

Affine Gap Penalties

Sushmita Roy

BMI/CS 576

www.biostat.wisc.edu/bmi576/

Sushmita Roy

sroy@biostat.wisc.edu

Sep 25th, 2012

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 opening a gap
 - a smaller penalty e for extending the gap

Global Alignment with general gap penalty function

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

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

k ranges over previous coordinates

consider every previous element in the row

consider every previous element in the column

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 (move in vertical direction)

$I_y(i, j)$

best score given that y_j is
aligned to a gap (move in horizontal direction)

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

$$M(0,0) = 0$$

$$I_x(i, 0) = -d - (i-1)e \quad \text{for } i > 0$$

$$I_y(0, j) = -d - (j-1)e \quad \text{for } j > 0$$

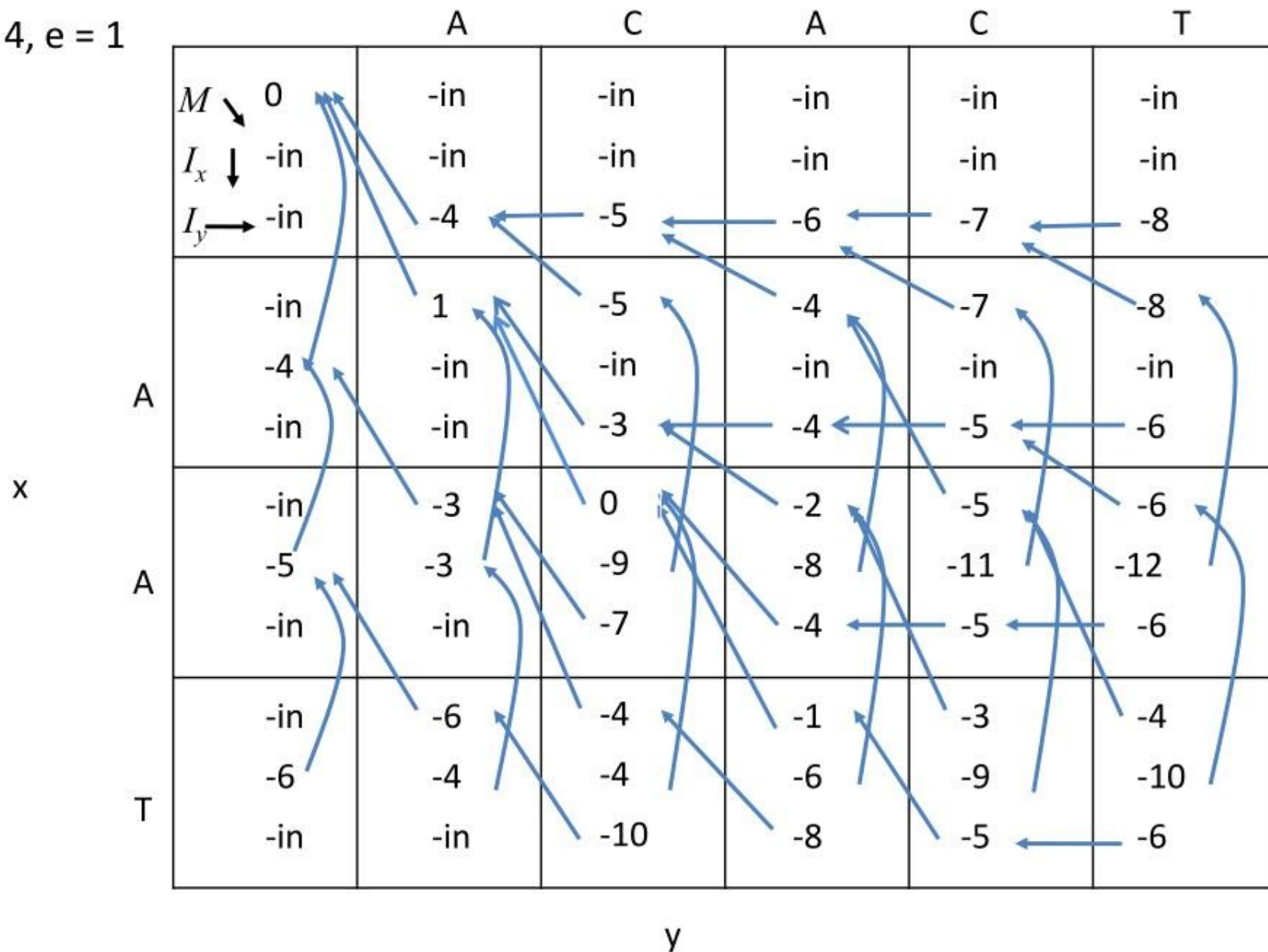
other cells in top row and leftmost column $= -\infty$

