Multiple Sequence Alignment:

Progressive Methods and HMMs

Many of these slides come from Terry Speed's Computational Biology Course at UC Berkeley. Please also see the additional notes on website about HMMs.

Why multiple alignment?

The simultaneous alignment of a number of DNA or protein sequences is one of the commonest tasks in bioinformatics.

Useful for:

phylogenetic analysis (inferring a tree, estimating rates of substitution, etc.)

detection of homology between a newly sequenced gene and an existing gene family

prediction of protein structure

demonstration of homology in multigene families

determination of a consensus sequence (*e.g.,* in assembly)

A multiple alignment (globin peptides)

```
                        10        20        30        40        50        60
                         .         .         .         .         .         .
Hbb_Human.pep   --------VHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDLST
Hbb_Horse.pep   --------VQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDLSN
Hba_Human.pep   ---------VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLS--
Hba_Horse.pep   ---------VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLS--
Myg_Phyca.pep   ---------VLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLKT
Glb5_Petma.pep  PIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLTT
Lgb2_Luplu.pep  --------GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE
                          *.      .   *                       * .   *

Hbb_Human.pep   PDAVMGNPKVKAHGKKVLGAFSDGLAHLD-----NLKGTFATLSELHCDKLHVDPENFRL
Hbb_Horse.pep   PGAVMGNPKVKAHGKKVLHSFGEGVHHLD-----NLKGTFAALSELHCDKLHVDPENFRL
Hba_Human.pep   ----HGSAQVKGHGKKVADALTNAVAHVD-----DMPNALSALSDLHAHKLRVDPVNFKL
Hba_Horse.pep   ----HGSAQVKAHGKKVGDALTLAVGHLD-----DLPGALSNLSDLHAHKLRVDPVNFKL
Myg_Phyca.pep   EAEMKASEDLKKHGVTVLTALGAILKKKG-----HHEAELKPLAQSHATKHKIPIKYLEF
Glb5_Petma.pep  ADQLKKSADVRWHAERIINAVNDAVASMDDT--EKMSMKLRDLSGKHAKSFQVDPQYFKV
Lgb2_Luplu.pep  VP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-VADAHFPV
                         .. *   .                         *   *     .

Hbb_Human.pep   LGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------
Hbb_Horse.pep   LGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH------
Hba_Human.pep   LSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------
Hba_Horse.pep   LSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSKYR------
Myg_Phyca.pep   ISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
Glb5_Petma.pep  LAAVIADTVAAG---------DAGFEKLMSMICILLRSAY-------
Lgb2_Luplu.pep  VKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA---
                .   .   .                         .
```

Extending the pairwise alignment algorithms

- Generally not feasible for more than a small number of sequences (~5), as the necessary computer time and space quickly becomes prohibitive. Computational time grows as N^m, where *m* = number of sequences. For example, for 100 residues from 5 species, 100^5 = 10,000,000,000 (*i.e.*, the equivalent of two sequences each 100,000 residues in length.)

- Nor is it wholly desirable to reduce multiple alignment to a similar mathematical problem to that tackled by pairwise alignment algorithms. Two issues which are important in discussions of multiple alignment are:

  - the treatment of gaps: position-specific and/or residue-specific gap penalties are both desirable and feasible, and

  - the phylogenetic relationship between the sequences (which must exist if they are alignable): it should be exploited.

Progressive alignment

Up until about 1987, multiple alignments would typically be constructed manually, although a few computer methods did exist. Around that time, algorithms based on the idea of **progressive alignment** appeared. In this approach, a pairwise alignment algorithm is used iteratively, first to align the most closely related pair of sequences, then the next most similar one to that pair, and so on.

The rule “once a gap, always a gap” was implemented, on the grounds that the positions and lengths of gaps introduced between more similar pairs of sequences should not be affected by more distantly related ones.

