Multiple Sequence Alignments

COS551, Fall 2003

Global Multiple Sequence Alignment (MSA)

- Ex: MSA of 4 sequences `MQPILLLV`, `MLRLL`, `MKILLL`, and `MPPVLILV`:

```
MQPILLLV
MLR—LL--
MK-ILLL-
MPPVLILV
```

No column is all gaps

Motivation

- Multiple sequence alignments are used for many reasons, including:
  - to detect regions of variability or conservation in a family of proteins
  - to provide stronger evidence than pairwise similarity for structural and functional inferences
  - first step in phylogenetic reconstruction, in RNA secondary structure prediction, and in building profiles (probabilistic models) for protein families or DNA signals.

Similarity Measures

- For pairwise alignments, we aligned sequences to maximize the similarity score.
- With multiple sequences, not obvious what best way to score an alignment is
- Sum-of-pairs (SP) is a commonly studied similarity measure for MSAs

Sum-of-pairs (SP) Measure

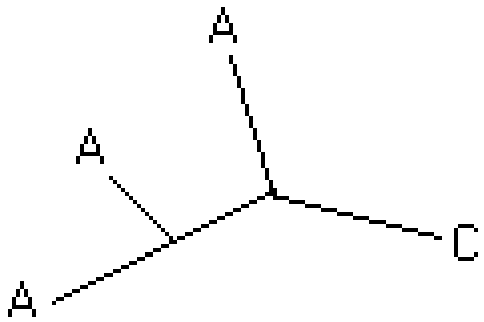
- Each column is scored by summing the scores of all pairs of symbols in that column.
- E.g., match = 1, a mismatch = -1, gap = -2

```
I
-
I
V
```

= score(I,-) + score(I, I) +score(I,V) + score(-,I) + score (-,V) + score(I,V)
= -2 + 1 + -1 + -2 + -2 + -1 = -7

Is SP a good measure?

- column in alignment : A,A,A,C
- SP score = 1+1-1+1-1-1=0
- But maybe evolutionary history described by:



single C ⟶ A mutation can explain the data,
and thus SP tends to overcount mutations

Optimal pairwise alignments (Review)

- Used dynamic programming
- If two length n sequences: (n+1) x (n+1) array
- Fill out each box in the array by considering what happens in the last column
  - 3 choices: align last letters from both sequences, align last letter from 1st sequence with gap, align last letter from 2nd sequence with gap
  - $O(n^2)$ algorithm

Finding optimal MSAs

Can use dynamic programming to find optimal solutions

If have *k* sequences of length *n*, array is of size $(n+1)^k$

In considering last column, have $2^k - 1$ choices

- E.g., align last letters from all sequences; align last letter from one sequence and gaps in all others, etc.

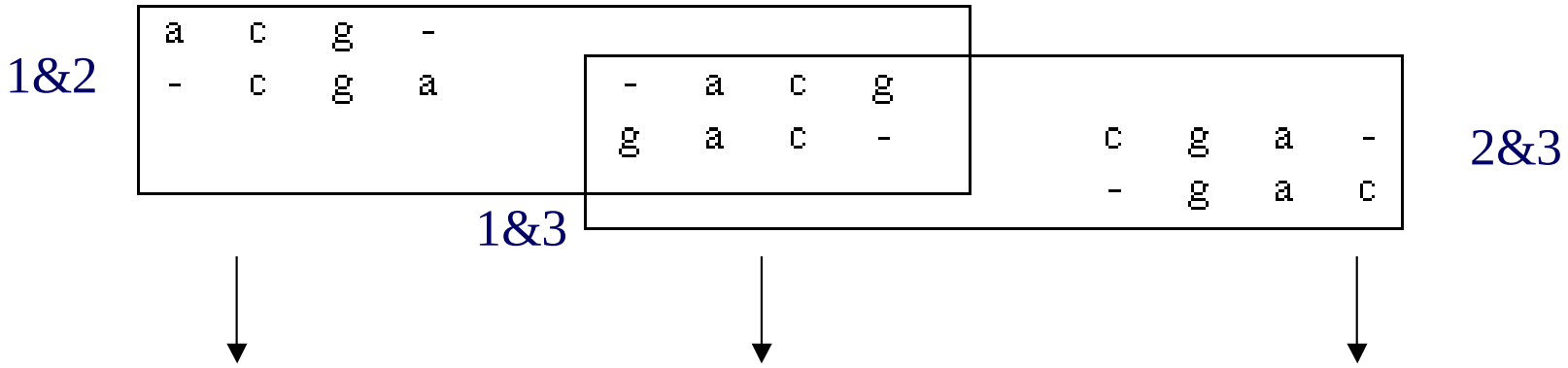
- Running time is exponential in the number of sequences !
- Impractical … MSA packages use heuristics