- traceback

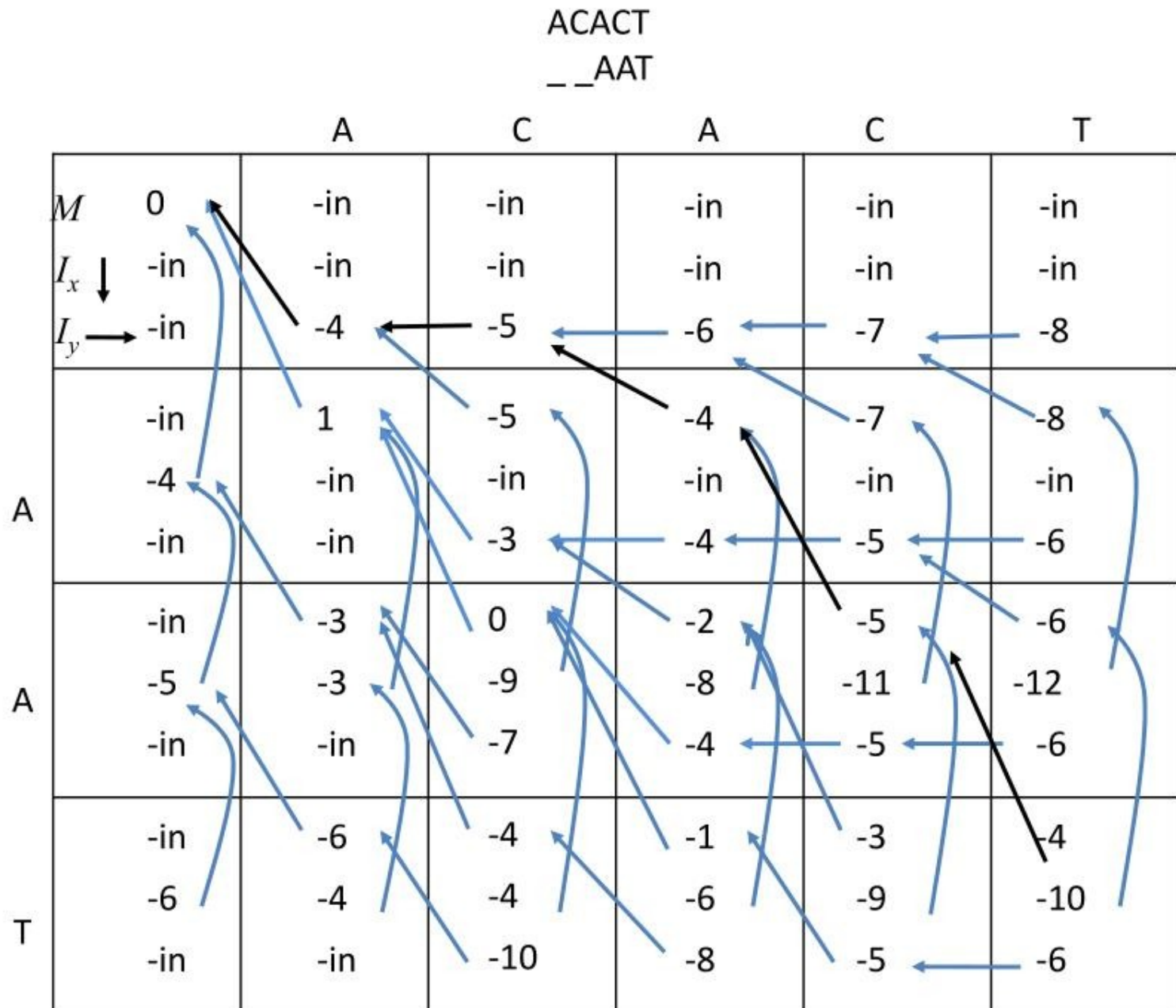
- start at largest of $M(m,n), I_x(m,n), I_y(m,n)$
- stop at $M(0,0)$
- note that pointers may traverse all three matrices

