Pairwise Sequence Alignment

BMI/CS 576
www.biostat.wisc.edu/bmi576/
Mark Craven
craven@biostat.wisc.edu
Fall 2011

Pairwise alignment: task definition

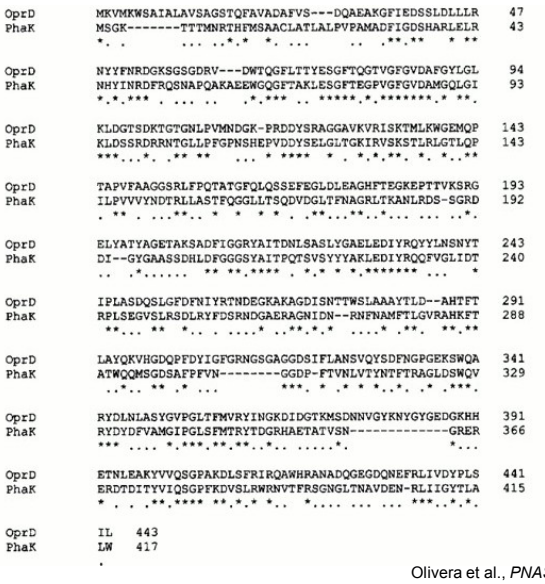
Given

- a pair of sequences (DNA or protein)
- a method for scoring a candidate alignment

Do

- determine the correspondences between substrings in the sequences such that the similarity score is maximized

Protein alignment example



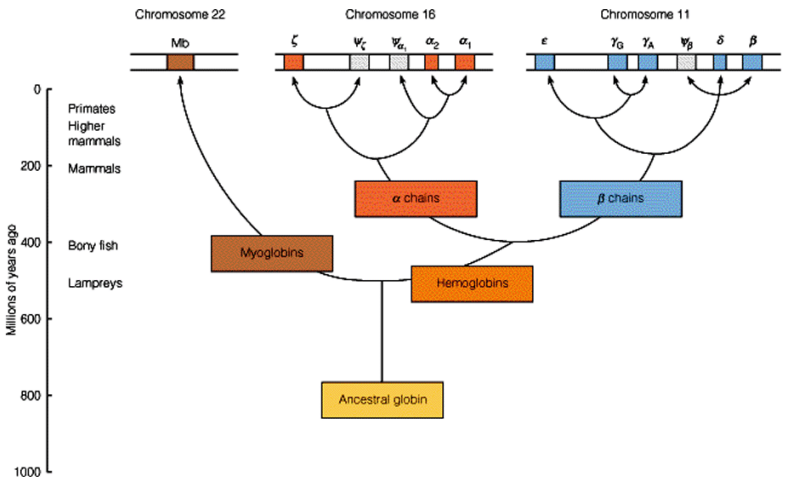
Alignment of the PhaK protein from Pseudomonas putida and OprD protein from Pseudomonas aeruginos

Olivera et al., *PNAS* 95:6419-6424, 1998

The role of homology in alignment

- *homology*: similarity due to descent from a common ancestor
- often we can infer homology from similarity
- thus we can sometimes infer structure/function from sequence similarity

Homology example: evolution of the globins



Homology

- homologous sequences can be divided into two groups
  - *orthologous sequences*: sequences that differ because they are found in different species (e.g. human α-globin and mouse α-globin)
  - *paralogous sequences*: sequences that differ because of a gene duplication event (e.g. human α-globin and human β-globin, various versions of both)

DNA sequence edits

- substitutions: A**C**GA ➡ A**G**GA
- insertions: ACGA ➡ AC**CGGA**GA
- deletions: A**CGGA**GA ➡ AGA
- transpositions: A**CGG**AGA ➡ AAG**CGG**A
- inversions: AC**GGA**GA ➡ AC**TCC**GA

Mismatches and gaps

- substitutions in *homologous* sequences result in mismatches in an alignment
- insertions/deletions in *homologous* sequences result in mismatches in an alignment

```
CA--GATTCGAAT
CGCCGATT---AT
```

mismatch

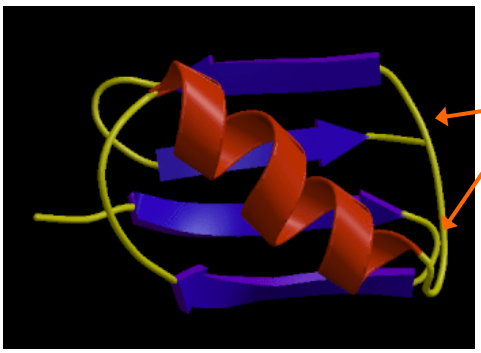
gap

Alignment scales

- for short DNA sequences (gene scale) we will generally only consider
  - substitutions
  - insertions/deletions
- for longer DNA sequences (genome scale) we will consider additional events
  - transpositions
  - inversions
- in this course we will focus on the case of short sequences

