Local Alignment

Ben Langmead

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Department of Computer Science

Please sign guestbook (www.langmead-lab.org/teaching-materials) to tell me briefly how you are using the slides. For original Keynote files, email me (ben.langmead@gmail.com).

Global alignment revisited

Y

		ϵ	T	A	T	G	T	C	A	T	G	C
	ϵ	0	8	16	24	32	40	48	56	64	72	80
	T	8	0	8	16	24	32	40	48	56	64	72
	A	16	8	0	8	16	24	32	40	48	56	64
	C	24	16	8	2	10	18	24	32	40	48	56
X	G	32	24	16	10	2	10	18	26	34	40	48
	T	40	32	24	16	10	2	10	18	26	34	42
	C	48	40	32	24	18	10	2	10	18	26	34
	A	56	48	40	32	26	18	10	2	10	18	26
	G	64	56	48	40	32	26	18	10	6	10	18
	C	72	64	56	48	40	34	26	18	12	10	10

$s(a, b)$

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Optimal global alignment value

Global alignment revisited

$s(a, b)$

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Could also use *larger* scores for similarities and *smaller* scores for dissimilarities...

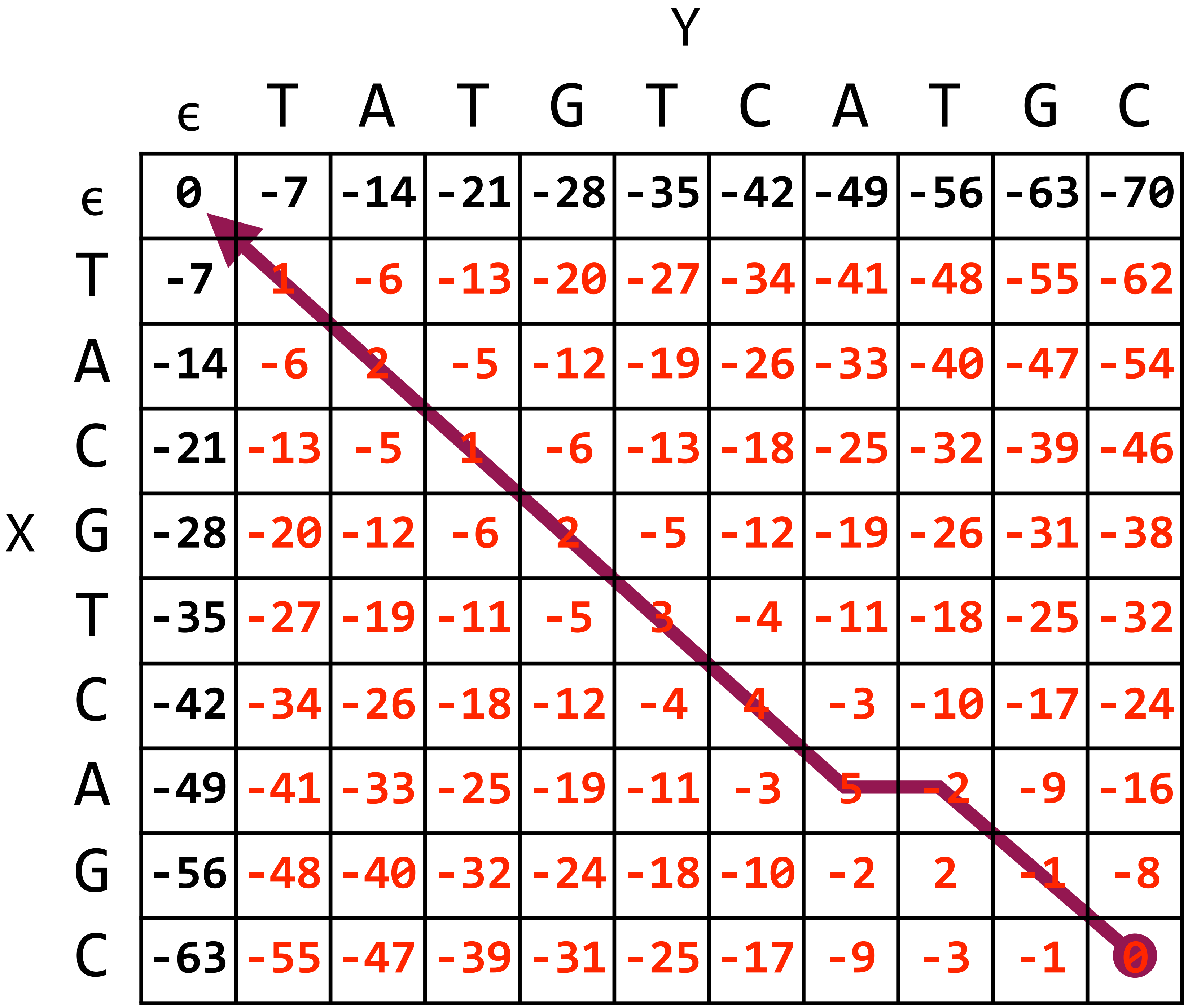
E.g. subtract one then change sign

	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

...as long as we switch min to max:

```
for i in range(1, len(x)+1):
    for j in range(1, len(y)+1):
        D[i, j] = max(D[i-1, j-1] + s(x[i-1], y[j-1]), # diagonal
                      D[i-1, j  ] + s(x[i-1], '-'),    # vertical
                      D[i  , j-1] + s('-',    y[j-1])) # horizontal
```