Progressive alignment heuristic

- basic idea: compute pairwise alignments and merge alignments consistently
- E.g., Align acg, cga, gac. Get optimal pairwise alignments:

```
acg-      -acg     cga-
-cga      gac-      -gac
```

Progressive alignment heuristic



Merge using alignments with 1st sequence

```
-acg-
--cga
gac--
```

Merge using alignments with 3rd sequence

```
--acg
cga--
-gac-
```

Merge using alignments with 2nd sequence

```
acg--
-cga-
--gac
```

Order of merging matters ! Note once a gap, always a gap …

ClustalW Package

- ClustalW is a popular heuristic package for computing MSAs,
- Based on progressive alignment
- We'll go over its main ideas via an example of aligning 7 globin sequences
- Keep in mind what types of problems the algorithm might have on real data!

Progressive Alignment: ClustalW Package

1. Determine all pairwise alignments between sequences and determine degrees of similarity between each pair.
2. Construct a "rough" similarity tree
3. Combine the alignments starting from the most closely related groups to most distantly related groups, while maintaining the "once a gap, always a gap" policy.

Step 1: Pairwise alignment & distance

- Given *k* sequences, determine all pairwise global alignments
- Use pairwise alignments to determine distances between pairs of sequences
  - E.g., sequences QKLMN & KLVN, alignment is:

```
QKLMN
-KLVN
```

Distance= # mismatches / #cols with no gaps

= ¼

Underestimate of actual distance!

Compute all distances

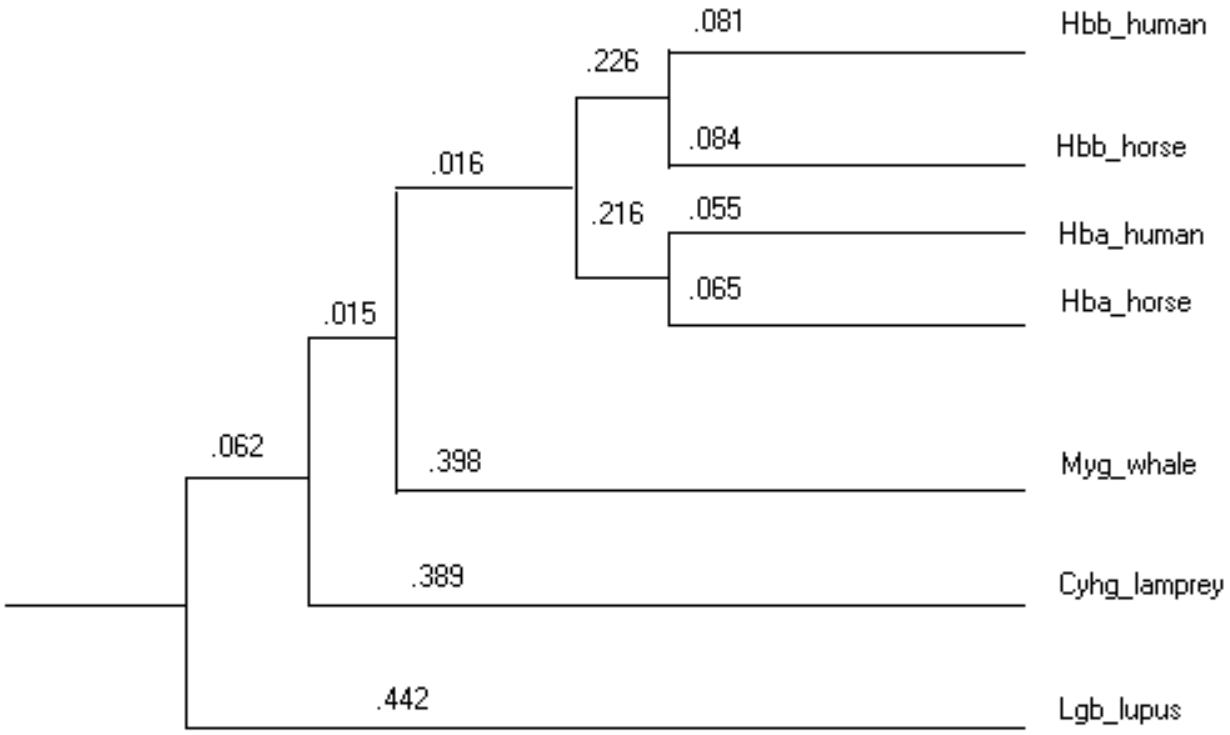
Globin type		1	2	3	4	5	6	7
Hbb_human	1	-						
Hbb_horse	2	.17	-					
Hba_human	3	.59	.60	-				
Hba_horse	4	.59	.59	.13	-			
Myg_whale	5	.77	.77	.75	.75	-		
Cyng_ lamprey	6	.81	.82	.73	.74	.80	-	
Lgb_lupus	7	.87	.86	.86	.88	.93	.90	-