Insertions/deletions and protein structure

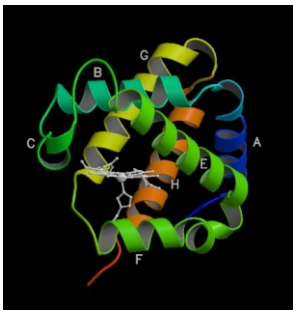
- Why is it that two "similar" sequences may have large insertions/deletions?
  - some insertions and deletions may not significantly affect the structure of a protein

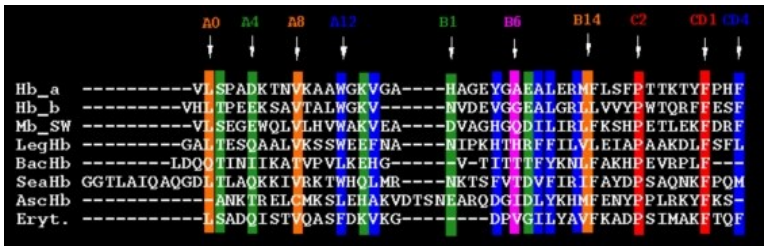


loop structures: insertions/deletions here not so significant

Example alignment: globins

- figure at right shows prototypical structure of globins
- figure below shows part of alignment for 8 globins





Issues in sequence alignment

- the sequences we're comparing typically differ in length
- there may be only a relatively small region in the sequences that matches
- we want to allow partial matches (i.e. some amino acid pairs are more substitutable than others)
- variable length regions may have been inserted/deleted from the common ancestral sequence

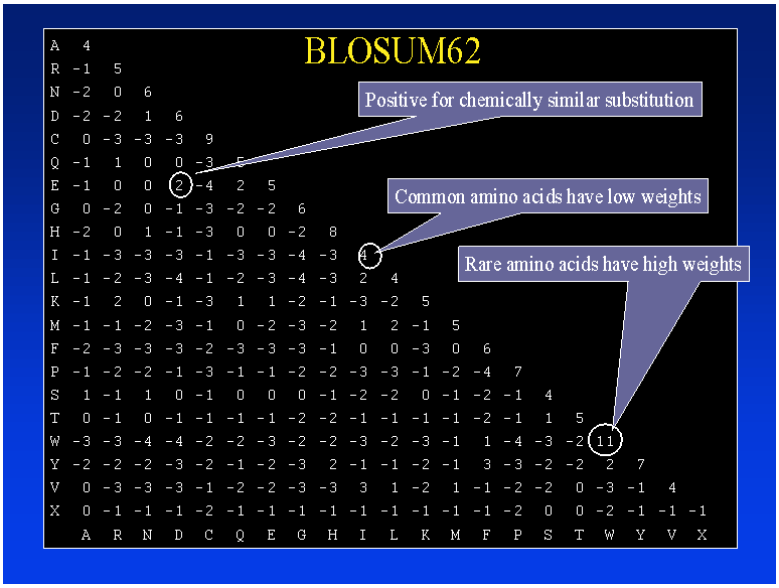
Types of alignment

- *global*: find best match of both sequences in their entirety
- *local*: find best subsequence match
- *semi-global*: find best match without penalizing gaps on the ends of the alignment

Scoring an alignment: what is needed?

- substitution matrix
  - *s(a,b)* indicates score of aligning character *a* with character *b*
- gap penalty function
  - *w(g)* indicates cost of a gap of length *g*

Blosum 62 substitution matrix



Linear gap penalty function

- different gap penalty functions require somewhat different dynamic programming algorithms
- the simplest case is when a linear gap function is used

$$w(g) = -g \times d$$

where d is a constant

- we'll start by considering this case

Scoring an alignment

- the score of an alignment is the sum of the scores for pairs of aligned characters plus the scores for gaps
- example: given the following alignment

```
VAHV---D--DMPNALSALSDLHAHKL
AIQLQVTGVVVTDATLKNLGSVHVSKG
```

- we would score it by

  s(**V**,**A**) + s(**A**,**I**) + s(**H**,**Q**) + s(**V**,**L**) – 3*d* + s(**D**,**G**) – 2d

  …

The space of global alignments

- some possible global alignments for `ELV` and `VIS`

```
ELV      -ELV      --ELV      ELV-
VIS      VIS-      VIS--      -VIS

E-LV     ELV--     EL-V
VIS-     --VIS     -VIS
```

- Can we find the highest scoring alignment by enumerating all possible alignments and picking the best?