Global alignment example (affine gap penalty)

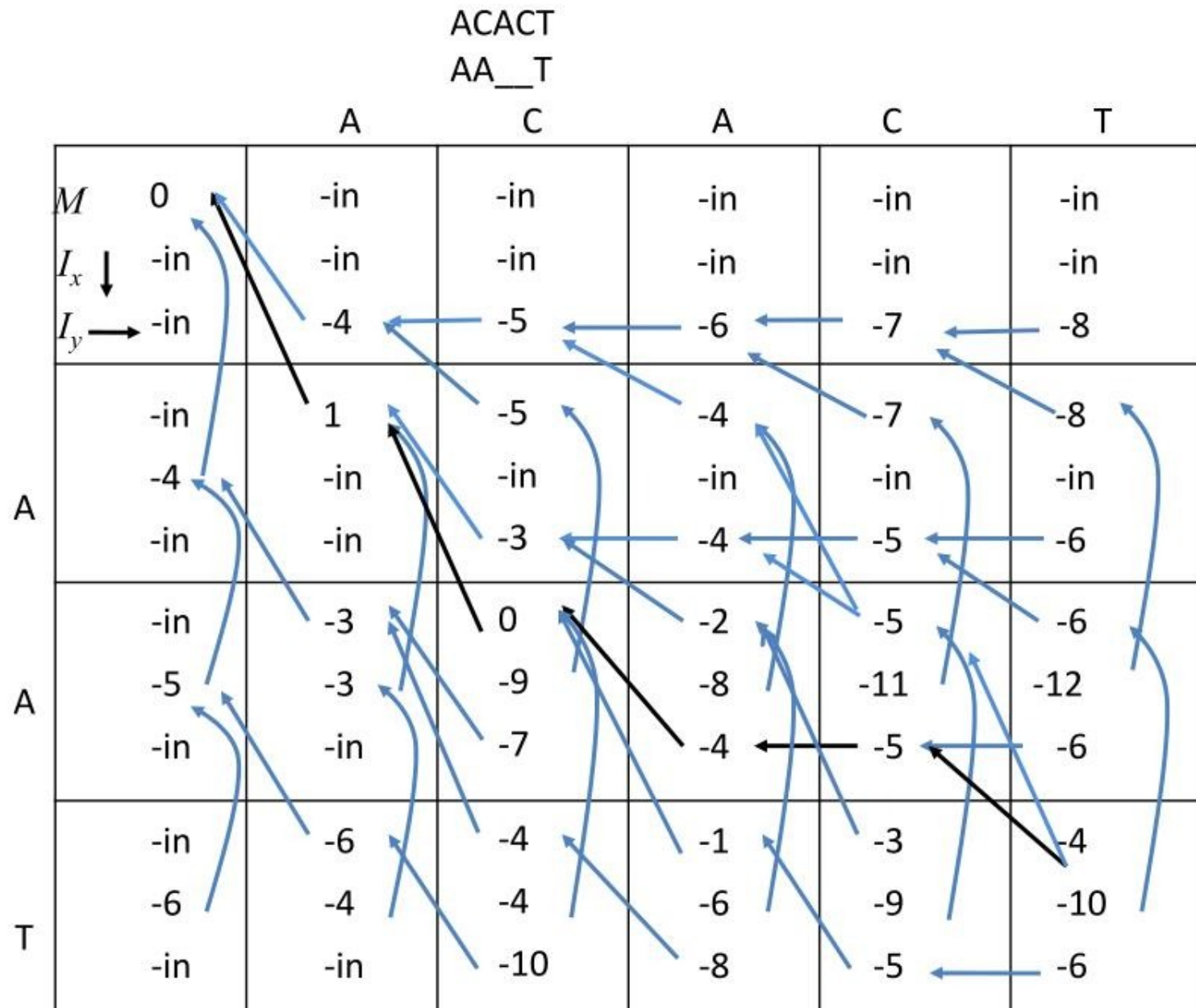
$d = 4, e = 1$



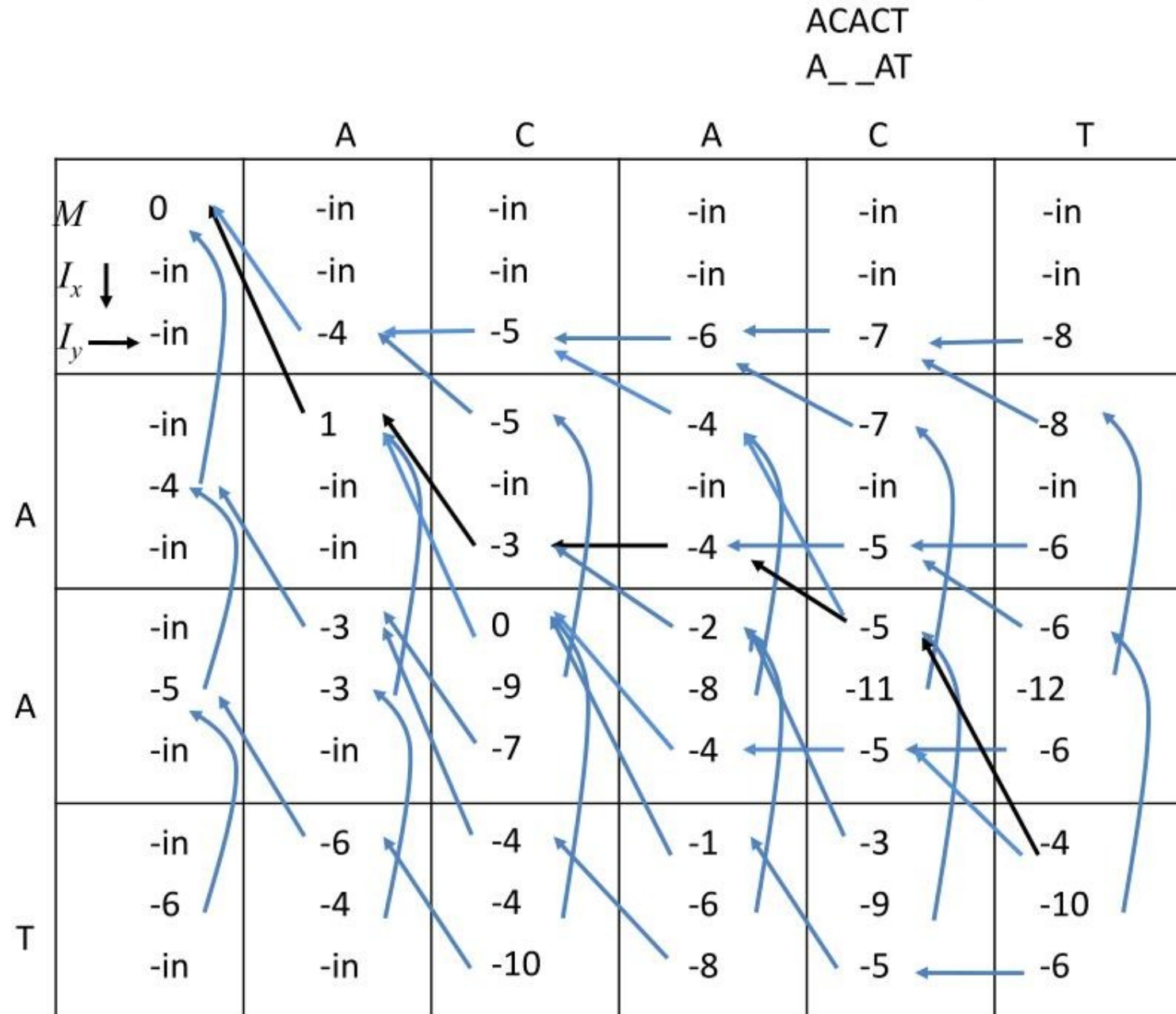
Global alignment example (affine gap penalty)



Global alignment example (affine gap penalty)



Global alignment example (affine gap penalty)



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: $O(n^2)$

general: $O(n^3)$

assuming two sequences of length n

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