Global alignment revisited



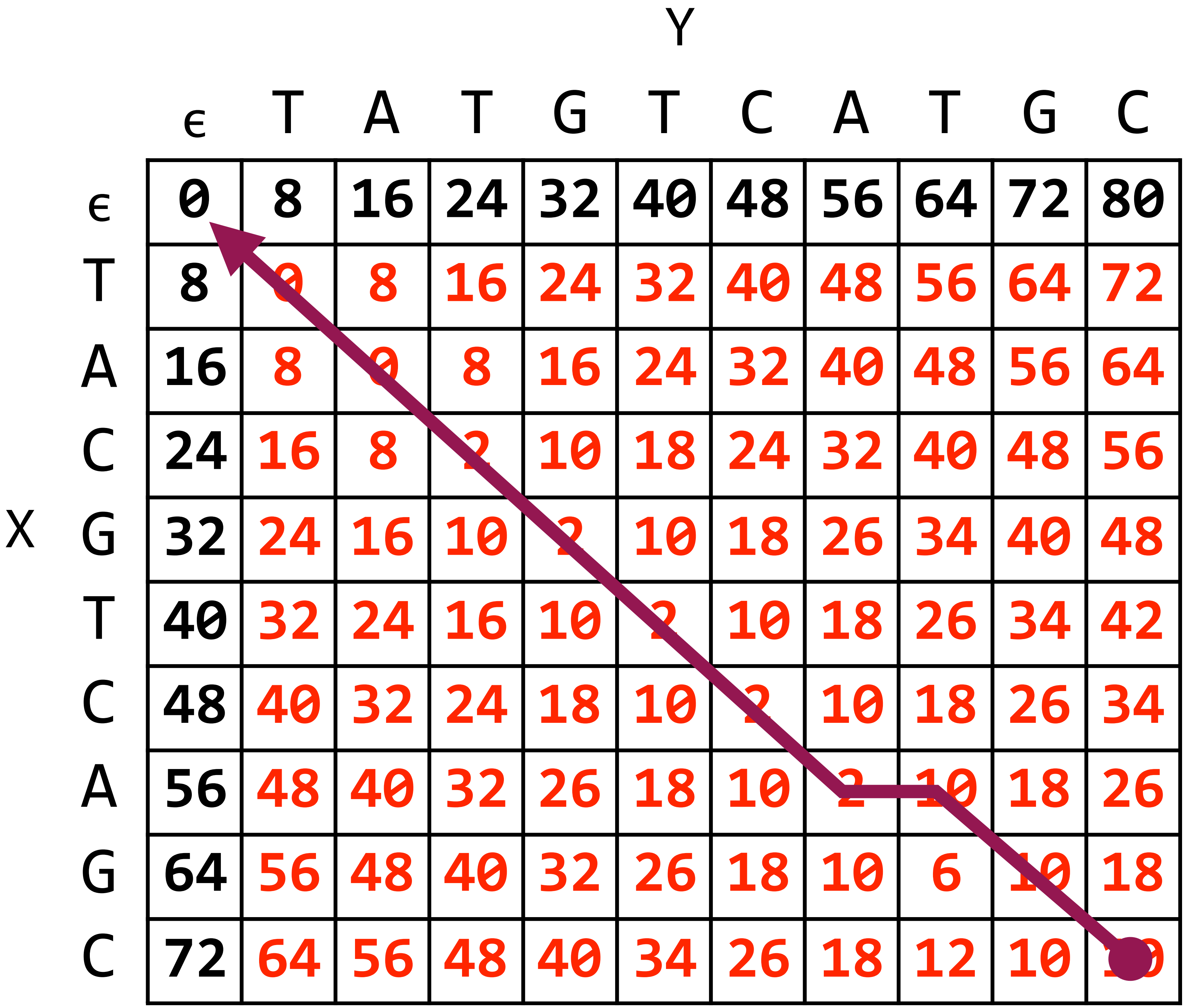
Y

		ε	T	A	T	G	T	C	A	T	G	C
	ε	0	-7	-14	-21	-28	-35	-42	-49	-56	-63	-70
	T	-7	1	-6	-13	-20	-27	-34	-41	-48	-55	-62
	A	-14	-6	2	-5	-12	-19	-26	-33	-40	-47	-54
	C	-21	-13	-5	1	-6	-13	-18	-25	-32	-39	-46
X	G	-28	-20	-12	-6	2	-5	-12	-19	-26	-31	-38
	T	-35	-27	-19	-11	-5	3	-4	-11	-18	-25	-32
	C	-42	-34	-26	-18	-12	-4	4	-3	-10	-17	-24
	A	-49	-41	-33	-25	-19	-11	-3	5	-2	-9	-16
	G	-56	-48	-40	-32	-24	-18	-10	-2	2	-1	-8
	C	-63	-55	-47	-39	-31	-25	-17	-9	-3	-1	0

$s(a, b)$

	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

Global alignment revisited



							Y				
		ϵ	T	A	T	G	T	C	A	T	G	C
	ϵ	0	8	16	24	32	40	48	56	64	72	80
	T	8	0	8	16	24	32	40	48	56	64	72
	A	16	8	0	8	16	24	32	40	48	56	64
	C	24	16	8	2	10	18	24	32	40	48	56