Number of possible alignments

- given sequences of length *m* and *n*
- assume we don't count as distinct $\begin{matrix} C- \\ -G \end{matrix}$ and $\begin{matrix} -C \\ G- \end{matrix}$
- we can have as few as 0 and as many as min{*m*, *n*} aligned pairs
- therefore the number of possible alignments is given by

$$\sum_{k=0}^{\min\{m,n\}} \binom{n}{k}\binom{m}{k} = \binom{n+m}{n}$$

Number of possible alignments

- there are

$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \approx \frac{2^{2n}}{\sqrt{\pi n}}$$

possible global alignments for 2 sequences of length *n*

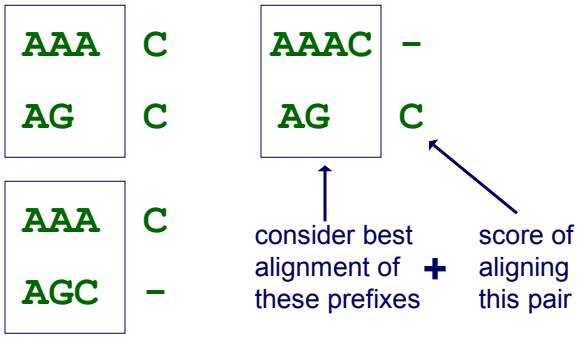
- e.g. two sequences of length 100 have $\approx 10^{77}$ possible alignments
- but we can use *dynamic programming* to find an optimal alignment efficiently

Pairwise alignment via dynamic programming

- first algorithm by Needleman & Wunsch, *Journal of Molecular Biology*, 1970
- *dynamic programming*: solve an instance of a problem by taking advantage of computed solutions for smaller subparts of the problem
- determine best alignment of two sequences by determining best alignment of all prefixes of the sequences

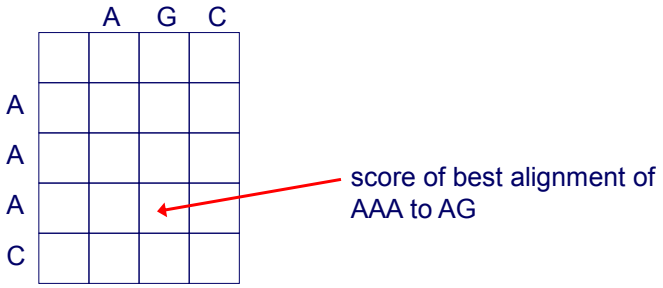
Dynamic programming idea

- consider last step in computing alignment of **AAAC** with **AGC**
- three possible options; in each we'll choose a different pairing for end of alignment, and add this to best alignment of previous characters



Dynamic programming idea

- given an *n*-character sequence *x,* and an *m*-character sequence *y*
- construct an (*n*+1) $\times$ (*m*+1) matrix *F*
- *F* (*i, j*) = score of the best alignment of *x*[1…*i*] with *y*[1…*j*]

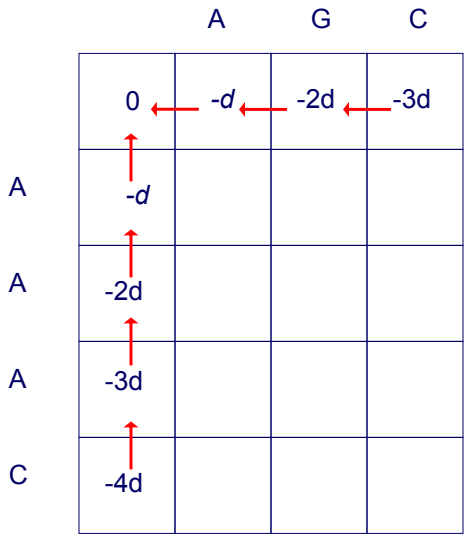


DP algorithm for global alignment with linear gap penalty

- one way to specify the DP is in terms of its recurrence relation:

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + s(x_i, y_j) \\ F(i-1,j) - d \\ F(i,j-1) - d \end{cases}$$

Initializing matrix: global alignment with linear gap penalty



DP algorithm sketch: global alignment

- initialize first row and column of matrix
- fill in rest of matrix from top to bottom, left to right
- for each $F(i, j)$, save pointer(s) to cell(s) that resulted in best score
- $F(m, n)$ holds the optimal alignment score; trace pointers back from $F(m, n)$ to $F(0, 0)$ to recover alignment

Global alignment example

- suppose we choose the following scoring scheme:

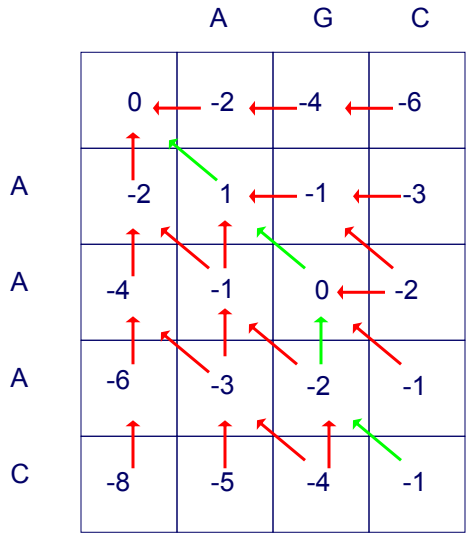
$s(x_i, y_i) =$

+1 when $x_i = y_i$

-1 when $x_i \neq y_i$

d (penalty for aligning with a gap) = 2

Global alignment example



one optimal alignment

x: A A A C

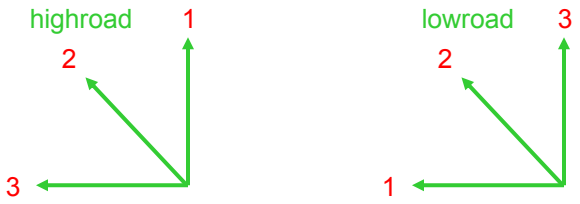
y: A G - C

DP comments

- works for either DNA or protein sequences, although the substitution matrices used differ
- finds an optimal alignment
- the exact algorithm (and computational complexity) depends on gap penalty function (we'll come back to this issue)

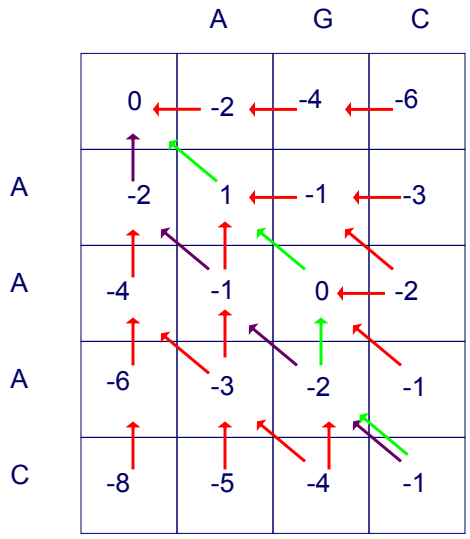
Equally optimal alignments

- many optimal alignments may exist for a given pair of sequences
- can use preference ordering over paths when doing traceback



- *highroad* and *lowroad* alignments show the two most different optimal alignments

Highroad & lowroad alignments



highroad alignment

x: A A A C

y: A G - C

lowroad alignment

x: A A A C

y: - A G C

Computational complexity

- initialization: $O(m)$, $O(n)$ where sequence lengths are m, n
- filling in rest of matrix: $O(mn)$
- traceback: $O(m + n)$
- hence, if sequences have nearly same length, the computational complexity is

$$O(n^2)$$

Related problems solved by DP

- **Local** alignment
  - the best match between subsequences of *x* and *y*
  - so far we have discussed *global alignment*, where we are looking for best match between sequences from one end to the other
- More realistic **gap functions**
  - a gap of length *k* is more probable than *k* gaps of length 1
  - a gap may be due to a single mutational event that inserted/deleted a stretch of characters
  - separated gaps are probably due to distinct mutational events

Local alignment

- Motivation
  - a common *motif* (conserved pattern) or *domain* (independently folded unit) but differ elsewhere
  - more sensitive when comparing highly diverged sequences
- Original formulation
  - Smith & Waterman, *Journal of Mol. Biology*, 1981
- Implementation
  - the recurrence relation is slightly different than for global algorithm
    - maximize also with 0
    - begins and ends anywhere

Gap penalty functions

- linear: $w(g) = -g \times d$
- affine: $$w(g) = \begin{cases} -d - (g-1)e, & g \geq 1 \\ 0, & g = 0 \end{cases}$$
- convex: as gap length increases, magnitude of penalty for each additional character decreases

  e.g. $w(g) = -d - \log(g) \times e$

Pairwise alignment summary

- the number of possible alignments is exponential in the length of sequences being aligned
- dynamic programming can find optimal-scoring alignments in polynomial time
- the specifics of the DP depend on
  - local vs. global alignment
  - gap penalty function
- affine penalty functions are most commonly used