Local alignment

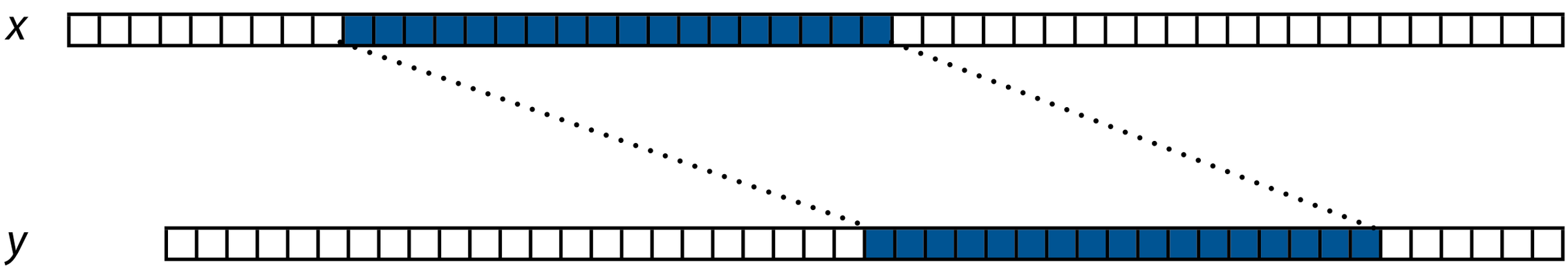
Ben Langmead

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Department of Computer Science

Local alignment

Given strings *x* and *y*, what is the optimal global alignment value of a *substring* of *x* to a *substring* of *y*. This is *local alignment*.



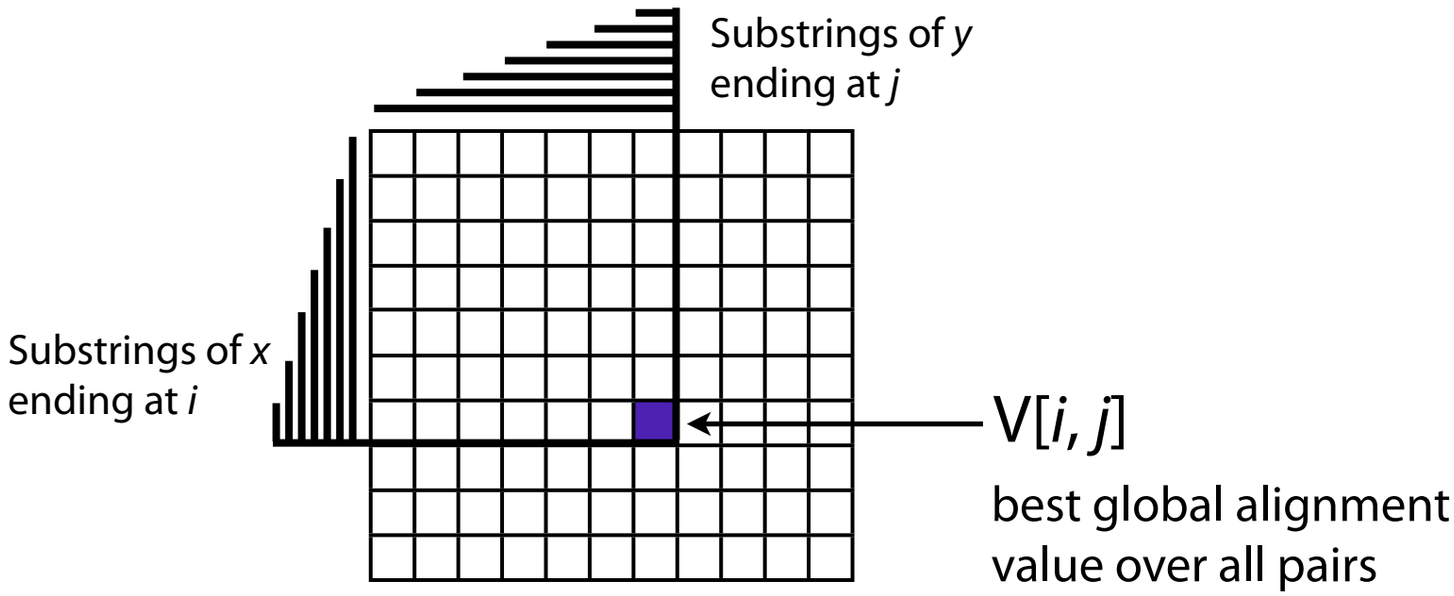
Assume global alignment scoring where: (a) similarities get > 0, (b) dissimilarities get < 0, (c) alignment of ϵ to any string has score 0