-distances between 0 and 1
-smaller distances, closer seqs

Step 2: Construct "rough" similarity tree

- Distance matrix is fed into an algorithm that will build a tree relating these sequences (Neighbor-joining, more in future lecture)
- Ideally, path length in tree between sequences is equal to distance in matrix (cannot always maintain this)

Neighbor Joining Tree



Note: Figure not drawn to scale

distance between Hbb_human and Hbb_horse tree
is .081 + .084 = .165 which is close to .17 from matrix

Step 3: Combine alignments

- Start from the most closely related groups to most distantly related groups (start from tips to root in tree), while maintaining the "once a gap, always a gap" policy.
- E.g., first align hba_human & hba_horse; then hbb_human & hbb_horse; then hba's with hbb's; then add to that alignment whale, lamprey and lupus in turn

Aligning pairs of alignments

- Can solve optimally using dynamic programming
- Similarity between a column in 2 alignments is now the average similarity between the sequences

Aligning Aligments

```
Alignment 1:   ATA
               CCA

Alignment 2:  TCAFE
              TAT-E
              TATF-
              AGTFD
```

Score 1st column of 1st alignment against 2nd column in the other alignments using:

```
= 1/8(score(A,C) + score(A,A) + score(A,A) + score(A,G) +
score(C,C) + score(C,A) + score(C,A) + score(C,G))
```