X	G	32	24	16	10	2	10	18	26	34	40	48
	T	40	32	24	16	10	2	10	18	26	34	42
	C	48	40	32	24	18	10	2	10	18	26	34
	A	56	48	40	32	26	18	10	2	10	18	26
	G	64	56	48	40	32	26	18	10	6	10	18
	C	72	64	56	48	40	34	26	18	12	10	10

Same traceback

$s(a, b)$

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Global alignment revisited

Global alignment value for ϵ, ϵ = 0

	ϵ	T	A	T	G	T	C	A	T	G	C
ϵ	0	-7	-14	-21	-28	-35	-42	-49	-56	-63	-70
T	-7	1	-6	-13	-20	-27	-34	-41	-48	-55	-62
A	-14	-6	2	-5	-12	-19	-26	-33	-40	-47	-54
C	-21	-13	-5	1	-6	-13	-18	-25	-32	-39	-46
G	-28	-20	-12	-6	2	-5	-12	-19	-26	-31	-38
T	-35	-27	-19	-11	-5	3	-4	-11	-18	-25	-32
C	-42	-34	-26	-18	-12	-4	4	-3	-10	-17	-24
A	-49	-41	-33	-25	-19	-11	-3	5	-2	-9	-16
G	-56	-48	-40	-32	-24	-18	-10	-2	2	-1	-8
C	-63	-55	-47	-39	-31	-25	-17	-9	-3	-1	0