Multiple alignment in 2002

The most widely used progressive alignment algorithm is currently **CLUSTAL W**. However, there are a number of more specialized procedures based on quite different principles, including the use of hidden Markov models built for protein families. A relatively new and promising approach uses Markov chain Monte Carlo methods to sample alignments according to certain probabilistic procedures and, by moving randomly around in the huge space of possible alignments, to find good alignments.

CLUSTAL W

The three basic steps in the **CLUSTAL W** approach are shared by all progressive alignment algorithms:

A. Calculate a matrix of **pairwise distances** based on pairwise alignments between the sequences

B. Use the result of A to build a **guide tree**, which is an inferred phylogeny for the sequences

C. Use the tree from B to guide the **progressive alignment** of the sequences

Calculating the pairwise distances (A)

A pair of sequences is aligned by the usual dynamic programming algorithm, and then a similarity or distance measure for the pair is calculated using the aligned portion (gaps excluded) - for example, percent identity.

Globin example

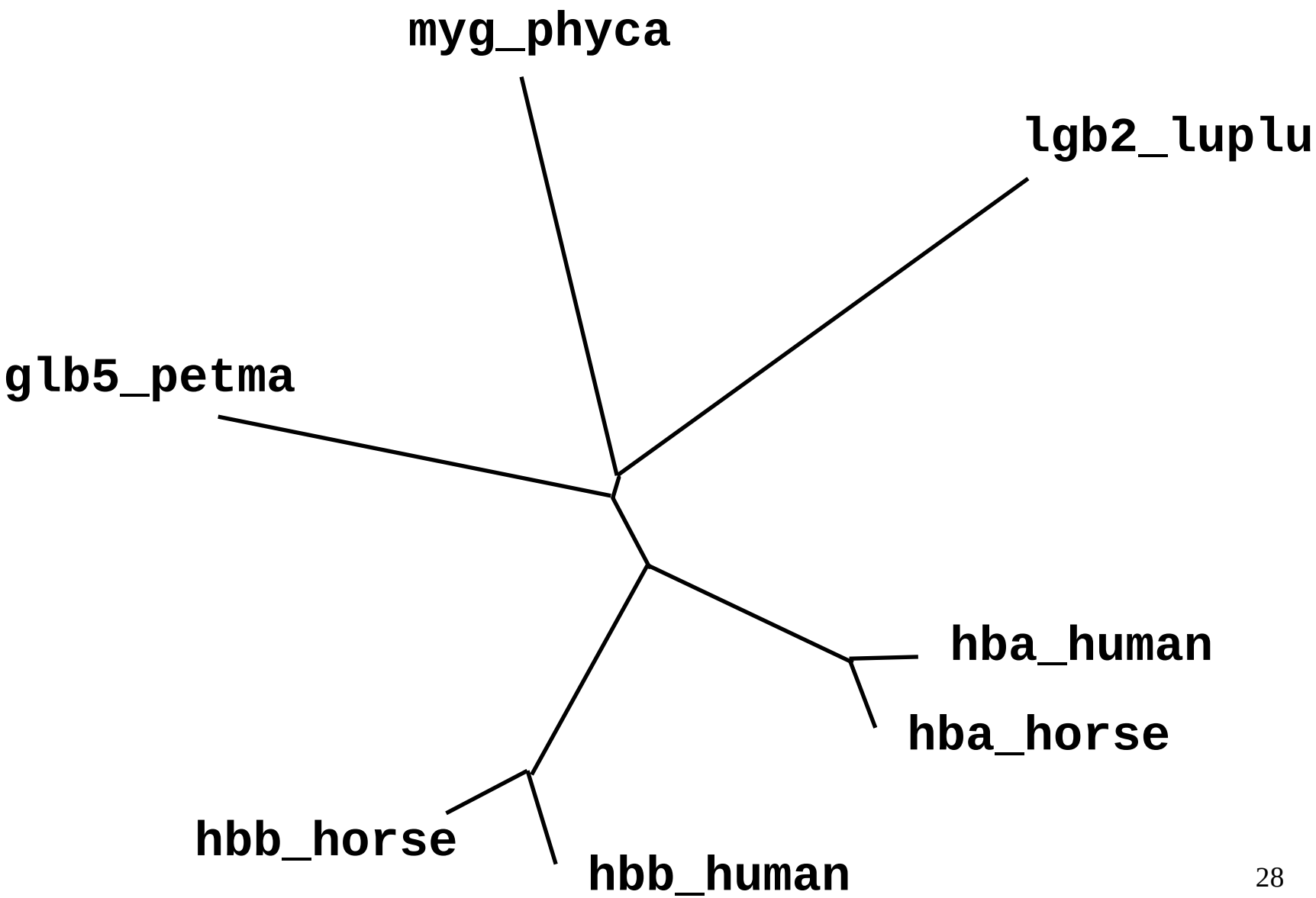
```
 DISTANCES between protein sequences:

 Calculated over: 1 to 167
 Correction method: Simple distance (no corrections)
 Distances are: observed number of substitutions per 100 amino acids
Symmatrix version 1
Number of matrices: 1

//
Matrix 1, dimension: 7

Key for column and row indices:

  1 hba_human
  2 hba_horse
  3 hbb_human
  4 hbb_horse
  5 glb5_petma
  6 myg_phyca
  7 lgb2_luplu

 Matrix 1: Part 1
```