Somehow we must weigh *all possible pairs* of substrings

What is bound for # substring pairs, assuming $|x| = n$, $|y| = m$? $O(m^2n^2)$

Local alignment

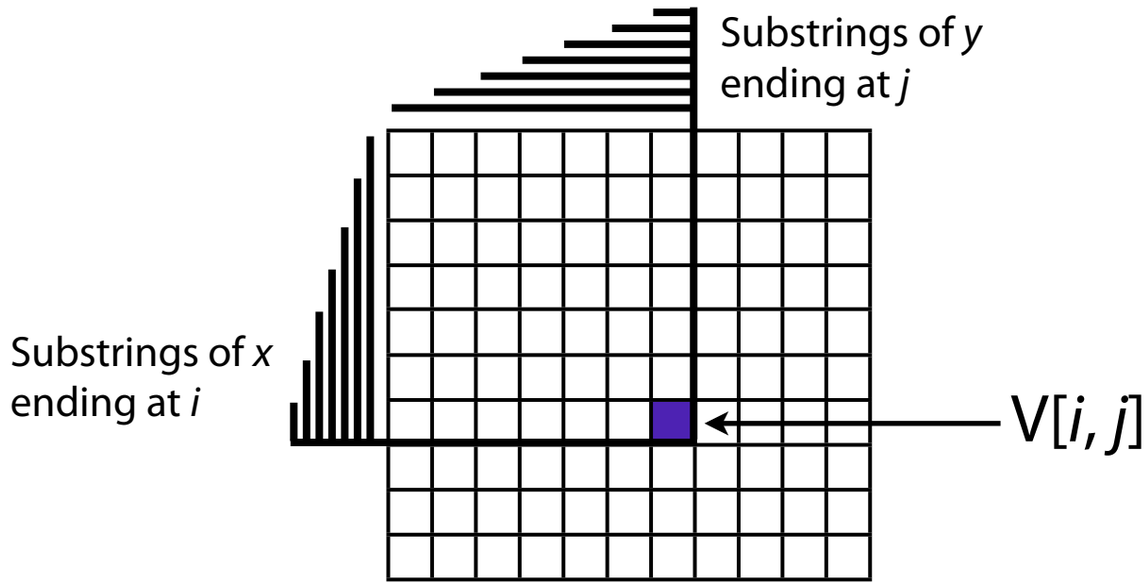
Let V[*i*, *j*] be the optimal global alignment value of a substring of *x* ending at *i* and a substring of *y* ending at *j*. The substrings may be empty.



The maximum V[*i*, *j*] over all *i*, *j* is the optimal score we're looking for

Local alignment

How to calculate V[*i*, *j*]?



Only 4 ways to build a new edit transcript from another one:

Vertical: append **I** to transcript for V[*i*-1, *j*], take gap penalty

Horizontal: append **D** to transcript for V[*i*, *j*-1], take gap penalty

Diagonal: append **M** or **R** to transcript for V[*i*-1, *j*-1], get match bonus or take replacement penalty as appropriate

Empty: let both substrings be empty, global alignment value = 0

Proof: Gusfield 11.7.1 - 11.7.2

Local alignment

Let $V[0, j] = 0$, and let $V[i, 0] = 0$

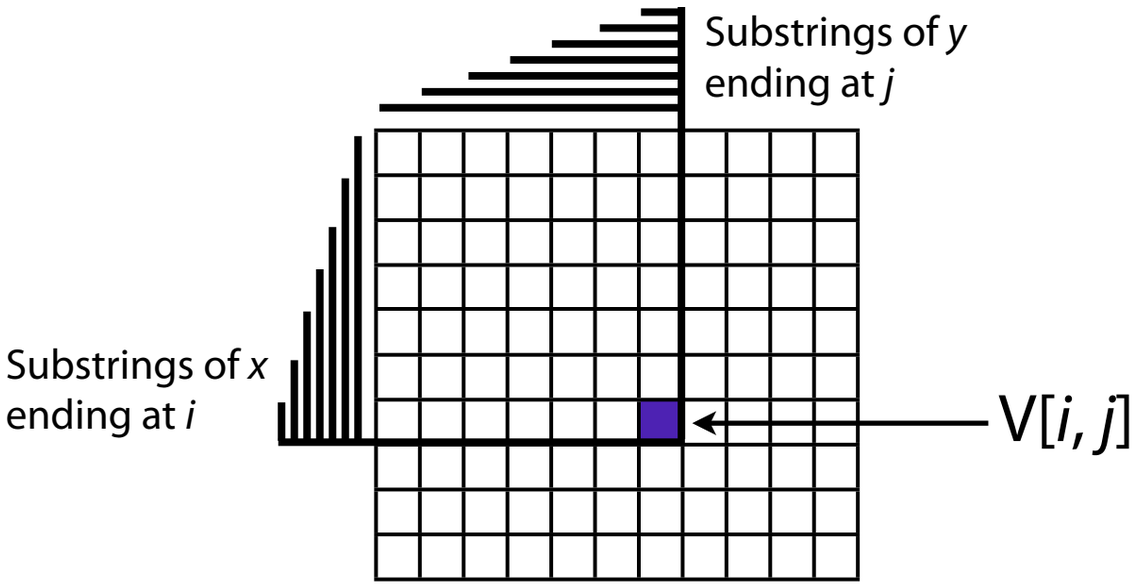
Otherwise, let $V[i, j] = \max \begin{cases} V[i-1, j] + s(x[i-1], -) \\ V[i, j-1] + s(-, y[j-1]) \\ V[i-1, j-1] + s(x[i-1], y[j-1]) \\ 0 \end{cases}$