$s(a, b)$

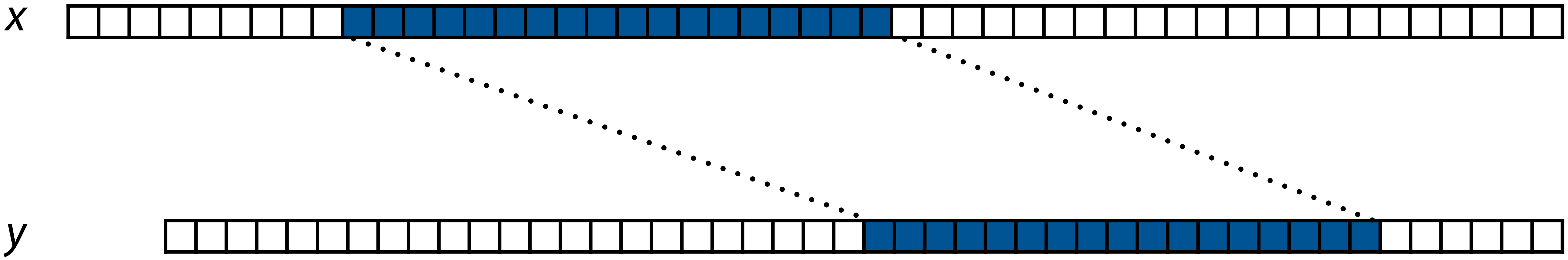
	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

Similarities (matches) get score >0

Dissimilarities (mismatches and gaps) get score <0

Local alignment

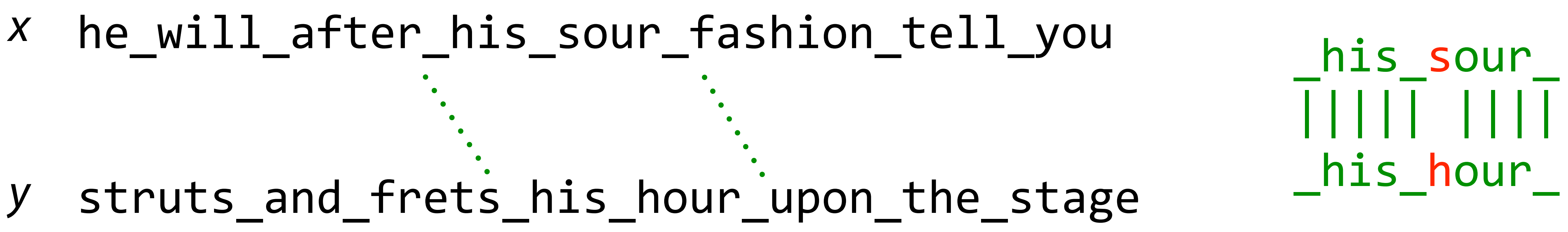
Given strings *x* and *y*, what is the optimal global alignment value of a *substring* of *x* to a *substring* of *y*. This is *local alignment*.



Assume scoring function where: (a) similarities get scores > 0, (b) dissimilarities get scores < 0, (c) global alignment value for $x = \epsilon$, $y = \epsilon$ is 0

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.*



```
x  he_will_after_his_sour_fashion_tell_you

y  struts_and_frets_his_hour_upon_the_stage
```

```
_his_sour_
||||| ||||
_his_hour_
```

Assume scoring function where: (a) similarities get scores > 0, (b) dissimilarities get scores < 0, (c) global alignment value for $x = \epsilon$, $y = \epsilon$ is 0