	1	2	3	4	5	6	7
1	0.00	12.06	54.68	55.40	64.12	71.74	83.57
2		0.00	55.40	53.96	64.89	72.46	82.86
3			0.00	16.44	74.26	73.94	82.52
4				0.00	75.74	73.94	81.12
5					0.00	75.91	82.61
6						0.00	80.95
7							0.00

Building the guide tree (B)

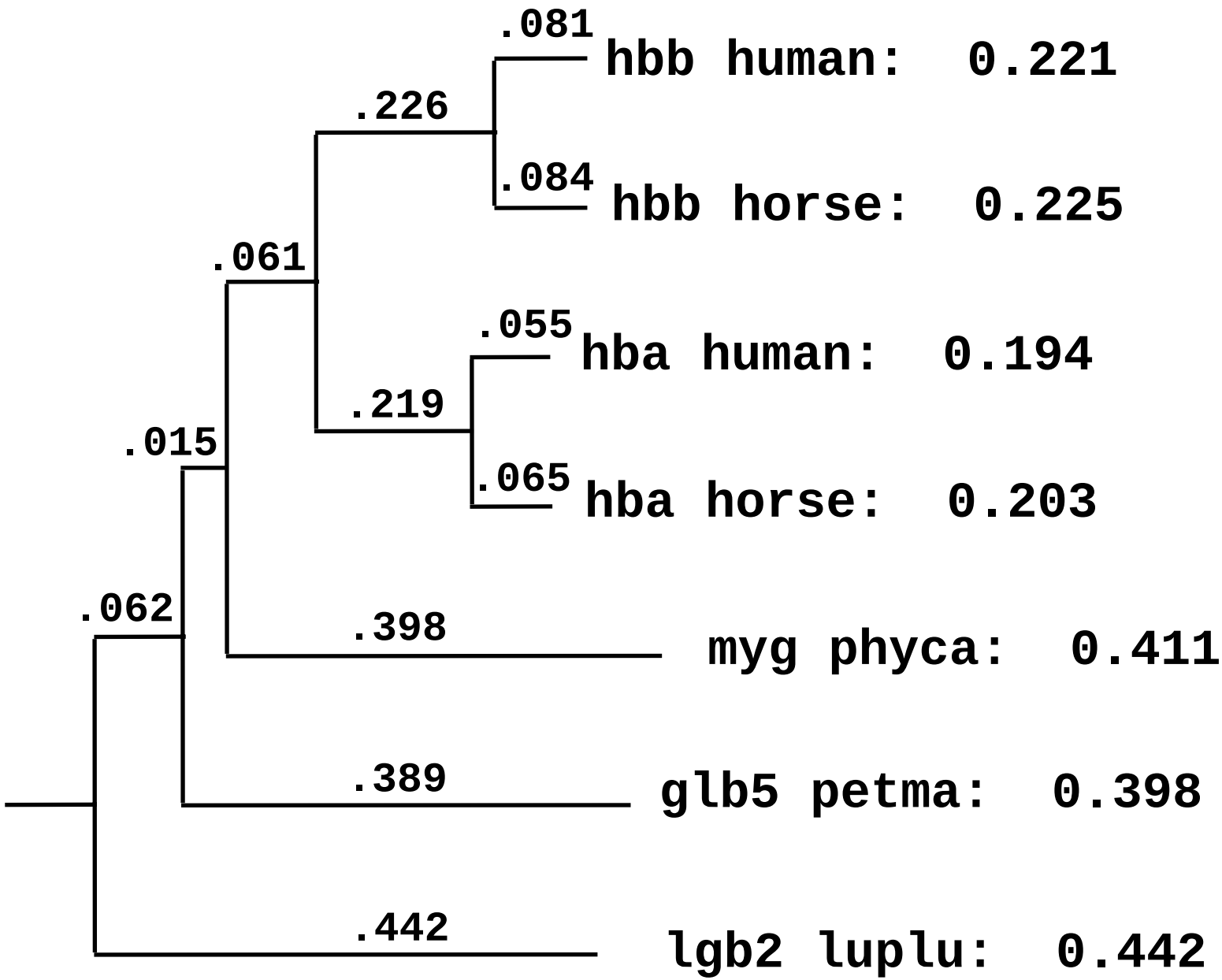
- There are many ways of building a tree from a matrix of pairwise distances. **CLUSTAL W** uses the *neighbour-joining* (NJ) method, which is the most favoured approach these days.
- A *root* of the tree is then determined by the so-called *mid-point method* (giving equal means for the branch lengths on either side of the root).
- The **W** in **CLUSTAL W** stands for Weights, an important feature of this program. These are calculated in a straightforward way. They correct for unequal sampling at different evolutionary distances.

