sequences X^1, X^2, \dots, X^r .
- We define $S(0, 0, \dots, 0) = 0$, and we calculate

$$S(j_1, j_2, \dots, j_r) = \max_{(\epsilon_1, \dots, \epsilon_r): \epsilon_i \in \{0, 1\}, \epsilon \neq 0} \left[S(j_1 - \epsilon_1, j_2 - \epsilon_2, \dots, j_r - \epsilon_r) + s(\epsilon_1 x_{j_1}, \dots, \epsilon_r x_{j_r}) \right],$$

where s is the **scoring function** (example $s(a, b, 0)$: joint score for aligning characters a, b and a gap) and

$\epsilon = (\epsilon_1, \dots, \epsilon_r)$ is a binary vector that indicates the **directions** of the alignment progress in the hyper-cube.

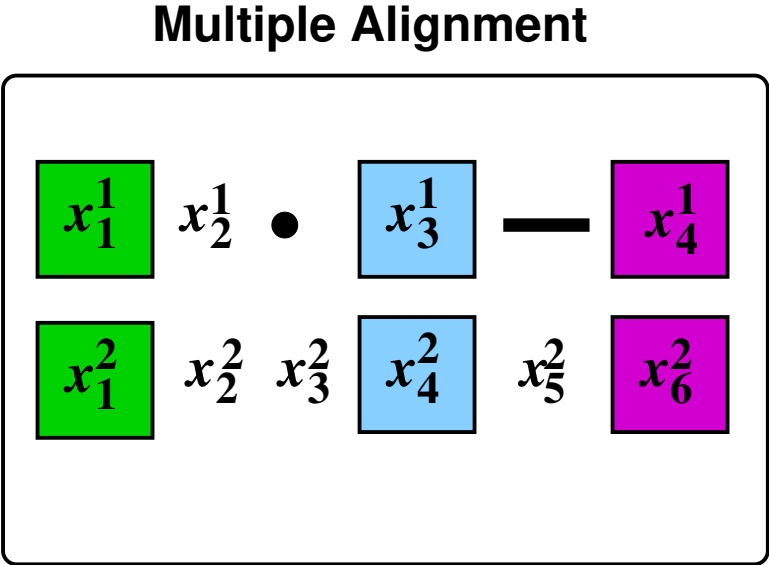
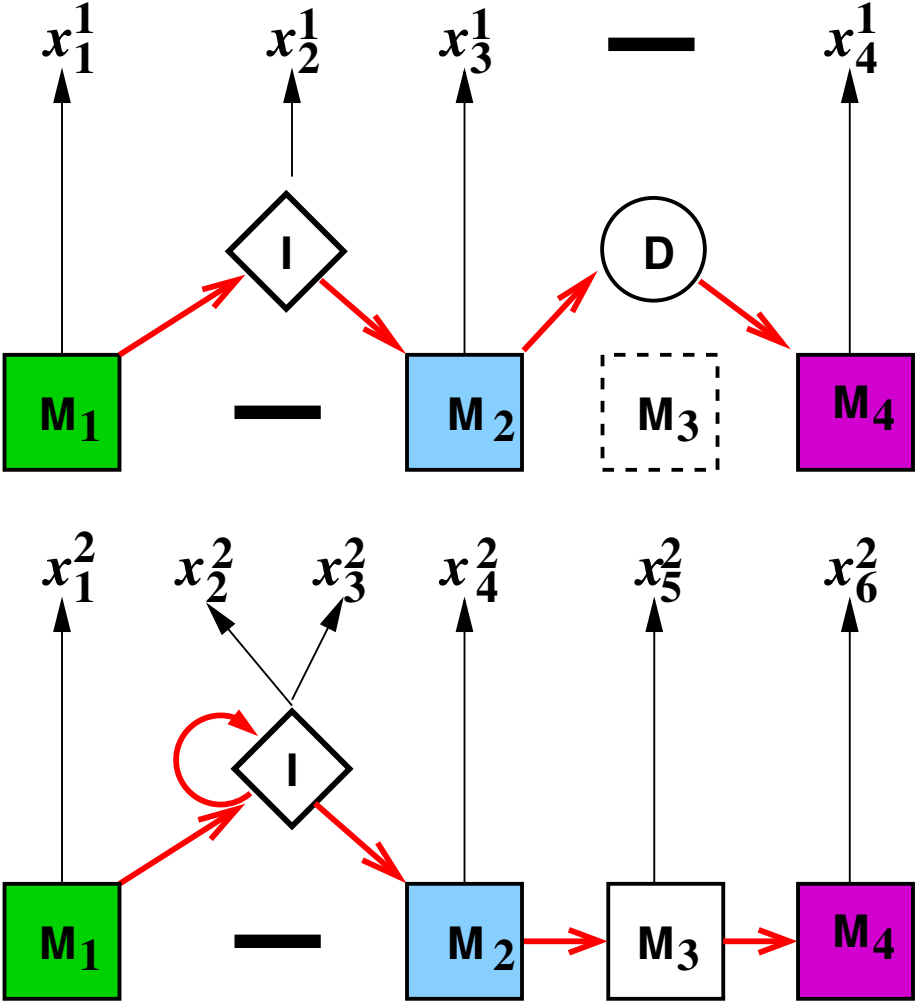
Scoring Metrics

