Pairwise Sequence Alignment (Continued)

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

Local alignment

- so far we have discussed *global alignment*, where we are looking for best match between sequences from one end to the other

- often we want a *local alignment*, the best match between subsequences of x and y

Example local alignment

- aligning my name against the sequence for dTDP-4-dehydrorhamnose reductase from the bacterium *opitutus terrae*

```
                              MARKCRAVEN
…LSGAYHLAASGHTSWHGFASAIIDLMPLDARKCRAVEAIT…
```

Local alignment motivation

- useful for comparing protein sequences that share a common *motif* (conserved pattern) or *domain* (independently folded unit) but differ elsewhere
- useful for comparing DNA sequences that share a similar *motif* but differ elsewhere
- useful for comparing protein sequences against *genomic DNA sequences* (long stretches of uncharacterized sequence)
- more sensitive when comparing highly diverged sequences

Local alignment DP algorithm

- original formulation: Smith & Waterman, *Journal of Molecular Biology*, 1981

- interpretation of array values is somewhat different: $F(i, j)$ = score of the best alignment of a suffix of $x[1\ldots i]$ and a suffix of $y[1\ldots j]$

Local alignment DP algorithm

- the recurrence relation is slightly different than for global algorithm

$$F(i,j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \\ 0 \end{cases}$$

Local alignment DP algorithm

- initialization: first row and first column initialized with 0's
- traceback:
  - find maximum value of *F(i, j)*; can be <u>anywhere</u> in matrix
  - stop when we get to a cell with value 0

Local alignment example

		A	A	G	A
	0	0	0	0	0
T	0	0	0	0	0
T	0	0	0	0	0
A	0	1	1	0	1
A	0	1	2	0	1
G	0	0	0	$\boxed{3}$	1

x: A A G
y: A A G

More on gap penalty 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

- a linear gap penalty function treats these cases the same

- it is more common to use gap penalty functions involving two terms
  - a penalty *d* associated with <u>opening</u> a gap
  - a smaller penalty *e* for <u>extending</u> the gap

Gap penalty functions

linear

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

affine

$$w(g) = \begin{cases} -d - (g-1)e, & g \geq 1 \\ 0, & g = 0 \end{cases}$$

Dynamic programming for the affine gap penalty case

- to do in $O(n^2)$ time, need 3 matrices instead of 1

$M(i, j)$	best score given that *x*[*i*] is aligned to *y*[*j*]
$I_x(i, j)$	best score given that *x*[*i*] is aligned to a gap
$I_y(i, j)$	best score given that *y*[*j*] is aligned to a gap

Global alignment DP for the affine gap penalty case

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

$$I_x(i,j) = \max \begin{cases} M(i-1,j) - d \\ I_x(i-1,j) - e \end{cases}$$

$$I_y(i,j) = \max \begin{cases} M(i,j-1) - d \\ I_y(i,j-1) - e \end{cases}$$

Global alignment DP for the affine gap penalty case

- initialization

  $M(0,0) = 0$

  $I_x(i,\ 0) = -d - (i-1)e$ for $i > 0$

  $I_y(0, j) = -d - (j-1)e$ for $j > 0$

  other cells in top row and leftmost column $= -\infty$

- traceback
  - start at largest of $M(m,n), I_x(m,n), I_y(m,n)$
  - stop at $M(0,0)$
  - note that pointers may traverse all three matrices

Global alignment example
(affine gap penalty)

d = 4, e = 1

		A	C	A	C	T
	M : 0 I_x : -∞ I_y : -∞	-∞ -∞ -4	-∞ -∞ -5	-∞ -∞ -6	-∞ -∞ -7	-∞ -∞ -8
A	-∞ -4 -∞	1 -∞ -∞	-5 -∞ -3	-4 -∞ -4	-7 -∞ -5	-8 -∞ -6
A	-∞ -5 -∞	-3 -3 -∞	0 -9 -7	-2 -8 -4	-5 -11 -5	-6 -12 -6
T	-∞ -6 -∞	-6 -4 -∞	-4 -4 -10	-1 -6 -8	-3 -9 -5	-4 -10 -6

Global alignment example (continued)

		A	C	A	C	T
	M : 0 I_x : -∞ I_y : -∞	-∞ -∞ -4	-∞ -∞ -5	-∞ -∞ -6	-∞ -∞ -7	-∞ -∞ -8
A	-∞ -4 -∞	1 -∞ -∞	-5 -∞ -3	-4 -∞ -4	-7 -∞ -5	-8 -∞ -6
A	-∞ -5 -∞	-3 -3 -∞	0 -9 -7	-2 -8 -4	-5 -11 -5	-6 -12 -6
T	-∞ -6 -∞	-6 -4 -∞	-4 -4 -10	-1 -6 -8	-3 -9 -5	-4 -10 -6

three optimal alignments:

```
ACACT    ACACT    ACACT
AA--T    A--AT    --AAT
```

Why three matrices are needed

- consider aligning the sequences **WFP** and **FW** using d = 5, e = 1 and the following values from the BLOSUM-62 substitution matrix:

  s(**F**, **W**) = 1   s(**W**, **W**) = 11
  s(**F**, **F**) = 6   s(**W**, **P**) = -4
  s(**F**, **P**) = -4

- the matrix shows the highest-scoring partial alignment for each pair of prefixes

		W	F	P
	0	-5	-6	-7
F	-5	1	1	-4
W	-6	6	2	0

```
-WFP
FW--
```
optimal alignment

```
WF
FW
```
best alignment of these prefixes; to get optimal alignment, need to also remember

```
-WF
FW-
```

Local alignment DP for the affine gap penalty case

$$M(i,j) = \max \begin{cases} M(i-1, j-1) + s(x_i, y_j) \\ I_x(i-1, j-1) + s(x_i, y_j) \\ I_y(i-1, j-1) + s(x_i, y_j) \\ 0 \end{cases}$$

$$I_x(i,j) = \max \begin{cases} M(i-1, j) - d \\ I_x(i-1, j) - e \end{cases}$$

$$I_y(i,j) = \max \begin{cases} M(i, j-1) - d \\ I_y(i, j-1) - e \end{cases}$$

Local alignment DP for the affine gap penalty case

- initialization

  $M(0,0) = 0$

  $M(i,0) = 0$

  $M(0,j) = 0$

  cells in top row and leftmost column of $I_x, I_y = -\infty$

- traceback
  - start at largest $M(i,j)$
  - stop at $M(i,j) = 0$

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

Computational complexity and gap penalty functions

linear: $O(n^2)$

affine: $O(n^2)$

convex: $O(n^2 \log n)$

general: $O(n^3)$

* assuming two sequences of length *n*

Alignment (global) with general gap penalty function

why the general case has time complexity $O(n^3)$

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

consider every previous element in the column

k ranges over previous coordinates

consider every previous element in the row

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