NJ globin tree



Tree, distances, and weights

Thompson *et al*. (1994)



Progressive alignment (C)

The basic idea is to use a series of pairwise alignments to align larger and larger groups of sequences, following the branching order of the guide tree. We proceed from the tips of the rooted tree towards the root.

In our globin example, we align in the following order:

a) human and horse beta-globin;
b) human and horse alpha-globin;
c) the two beta-globins and the two alpha-globins;
d) myoglobin and the haemoglobins;
e) cyanohaemoglobin and the combined haemoglobin, myoglobin group;
f) leghaemoglobin and the rest.

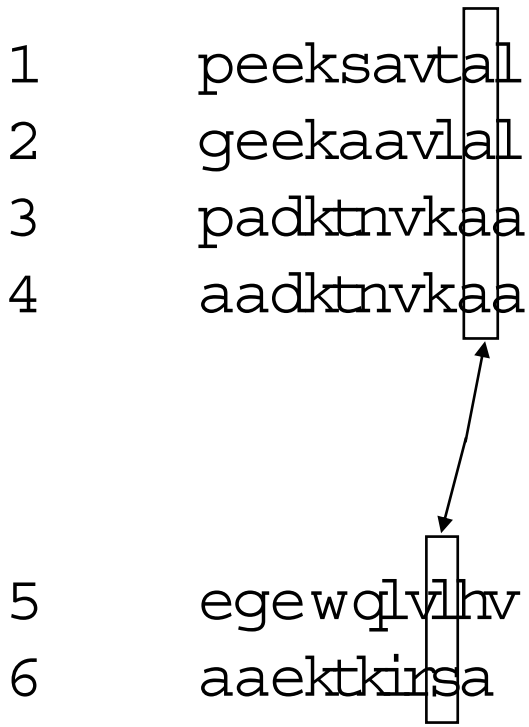
Progressive alignment (C, 2)

At each stage a full dynamic programming algorithm is used, with a residue scoring matrix (*e.g.,* a PAM or a BLOSUM matrix) and gap opening and extension penalties.