- A scoring scheme should take into account that...
 1. some positions are more conserved than others
 ~> **position specific scoring**;
 2. the sequences are not independent, but are related by a **phylogenetic tree**.
- **Ideal scoring**: Complete probabilistic model of evolution
 ~> Probability of a multiple alignment is composed of the probabilities of all evolutionary events necessary to produce the alignment.
- In practice, we do not have such a model
 ~> **simplifying assumptions**: Two main concepts:
 1. Position specific, but ignoring the phylogenetic tree;
 2. explicit tree model, but position independent.

Multiple alignments by Profile HMM training

- Suppose we have successfully trained a **profile HMM** from a set of labeled sequences.
How can we use this HMM to derive the multiple alignment of n sequences?
- **Answer:** align all n sequences to the profile using the Viterbi algorithm \rightsquigarrow most probable state paths for all sequences.
- Characters aligned to the same match state are aligned in columns.
- Multiple alignments from HMMs are approximations of type one:
 - Score is **position specific**,
 - but sequences are treated as **independent objects**.

Computing the multiple alignment: example



Computing the multiple alignment: Real example

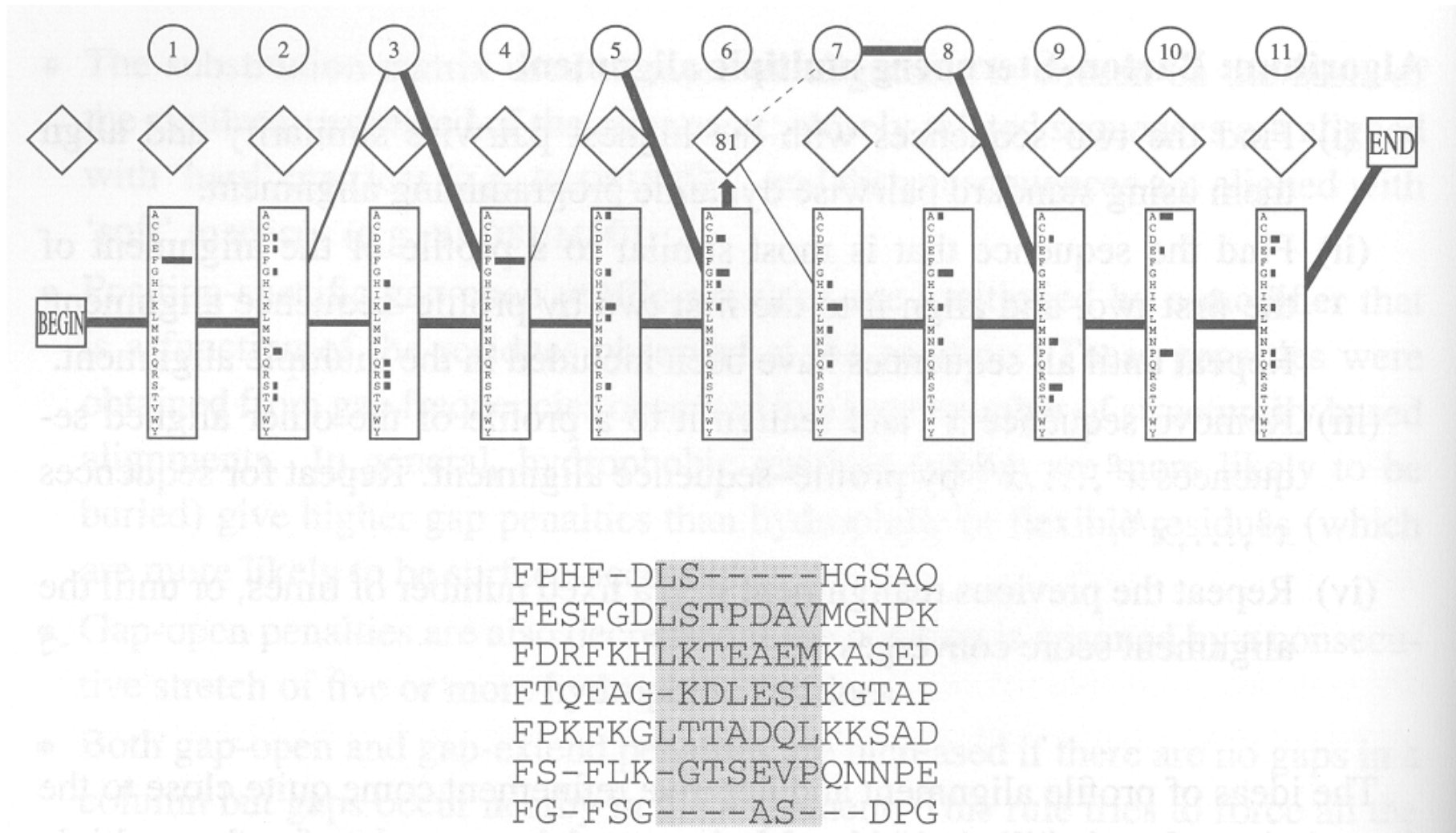


Figure 6.4 A model (top) estimated from an alignment (bottom). The characters in the shaded area of the alignment were treated as inserts.