$s(a, b)$ assigns a score to a particular match, gap, or replacement

What's different from global alignment?

- First row and columns initialized to all 0s
- 0 is one of the arguments of the max

Local alignment: Smith-Waterman



	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

Let $V[0, j] = 0$, and let $V[i, 0] = 0$

Otherwise, let $V[i, j] = \max \begin{cases} V[i-1, j] + s(x[i-1], -) \\ V[i, j-1] + s(-, y[j-1]) \\ V[i-1, j-1] + s(x[i-1], y[j-1]) \\ 0 \end{cases}$

$s(a, b)$ assigns a score to a particular match, gap, or replacement

Local alignment: Smith-Waterman

Does it make sense that first row and column get all 0s?

Yes, b/c global alignment value of ϵ, ϵ (0) always best

		Y													
		ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
X	ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	G	0														
	G	0														
	T	0														
	A	0														
	T	0														
	G	0														
	C	0														
	T	0														
	G	0														
	G	0														
	C	0														
	G	0														
	C	0														
	T	0														
	A	0														

$s(a, b)$

	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

Local alignment: Smith-Waterman

$$V[i,j] = \max \begin{cases} V[i-1,j] + s(x[i-1], -) \\ V[i,j-1] + s(-, y[j-1]) \\ V[i-1,j-1] + s(x[i-1], y[j-1]) \\ 0 \end{cases}$$

	ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	0	2	2	0	2	0	0	0
G	0	0	0	0	0	0	2	0	2	4	0	2	0	0	0
T	0	2	0	2	0	2	0	0	0	0	0	0	4	2	2
A	0	0	4	0	?										
T	0														
G	0														
C	0														
T	0														
G	0														
G	0														
C	0														
G	0														
C	0														
T	0														
A	0														

$s(a,b)$

	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Local alignment: Smith-Waterman

$$V[i,j] = \max \begin{cases} V[i-1,j] + s(x[i-1], -) \\ V[i,j-1] + s(-, y[j-1]) \\ V[i-1,j-1] + s(x[i-1], y[j-1]) \\ 0 \end{cases}$$

	ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	0	2	2	0	2	0	0	0
G	0	0	0	0	0	0	2	0	2	4	0	2	0	0	0
T	0	2	0	2	0	2	0	0	0	0	0	0	4	2	2
A	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0
T	0	2	0	6	0	6	0	0	0	0	0	0	2	2	2
G	0	0	0	0	2	0	8	2	2	2	0	2	0	0	0
C	0	0	0	0	0	0	2	10	4	0	4	0	0	0	0
T	0	2	0	2	0	2	0	4	6	0	0	0	2	2	2
G	0	0	0	0	0	0	4	0	6	8	2	2	0	0	0
G	0	0	0	0	0	0	2	0	2	8	4	4	0	0	0
C	0	0	0	0	0	0	0	4	0	2	10	4	0	0	0
G	0	0	0	0	0	0	2	0	6	2	4	12	6	0	0
C	0	0	0	0	0	0	0	4	0	2	4	6	8	2	0
T	0	2	0	2	0	2	0	0	0	0	0	0	8	10	4
A	0	0	4	0	4	0	0	0	0	0	0	0	2	4	6

$s(a,b)$

	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

0's in essence allow peaks of similarity to rise above "background" of 0s

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Local alignment: Smith-Waterman

Backtrace: (a) start from *maximal* cell in the matrix, (b) stop backtrace when we reach a cell with score = 0

	ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	0	2	2	0	2	0	0	0
G	0	0	0	0	0	0	2	0	2	4	0	2	0	0	0
T	0	2	0	2	0	2	0	0	0	0	0	0	4	2	2
A	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0
T	0	2	0	6	0	6	0	0	0	0	0	0	2	2	2
G	0	0	0	0	2	0	8	2	2	2	0	2	0	0	0
C	0	0	0	0	0	0	2	10	4	0	4	0	0	0	0
T	0	2	0	2	0	2	0	4	6	0	0	0	2	2	2
G	0	0	0	0	0	0	4	0	6	8	2	2			
G	0	0	0	0	0	0	2	0	2	8	4	4			
C	0	0	0	0	0	0	0	4	0	2	10	4			
G	0	0	0	0	0	0	2	0	6	2	4	12	6	0	0
C	0	0	0	0	0	0	0	4	0	2	4	6	8	2	0
T	0	2	0	2	0	2	0	0	0	0	0	0	8	10	4
A	0	0	4	0	4	0	0	0	0	0	0	0	2	4	6

$s(a, b)$

	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

```
y:  TATATGC-GGCGTTT
    | | | | | | | | | |
x:  GGTATGCTGGCGCTA
```

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Local alignment: Smith-Waterman

What if we didn't have a positive "bonus" for matches?

All cells would = 0

	ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	0	2	2	0	2	0	0	0
G	0	0	0	0	0	0	2	0	2	4	0	2	0	0	0
T	0	2	0	2	0	2	0	0	0	0	0	0	4	2	2
A	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0
T	0	2	0	6	0	6	0	0	0	0	0	0	2	2	2
G	0	0	0	0	2	0	8	2	2	2	0	2	0	0	0
C	0	0	0	0	0	0	2	10	4	0	4	0	0	0	0
T	0	2	0	2	0	2	0	4	6	0	0	0	2	2	2
G	0	0	0	0	0	0	4	0	6	8	2	2	0	0	0
G	0	0	0	0	0	0	2	0	2	8	4	4	0	0	0
C	0	0	0	0	0	0	0	4	0	2	10	4	0	0	0
G	0	0	0	0	0	0	2	0	6	2	4	12	6	0	0
C	0	0	0	0	0	0	0	4	0	2	4	6	8	2	0
T	0	2	0	2	0	2	0	0	0	0	0	0	8	10	4
A	0	0	4	0	4	0	0	0	0	0	0	0	2	4	6

$s(a, b)$

	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

What if we didn't have negative "penalties" for edits?

Rule for ϵ, ϵ would never be used and alignment would essentially be global

$$\max \begin{cases} V[i-1,j] + s(x[i-1], -) \\ V[i,j-1] + s(-, y[j-1]) \\ V[i-1,j-1] + s(x[i-1], y[j-1]) \\ 0 \end{cases}$$

Local alignment: Smith-Waterman

```
def smithWaterman(x, y, s):
    """ Calculate local alignment values of sequences x and y using
        dynamic programming.  Return maximal local alignment value. """
    V = numpy.zeros((len(x)+1, len(y)+1), dtype=int)
    for i in xrange(1, len(x)+1):
        for j in xrange(1, len(y)+1):
            V[i, j] = max(V[i-1, j-1] + s(x[i-1], y[j-1]), # diagonal
                          V[i-1, j  ] + s(x[i-1], '-'),    # vertical
                          V[i  , j-1] + s('-',    y[j-1]), # horizontal
                          0)                               # empty
    argmax = numpy.where(V == V.max())
    return int(V[argmax])
```

Python example: http://nbviewer.ipython.org/6994170

Local alignment: Smith-Waterman

We might be interested in the *best* local alignment, or in many *good-enough* local alignments

	ϵ	T	A	T	A	T	G	C	G	G	C	G	T	T	T
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	0	2	2	0	2	0	0	0
G	0	0	0	0	0	0	2	0	2	4	0	2	0	0	0
T	0	2	0	2	0	2	0	0	0	0	0	0	4	2	2
A	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0
T	0	2	0	6	0	6	0	0	0	0	0	0	2	2	2
G	0	0	0	0	2	0	8	2	2	2	0	2	0	0	0
C	0	0	0	0	0	0	2	10	4	0	4	0	0	0	0
T	0	2	0	2	0	2	0	4	6	0	0	0	2	2	2
G	0	0	0	0	0	0	4	0	6	8	2	2	0	0	0
G	0	0	0	0	0	0	2	0	2	8	4	4	0	0	0
C	0	0	0	0	0	0	0	4	0	2	10	4	0	0	0
G	0	0	0	0	0	0	2	0	6	2	4	12	6	0	0
C	0	0	0	0	0	0	0	4	0	2	4	6	8	2	0
T	0	2	0	2	0	2	0	0	0	0	0	0	8	10	4
A	0	0	4	0	4	0	0	0	0	0	0	0	2	4	6

Reducing *good-enough* threshold risks allowing lots of tiny alignments that aren't very relevant