Each step consists of aligning two existing alignments. Scores at a position are averages of all pairwise scores for residues in the two sets of sequences using matrices with only positive values. Gap vs. residue scores zero. Sequence weights are used at this stage. See next slide.

Gaps that are present in older alignments remain fixed. New gaps introduced at each stage initially get full opening and extension penalties, even if inside old gap positions. This gets modified.

Scoring an alignment of two partial alignments



Sequence weights

$w_1, \ldots, w_6$

Score: $\frac{1}{8}[M(t,v)w_1w_5 + M(t,i)w_1w_6 + \ldots + M(k,i)w_4w_6]$

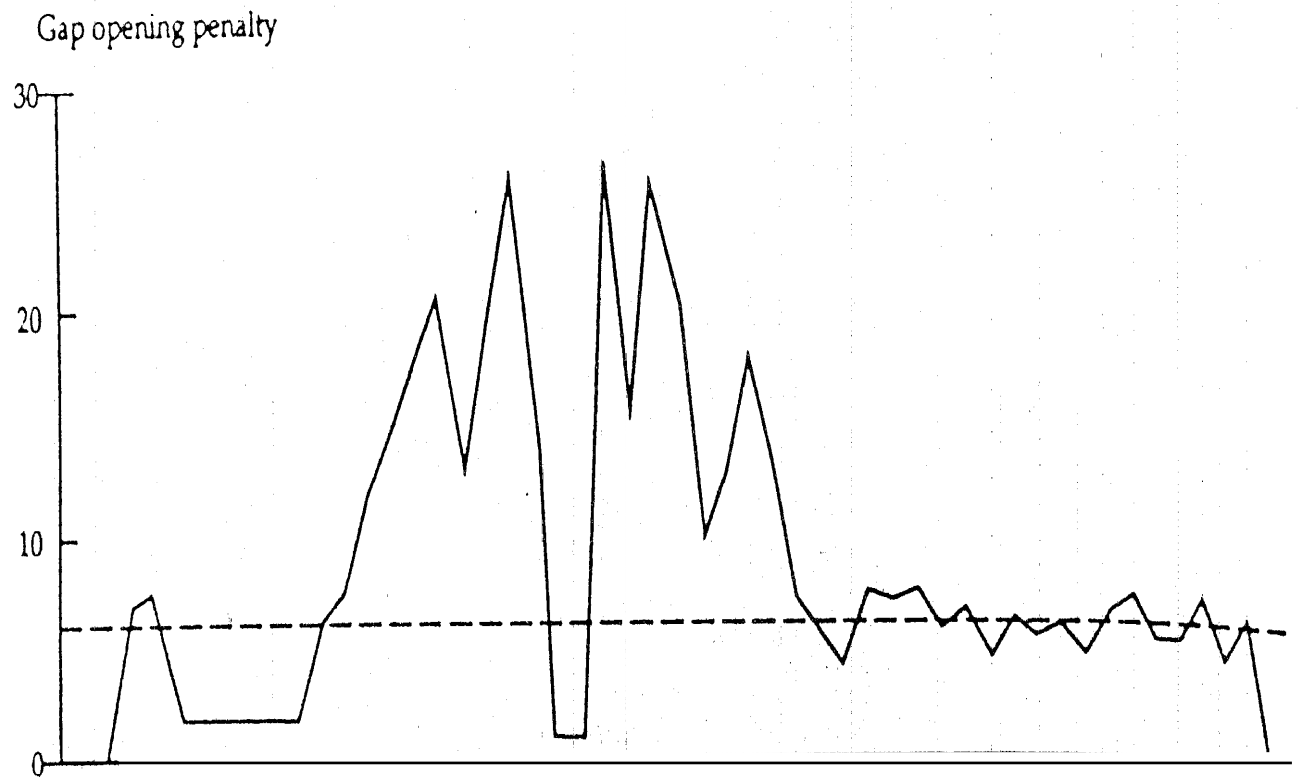
Progressive alignment (C) - gaps

CLUSTAL W has quite a sophisticated treatment of gaps, incorporating into opening and extension penalties a dependence on a) weight matrix, b) sequence similarity, c) sequence length, d) difference in sequence length, e) position of gaps (see figure), f) residues at gaps.