Computing the multiple alignment: Real example

FPHF-Dls.....HGSAQ	FS-FLKngvdptaa-i--NPK
FESFGDlstpdavMGNPk	FPHF-Dls.....HGSAQ
FDRFKHlkteaemKASED	FESFGDlstpdav..MGNPk
FTQFAGkdlesi..KGTAP	FDRFKHlkteaem..KASED
FPKFKGlttadqlKKSAD	FTQFAGkdlesi...KGTAP
FS-FLKgtsevp.QNNPE	FPKFKGlttadql..KKSAD
FG-FSGas.....--DPG	FS-FLKgtsevp...QNNPE
	FG-FSGas.....--DPG

Figure 6.6 *Left: the alignment of the seven sequences is shown with lower-case letters meaning inserts. The dots are just space-filling characters to make the matches line up correctly. Right: the alignment is shown after a new sequence was added to the set. The new sequence is shown at the top, and because it has more inserts more space-filling dots were added.*

Multiple Alignments by Profile HMM training

- For parameter estimation in Profile HMMs, aligned training sequences are often unavailable
↪ usually we only have a sample of unaligned sequences, the **state paths are unknown**.
- **Idea:** Use EM algorithm for iterative parameter optimization (Baum-Welch algorithm).
- Recall: for the EM algorithm, we need the forward and backward probabilities in the E-step for calculating
 - E_{bl} (the expected emission counts) and
 - $A_{l'l}$ (expected transition counts).

Simpler Multiple Alignment Algorithms

- Alternative to the probabilistic HMM formulation:

Sum of Pairs score:

Sum of scores between all pairs of sequences.