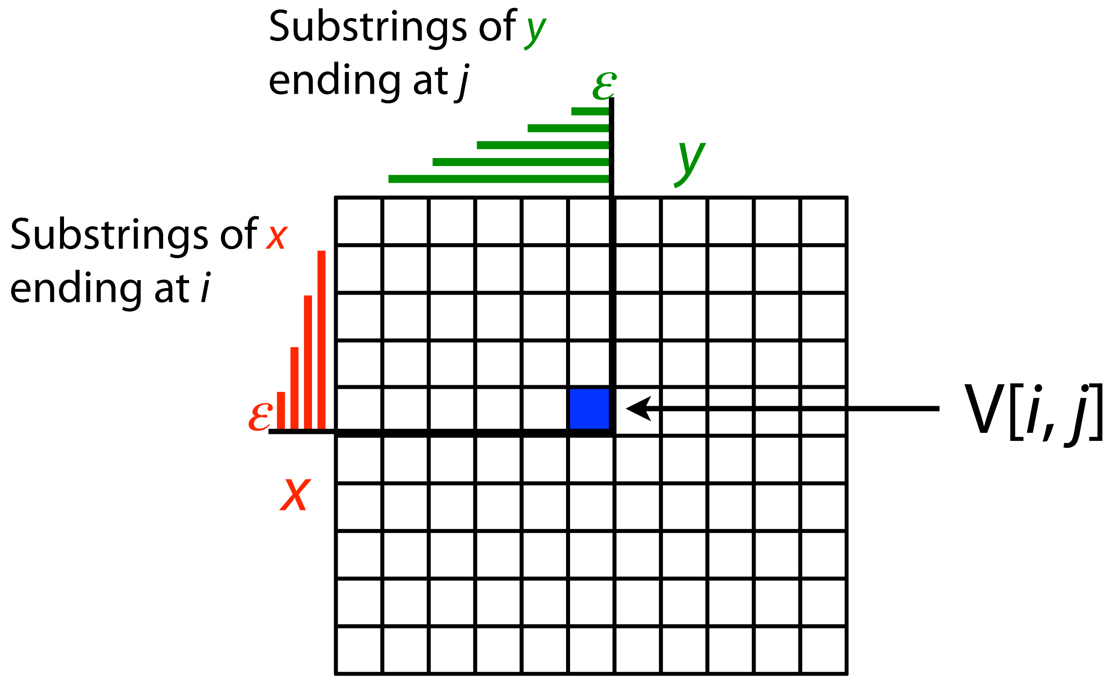
In some way, we're considering all possible *pairs* of substrings

What roughly is # substring pairs, where $|x| = n$, $|y| = m$? $O(m^2n^2)$

Surprisingly, we'll do it in $O(mn)$

Local alignment

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



Local alignment

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

Small example:

	ϵ	T	C	A	G
ϵ					
C					
A					
C					

What value goes here?

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

Small example:

	ϵ	T	C	A	G
ϵ					
C					
A					
C					1

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

Small example:

	ϵ	T	C	A	G
ϵ					
C					
A					
C					**1**

What value goes here?

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

Small example:

	ϵ	T	C	A	G
ϵ					
C					
A				2	
C					1

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

Small example:

What value goes here?

	ϵ	T	C	A	G
ϵ					
C					
A				**2**	
C					**1**

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

Small example:

	ε	T	C	A	G ε
ε					0
C					
A				2	
C					1

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

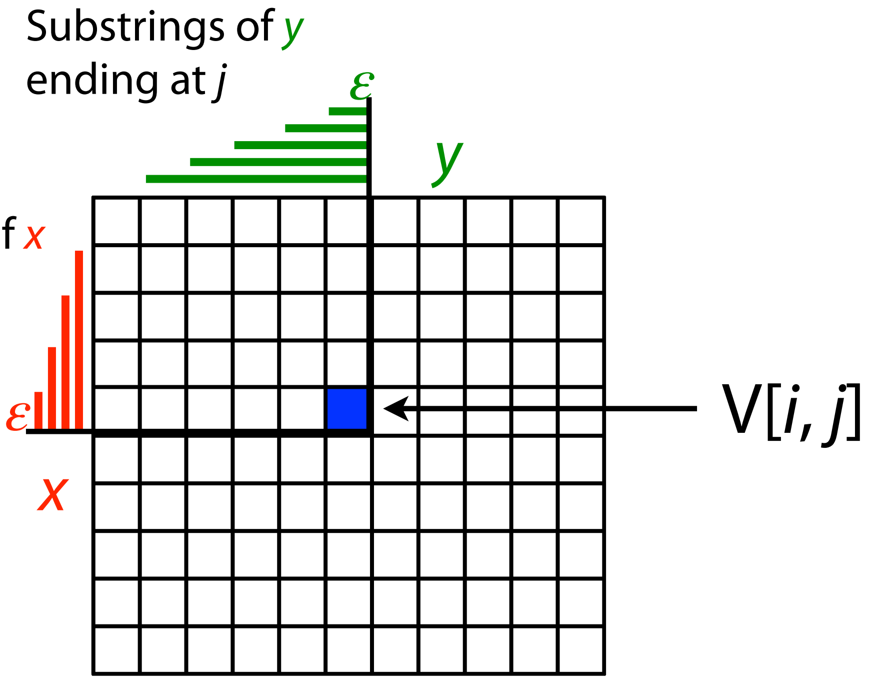
Small example:

	ϵ	T	C	A	G
ϵ	0	0	0	0	0
C	0	0	1	0	0
A	0	0	0	2	1
C	0	0	1	1	1

	A	C	G	T	-
A	1	-1	-1	-1	-1
C	-1	1	-1	-1	-1
G	-1	-1	1	-1	-1
T	-1	-1	-1	1	-1
-	-1	-1	-1	-1	

Local alignment

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

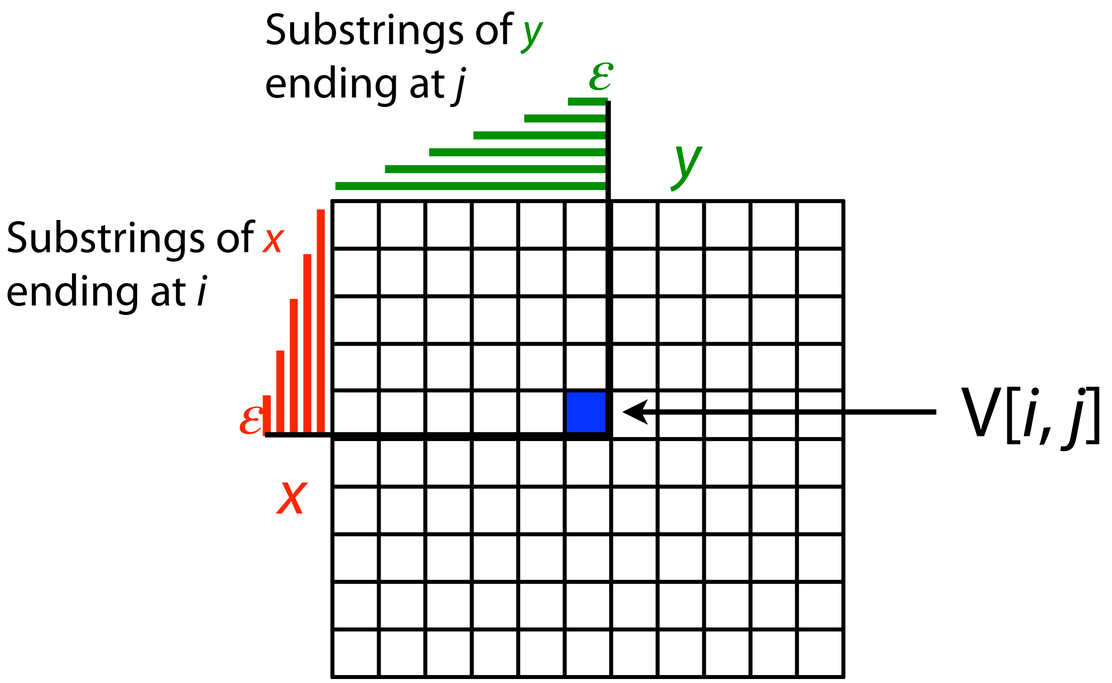


How to find best *local alignment*, i.e. the pair of substrings of *X* and *Y* with highest global alignment value?

max(V[*i*, *j*]) over all *i*, *j*

Local alignment

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



As for edit distance, there are only so many possibilities:

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

Vertical: append **D** to transcript for V[*i*-1, *j*], add penalty

Horizontal: append **I** to transcript for V[*i*, *j*-1], add penalty

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

See also: Gusfield 11.7.1 - 11.7.2

Local alignment

$$\boxed{\text{Let } V[0,j]=0, \text{ and let } V[i,0]=0}$$

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

$\boxed{s(a,b)}$ assigns a score to a particular match, gap, or replacement

(*gap* = insertion or deletion)

What's different from global alignment?

First row, column initialized to 0s

0 is one of the arguments of the max (because of ε, ε)

Scoring function with differences < 0, matches > 0

Dynamic-programming implementation of this is called *Smith-Waterman*

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

X

	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														
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	