Regarding e) and f), the motivation is as follows: if one knew the positions of all secondary structure elements (alpha-helices, beta-strands) in all or some of the sequences, one could increase the gap penalties inside and decrease outside them, forcing gaps to occur most often in loop regions, which is what is observed in alignments of sequences with known 3-D structure.

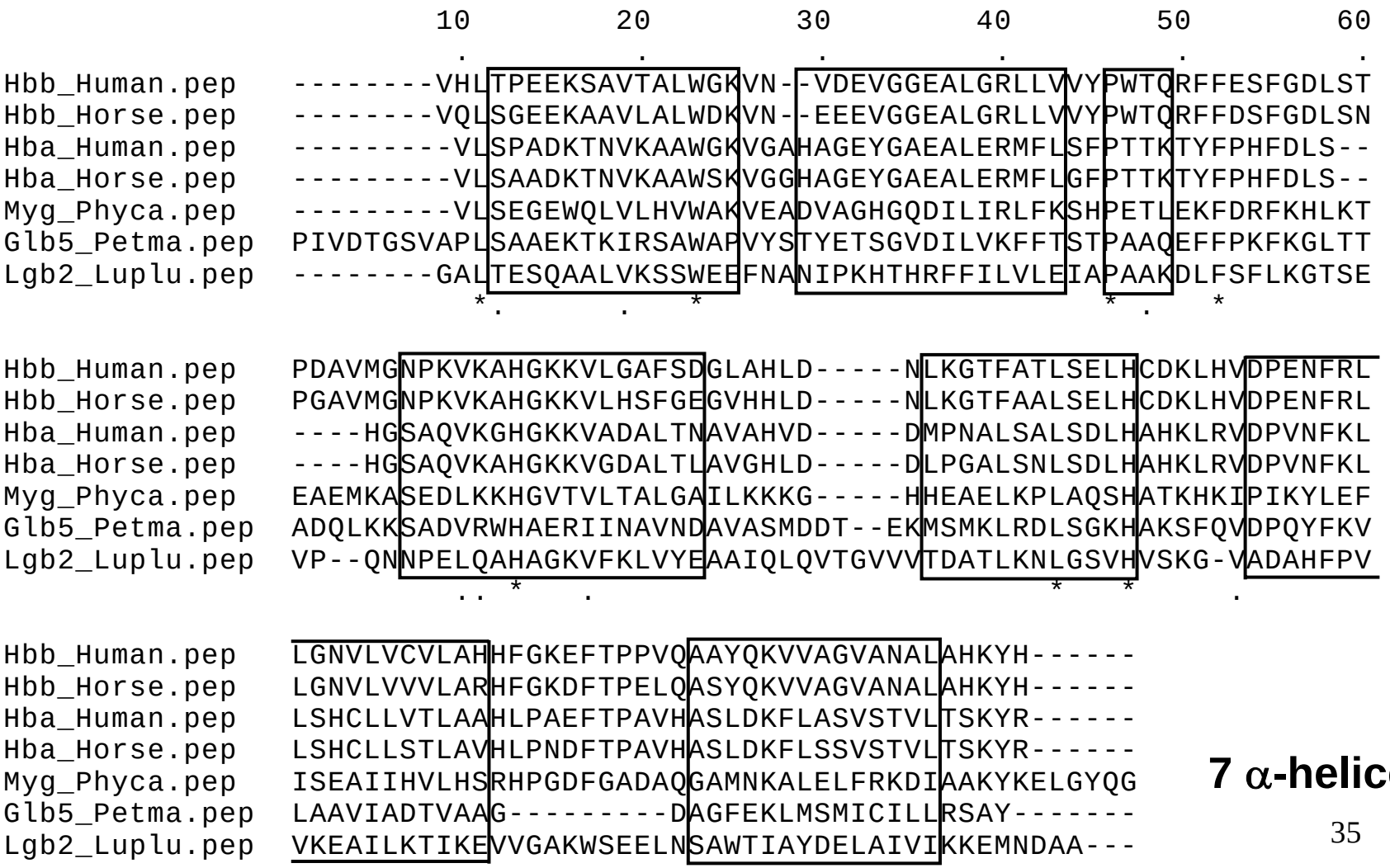
For further details, see Thompson *et al*., **NAR** 1994, **22:**4673 or **Methods in Enz**. 1996, **266:**article 22.