- The SP score for a column m_j of the multiple alignment is

$$S(m_j) = \sum_{k < l} \underbrace{s(m_j^k, m_j^l)}_{\text{from scoring matrix}}$$

- SP scores lack a probabilistic justification:

Correct log-odds score for 3-way alignment would be

$$s(a, b, c) = \log \frac{p_{abc}}{q_a q_b q_c} \neq \underbrace{\log \frac{p_{ab}}{q_a q_b} + \log \frac{p_{bc}}{q_b q_c} + \log \frac{p_{ac}}{q_a q_c}}_{\text{SP score}}$$

Approximation Algorithms for MSA

- Even for SP scores, MSA has exponential time complexity.
- Denote by $D(S, T)$ the **minimum cost** of aligning S with T .
- Let $\sigma(x, y)$ be our **cost function**, i.e. the cost of aligning the character x with the character y , for $x, y \in \Sigma \cup \{-\}$.
- Here we minimize costs σ instead of maximizing scores s .
Example transformation: $\sigma(x, y) = \exp(-\lambda s(x, y))$.
- We assume that $\sigma(-, -) = 0$, $\sigma(x, y) = \sigma(y, x)$,
and that the triangle inequality holds: $\sigma(x, y) \leq \sigma(x, z) + \sigma(z, y)$

Problem: The SP alignment problem.

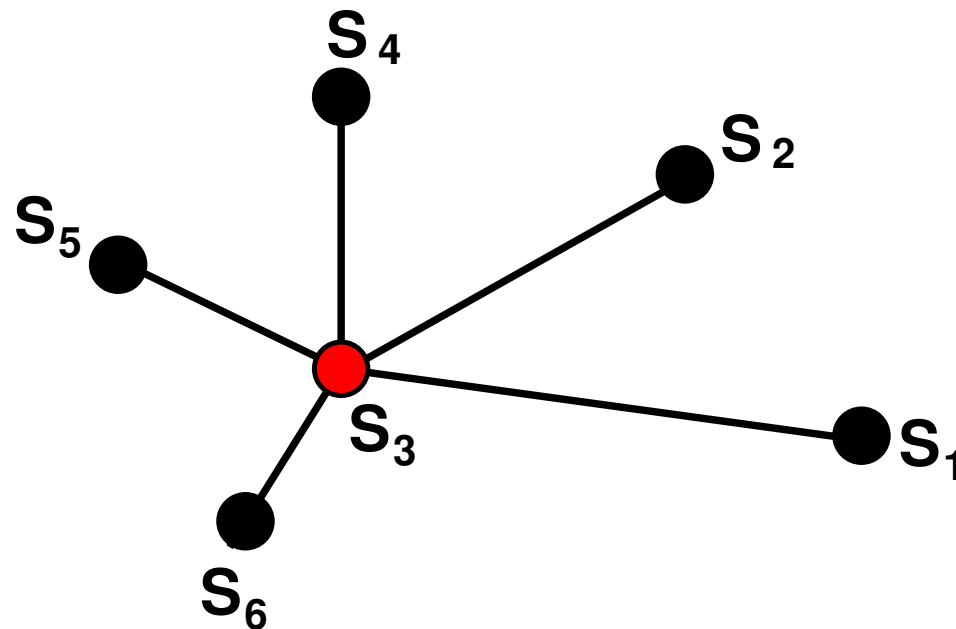
INPUT: A set of sequences $\mathcal{S} = \{S_1, \dots, S_k\}$.

QUESTION: Compute a global multiple alignment \mathcal{M} with minimum SP-costs, given the above assumptions on $\sigma(\cdot, \cdot)$.

The Center Star Method for Alignment

Approximation algorithm for calculating the optimal multiple alignment under the SP metric with **approximation ratio of two.**