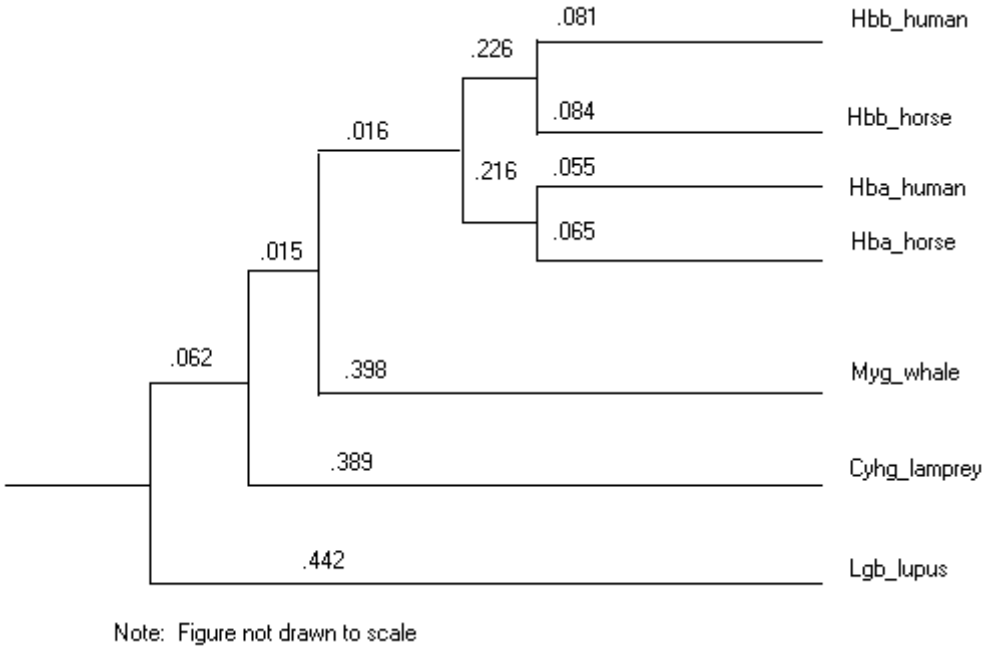
Weighting Sequences

- Note that when aligning alignments, we are just averaging over all sequences
- If have some very closely related sequences, this is problematic (duplicate information)
- Will use tree to weight our sequences, with highly diverged sequences getting larger weights

Weighting Sequences

- Use length from root to sequences to compute weights à increased weights for more divergent species
- If 2 or more sequences share a branch, length of branch is split amongst sequences à reduced weight for related sequences
- Use these weights when scoring alignments of alignments (instead of just averaging equally)

Weighting Sequences



Lgb_lupus: weight of .442

Hba_human: weight of .055 + .216/2 + .061/4 + .015/5 + .062/6 = .194

Caveats for MSAs and ClustalW

- Progressive alignment says nothing about the optimum MSA (sum-of-pairs or any other measure).
- Initial errors from "once a gap, always a gap" are propagated/compounded
- More than one optimum pairwise alignment possible, yet we are committing ourselves to only one at the outset

Caveats for MSAs and ClustalW

- Order in which we add sequences to the alignment (e.g. based on the guide tree) changes alignment.
- Parameter setting always an issue with alignments. (Which matrices, gap penalties?)
- If any pair of sequences are less than 25% identical, then the alignments are prone to be bad.
- In general, one needs to correct some alignments manually.

Using MSAs to search for other sequences

- Once have a MSA, may want to search for other similar sequences (more sensitivity than pairwise searches)
- Often observe blocks of conserved regions, sometimes called motifs
- Can use these blocks (or even entire alignments) to make probabilistic profiles that search for similar sequences

Conserved Areas in MSAs

```
--------VHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVV
--------VQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVV
---------VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLS
---------VLSAADKTNVKAAWSKVGGHAGEYGAELERMFLGF
---------VLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKS
PIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTS
--------GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEI
```

In fact, these are fragments of the globin sequences, and first 2 helices are highlighted

Profiles

- Libraries online of common motifs (e.g., Pfam, BLOCKS, etc.)
- Can input your sequence and it tries to find (known) motifs in it
- Motifs could be, e.g., helicase domains, zinc finger domains, etc.
- May want to make your own motifs …

Profile analysis framework

- Given subsequences that belong to a particular family (e.g., helicase)
- Identify whether a new sequence belongs to that family
- Idea
  - Align sequences
  - Create “profile” (probabilistic approach)
  - Test new sequences

Profiles

Step 1: Align members of family

```
LEVK
LDIR
LEIK
LDVE
```

l positions,

l=4 here

Step 2: Compute $f_{i,j}$ = % of column *j* that is amino acid *i*; b_i = % of "background" that is amino acid *i*; and finally $p_{i,j}=f_{ij}/b_i$

e.g. $p_{E,2}$ = (2/4) / (1/20) = 10, assuming uniform background

Intuition: $p_{i,j}$ is "propensity" for position (> 1 is favorable, < 1 is unfavorable); E is 10x more likely in 2nd position than at random

Profiles

- Step 2 gives a 20 x *l* array of propensities
- Step 3: Now to score an *l* long sequence, say LEVE, compute $p_{L,1} \times p_{E,2} \times p_{V,3} \times p_{E,4}$
  - If this is greater than some cutoff, then say "member of the family" otherwise not.
  - In practice, compute $\log(p_{L,1} \times p_{E,2} \times p_{V,3} \times p_{E,4}) = \log(p_{L,1}) + \log(p_{E,2}) + \log(p_{V,3}) + \log(p_{E,4})$
  - So set $score_{i,j} = \log(p_{i,j})$