Position and residue -specific gap opening penalties




```
HLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDL
QLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDL
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLS
VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLS
```

Final CLUSTALW alignment (using eclustalw)



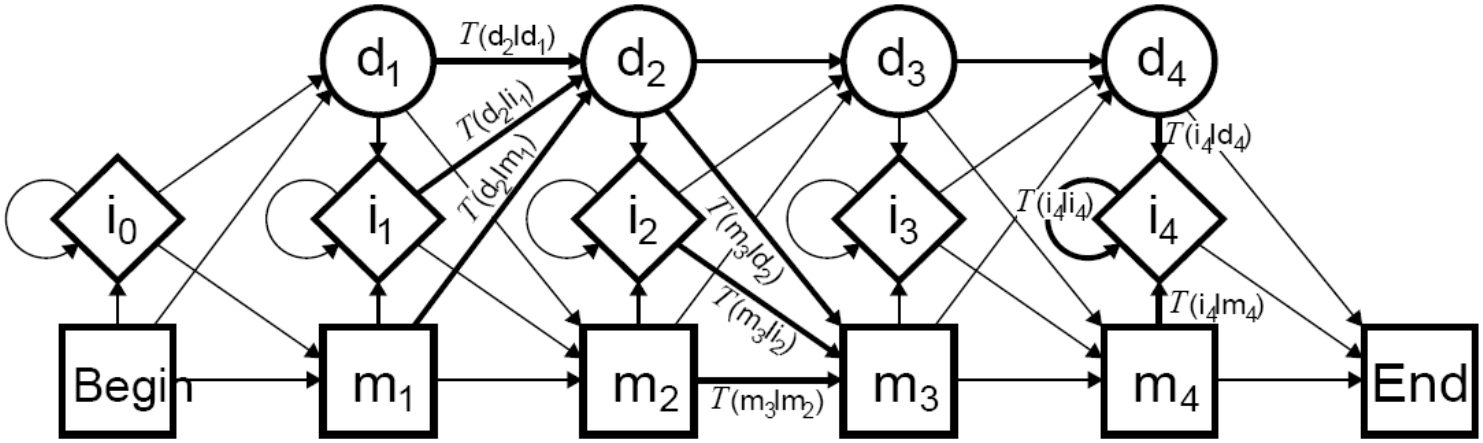
7 α-helices

Alignment using Hidden Markov models

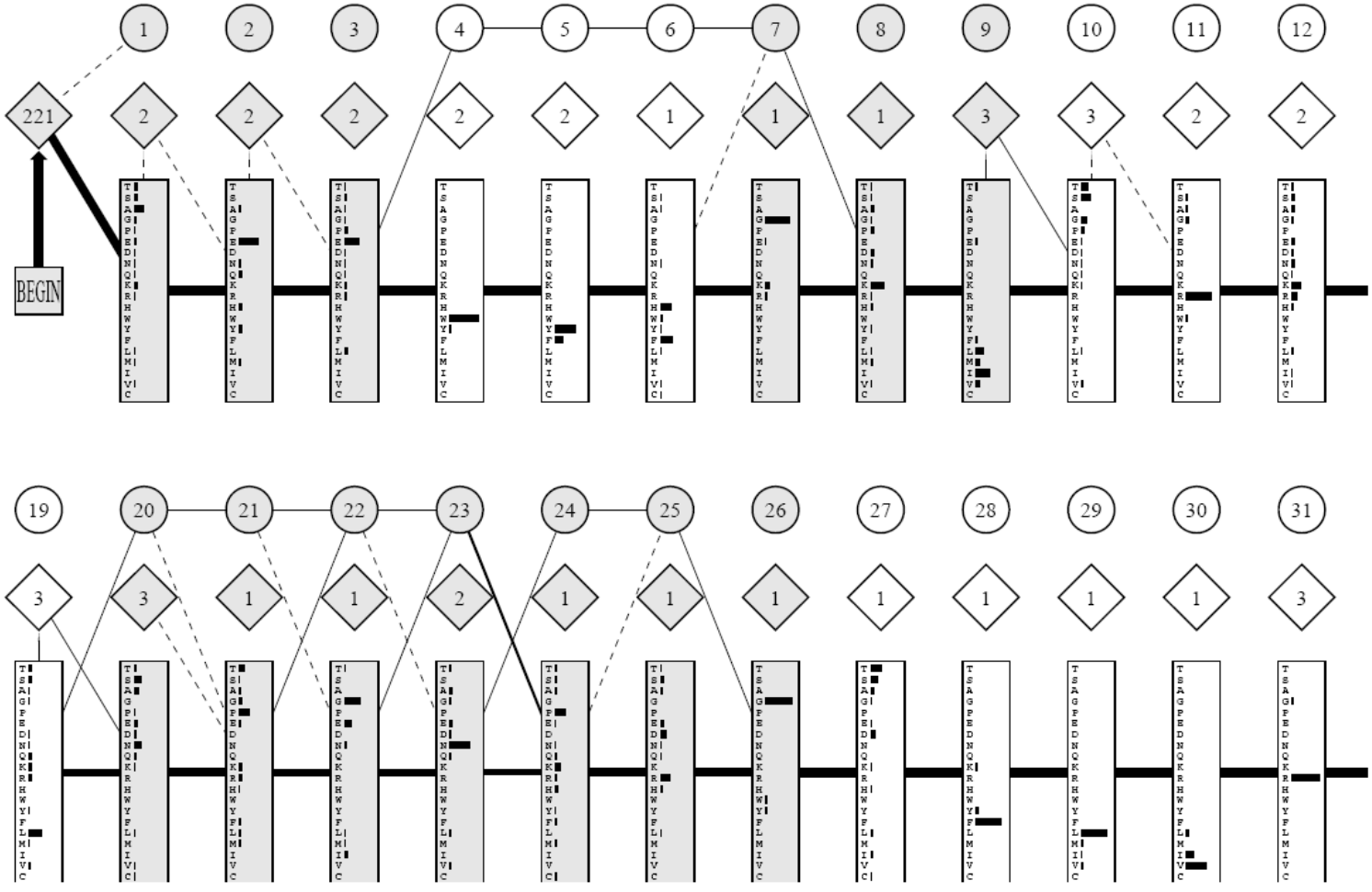
There are now many HMMs for protein families such as globins, and these models can be used to infer alignments of new globin sequences to other members of the family.

Such models can also be used to determine whether a given sequence is or is not a member of a specified family.

HMMs for Multiple Alignment



- Markov chain moves to the right.
- $S_t \in \{$ delete, insert, match$\}$.
- Delete states output δ with probability 1.
- Insert and match states have their own specific HMM.
- The algorithm is trained on a set of sequences from a protein family using the Baum-Welch algorithm.
- The trained HMM can be used to align new sequences (Viterbi algorithm).



(See complete figure in Hughey and Krogh, 1995)