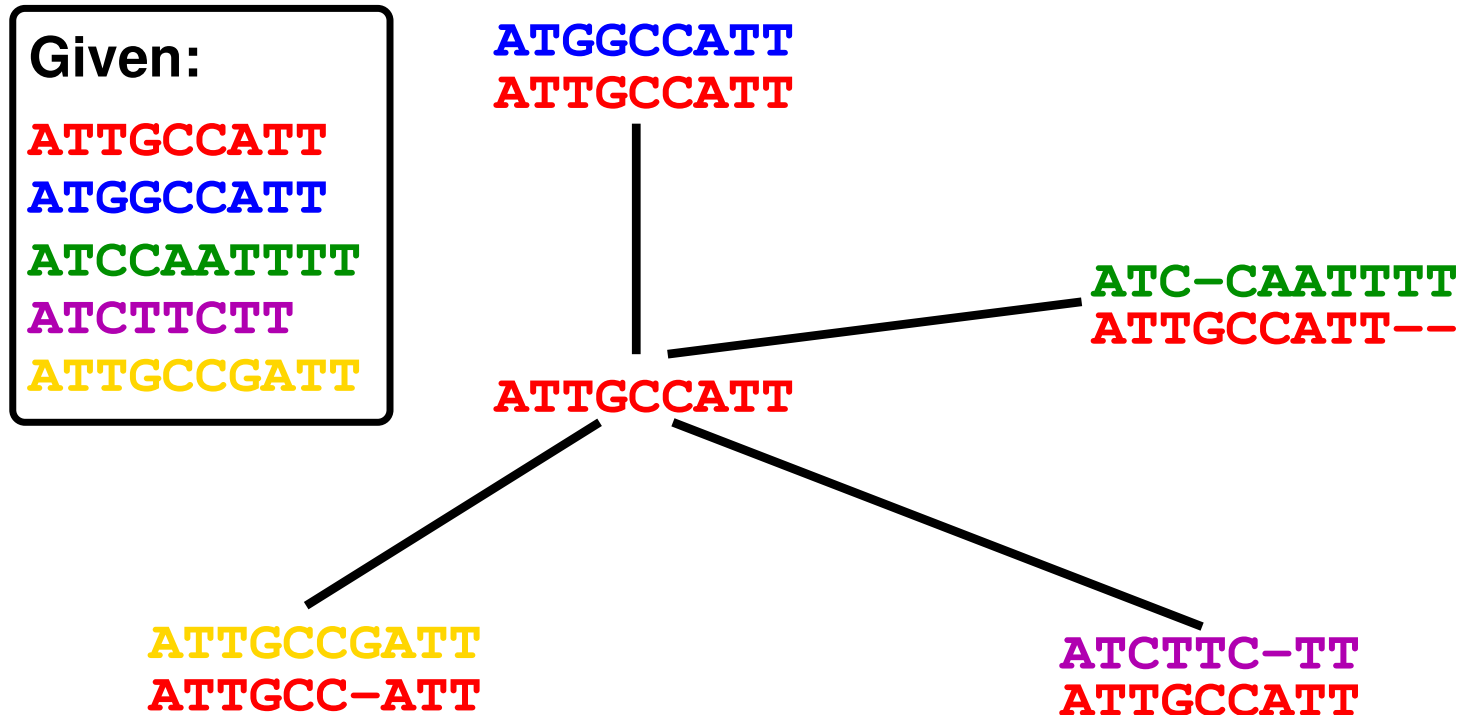
- **Center string:** String that minimizes $\sum_{S_j \in \mathcal{S}} D(S_c, S_j)$.
- **Center star:** A star tree of k nodes, center node labeled S_c , each of the $k - 1$ remaining nodes labeled by $\mathcal{S} \setminus \{S_c\}$.



Type-2 approximation: explicit (star-)tree model, but position independent scoring.

The Center Star Algorithm

1. Find $S_t \in \mathcal{S}$ minimizing $\sum_{i \neq t} D(S_i, S_t)$ and let $\mathcal{M} = \{S_t\}$



2. Add sequences in $\mathcal{S} \setminus \{S_t\}$ to \mathcal{M} one by one so that the **pairwise alignment** of every newly added sequence with S_t is optimal.
Add spaces, when needed, to all pre-aligned sequences.

The Center Star Algorithm

Given:

ATTGCCATT
ATGGCCATT
ATCCAATTTT
ATCTTCTT
ATTGCCGATT

Pair:

ATGGCCATT
ATTGCCATT

ATC-CAATTTT
ATTGCCATT--

ATCTTC-TT
ATTGCCATT

ATTGCCGATT
ATTGCC-ATT

Alignment:

ATTGCCATT
ATGGCCATT

ATTGCCATT--
ATGGCCATT--
ATC-CAATTTT

ATTGCCATT--
ATGGCCATT--
ATC-CAATTTT
ATCTTC-TT--

ATTGCC-ATT--
ATGGCC-ATT--
ATC-CA-ATTTT
ATCTTC--TT--
ATTGCCGATT--

The Center Star Algorithm: Analysis

- \mathcal{M} : Multiple alignment produced by the center-star algorithm.
- $d(i, j)$: Cost of the resulting pairwise alignment of S_i and S_j ,
induced by \mathcal{M} .