Profiles

E.g., New sequence LEVEER, find if it contains motif

Score each *l*-long window:
LEVE, EVEE, VEER

Score of LEVE = score $_{L,1}$+score $_{E,2}$+score $_{V,3}$+score $_{E,4}$
Score of EVEE = score $_{E,1}$+score $_{V,2}$+score $_{E,3}$+score $_{E,4}$
Score of VEER = score $_{V,1}$+score $_{E,2}$+score $_{E,3}$+score $_{R,4}$

If any of these larger than cutoff, have found motif & position in sequence

Profiles

- Simple probabilistic interpretation of profiles (important in terms of assumptions and for future topics)
- We'll talk about that more next time … but first some background …

Detour: Estimating parameters

- given some data, how can we determine the probability parameters of our model?
- one approach: *maximum likelihood estimation*
  - given a set of data *D*
  - set the parameters to make the data *D* look most likely under the model

Maximum Likelihood (ML) Estimation

- suppose we want to estimate the parameters Pr(g), Pr(a), Pr(t), Pr(c)
- and we're given the sequences

  gcgcttaacc

  gcttgactct

  cgtttagcac

- then the maximum likelihood estimates are

$$\Pr(g) = \frac{6}{30} \qquad \Pr(a) = \frac{5}{30}$$

$$\Pr(t) = \frac{9}{30} \qquad \Pr(c) = \frac{10}{30}$$

Maximum Likelihood Estimation

- suppose instead we saw the following sequences

  gcgcttggcc

  gcttggctct

  cgttttgctc

- then the maximum likelihood estimates are

$$\Pr(g) = \frac{9}{30} \qquad \Pr(a) = \frac{0}{30}$$

$$\Pr(t) = \frac{11}{30} \qquad \Pr(c) = \frac{10}{30}$$

Do we really want to set this to 0? Maybe we just got unlucky …

Alternate Approach

- instead of estimating parameters strictly from the data, we could use *Laplace estimates* (also known as "add-one rule")

$$\Pr(a) = \frac{n_a + 1}{\sum_i (n_i + 1)}$$

pseudocount

- Bayesian interpretation for this "hack"
- Using Laplace estimates with the sequences

gcgcttggcc

gcttggctct

cgttttgctc

$$\Pr(a) = \frac{0+1}{34}$$

$$\Pr(c) = \frac{10+1}{34}$$

Now nothing is zeroed out

Maximum Likelihood Estimation

- suppose we want to estimate the parameters Pr(a), Pr(c), Pr(g), Pr(t)
- and we're given the sequences

  accgcgctta

  gcttagtgac

  tagccgttac

- then the maximum likelihood estimates are

$$\Pr(a) = \frac{6}{30} = 0.2 \qquad \Pr(g) = \frac{7}{30} = 0.233$$

$$\Pr(c) = \frac{9}{30} = 0.3 \qquad \Pr(t) = \frac{8}{30} = 0.267$$

Maximum Likelihood Estimation

- suppose instead we saw the following sequences

  gccgcgcttg

  gcttggtggc

  tggccgttgc

- then the maximum likelihood estimates are

$$\Pr(a) = \frac{0}{30} = 0 \qquad \Pr(g) = \frac{13}{30} = 0.433$$

$$\Pr(c) = \frac{9}{30} = 0.3 \qquad \Pr(t) = \frac{8}{30} = 0.267$$

But do we really want to set this to 0? Maybe we just got unlucky …

Alternate Approach

- instead of estimating parameters strictly from the data, we could use *Laplace estimates* (also known as "add-one rule")

$$\Pr(a) = \frac{n_a + 1}{\sum_i (n_i + 1)}$$

pseudocount

- Bayesian interpretation for this "hack"
- Using Laplace estimates with the sequences

gccgcgcttg
gcttggtggc
tggccgttgc

$$\Pr(a) = \frac{0+1}{34}$$

$$\Pr(c) = \frac{9+1}{34}$$

Now nothing is zeroed out