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

$s(a,b)$

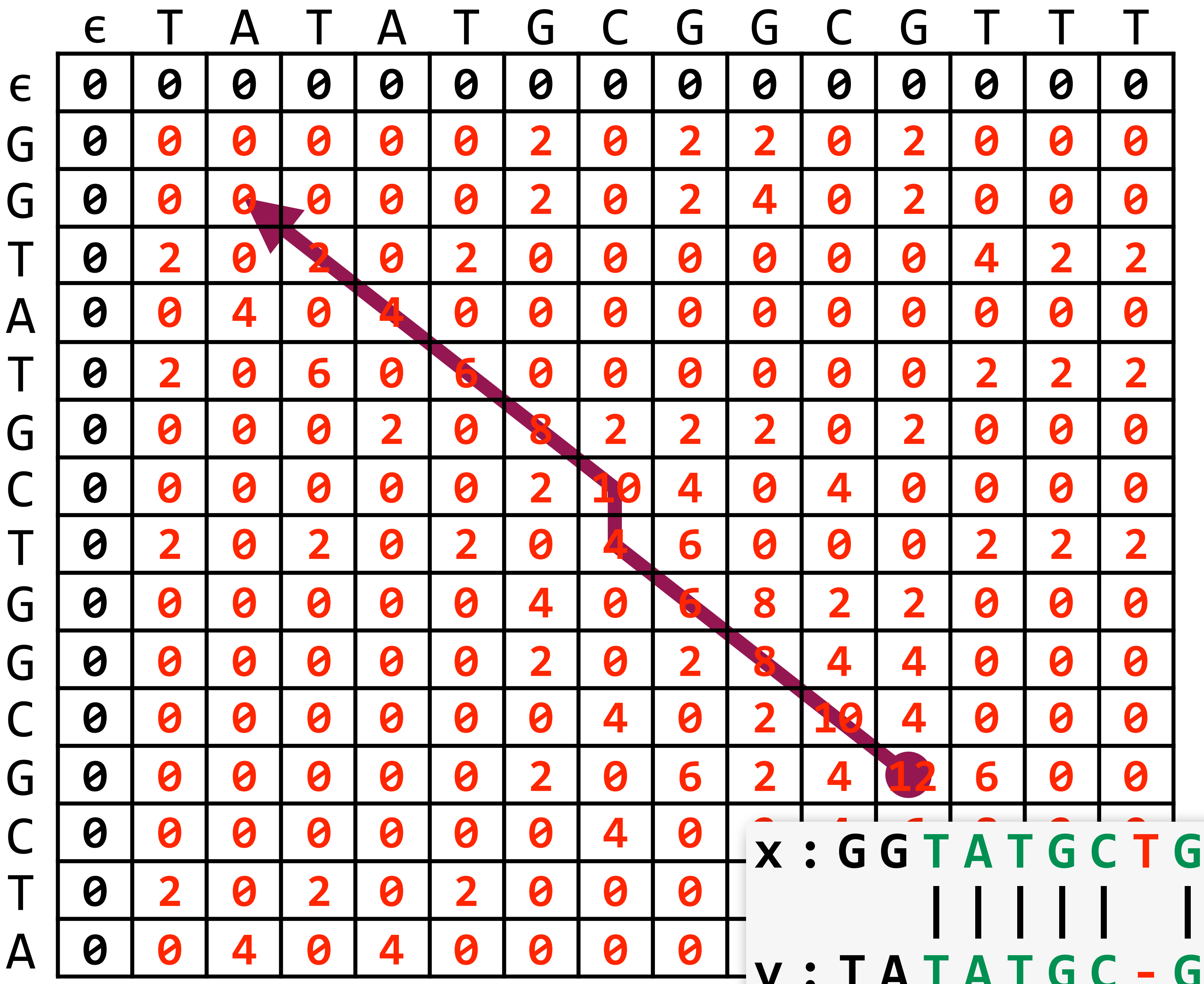
	A	C	G	T	-
A	2	-4	-4	-4	-6
C	-4	2	-4	-4	-6
G	-4	-4	2	-4	-6
T	-4	-4	-4	2	-6
-	-6	-6	-6	-6	

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

Where / how to backtrace?