Note that
$$\underbrace{D(S_i, S_j)}_{\text{cost of best pairwise alignment}} \leq \underbrace{d(i, j)}_{\text{cost of induced alignment}}$$

- SP-costs of center-star alignment: $\sigma(\mathcal{M}) = \sum_{i=1}^k \sum_{j=1, j \neq i}^k d(i, j)$
- \mathcal{M}^* : Optimal SP-alignment of all strings in \mathcal{S} with costs $\sigma(\mathcal{M}^*)$.

Theorem 1.

$$\frac{\sigma(\mathcal{M})}{\sigma(\mathcal{M}^*)} \leq \frac{2(k-1)}{k} \leq 2.$$

Theorem 2. *The running time of the center star algorithm for k strings with length $\leq n$ is $O(k^2 \cdot n^2)$.*

Proofs: see exercises.

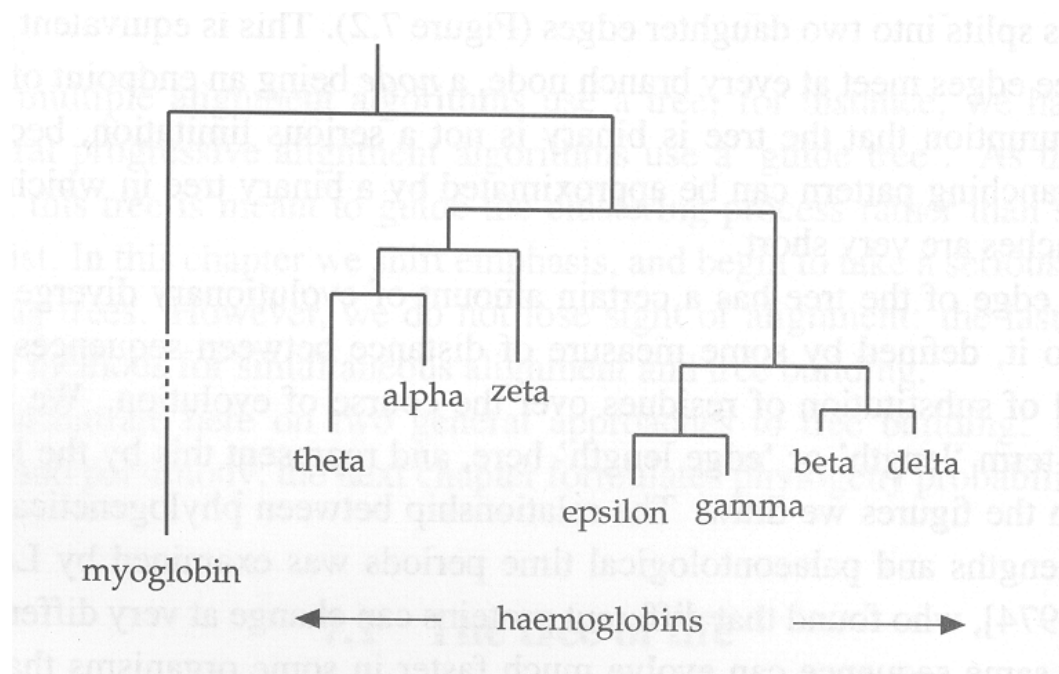
Progressive alignment heuristics

Idea: Use a binary “guide tree” instead of a star tree

(Guide tree defines a model of evolution)

Leaves: sequences, inner nodes: alignments

(sequence-sequence, sequence-profile, or profile-profile).



Durbin et al., Cambridge University Press. <https://doi.org/10.1017/CBO9780511790492.004>

Progressive alignment: ClustalW

ClustalW is a software package for multiple alignment (implementing an algorithm of Thompson, Higgins, Gibson 1994).

1. Calculate all pairwise alignment scores, convert to **pairwise distances**.
2. Use **Neighbor-Joining** algorithm to build a **tree** from the distances.
3. Align **sequence - sequence**, **sequence - profile**, **profile - profile**.

This algorithm makes use of many ad-hoc rules such as weighting, different matrix scores and special gap scores.