Local alignment: Smith-Waterman

Backtrace: (a) start from *maximal* cell, (b) stop upon reaching 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	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	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
T	0	2	0	2	0	2	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
A	0	0	4	0	4	0	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

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

$s(a, b)$

	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

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

```python
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 range(1, len(x)+1):
        for j in range(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])
```

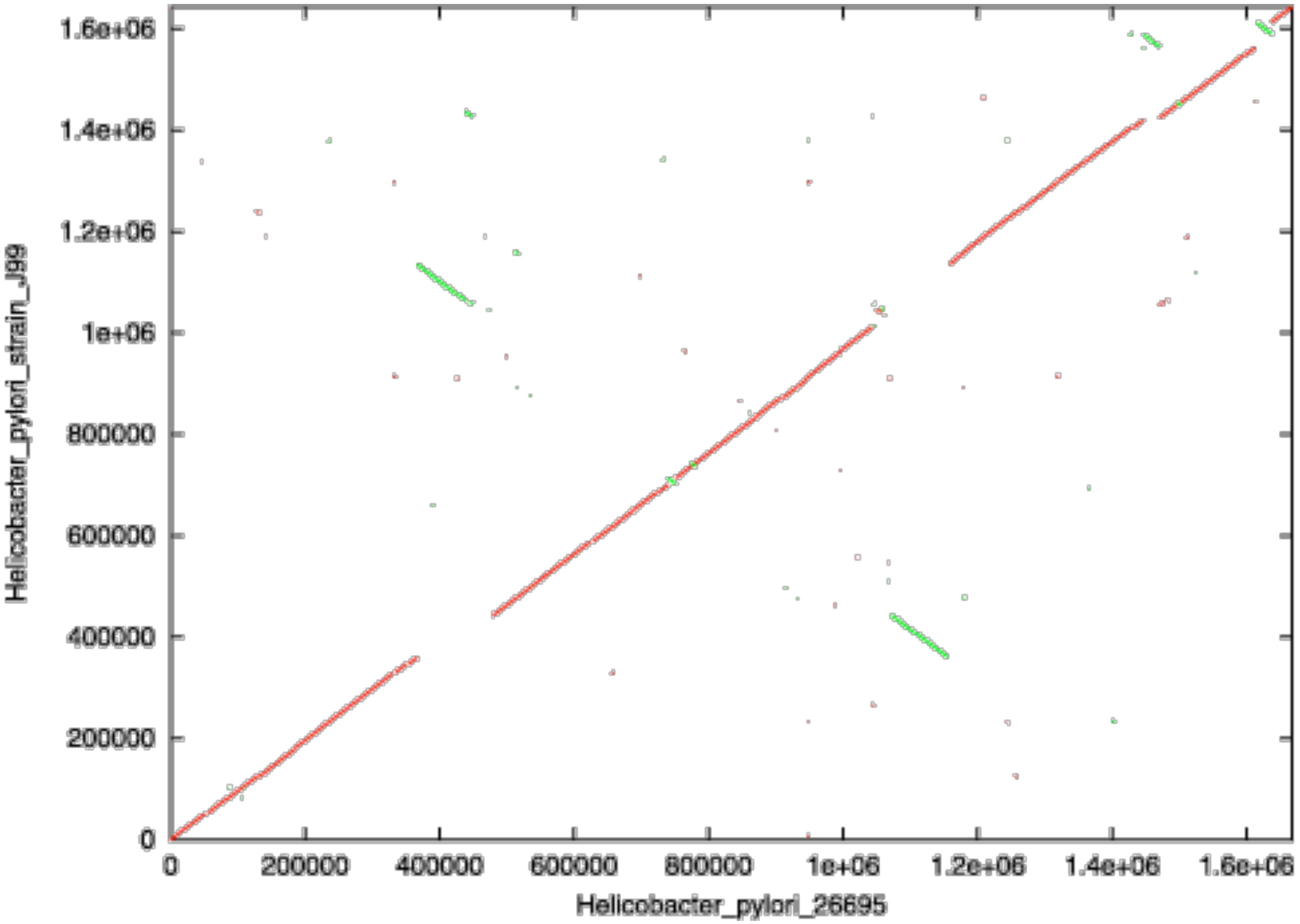
Let $V[0, j] = 0$, and let $V[i, 0] = 0$

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

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

Python example: http://bit.ly/CG_DP_Local

Local alignment in whole-genome alignment



MUMmer used a suffix tree to make this plot. Could we make it with dynamic programming alignment?

Global or local?

Might do *local* first, then string local alignments together ("chaining"). Sometimes called *glocal* alignment.

Axes show two strains of Helicobacter pylori, bacterium found in the stomach & associated with gastric ulcers