Exercises

1) Construct BWT(T), with T = 'CATGCAT'

T=CATGCAT$

```
CATGCAT$
ATGCAT$C
TGCAT$CA
GCAT$CAT
CAT$CATG
AT$CATGC
T$CATGCA
$CATGCAT
```

Rotation

```
$CATGCAT
AT$CATGC
ATGCAT$C
CAT$CATG
CATGCAT$
GCAT$CAT
T$CATGCA
TGCAT$CA
```

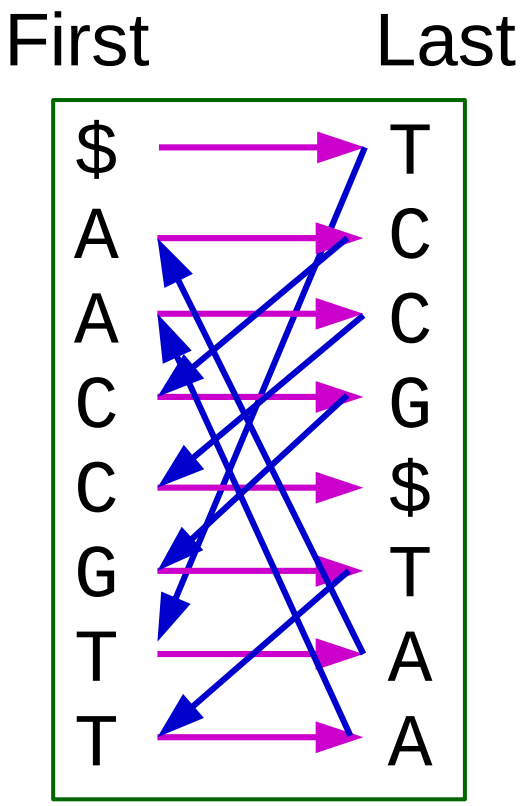
Sort
(BW matrix)

BWT(T)=
TCCG$TAA

Last
column

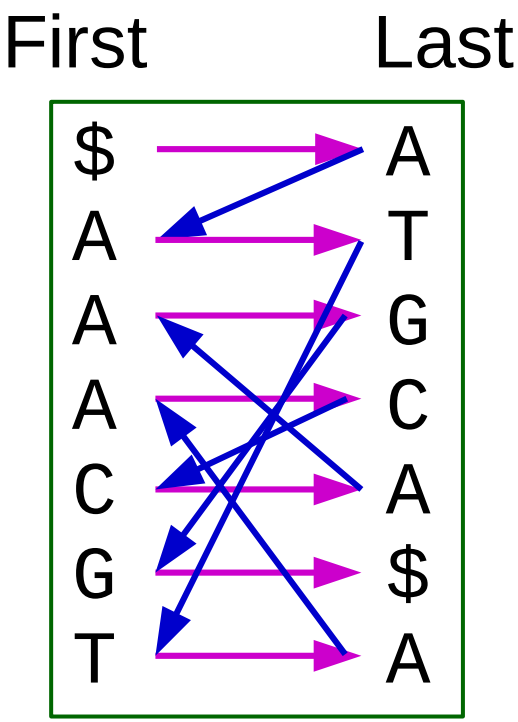
Reverse BWT

2) Draw out LF mapping to show how to reverse BWT(T) using arrows and LF columns as shown on the right:




```
BWT(T) = TCCG$TAA
T = CATGCAT$
```

Reverse BWT



BWT(T) = ATGCA$A
T = GACATA$

Exercises

3) Find positions of F for 'baa', 'aab' and 'ab' using FM index query

F	L	Position of F
$	b	x[0]
a	b	x[1]
a	b	x[2]
a	b	x[3]
a	$	x[4]
a	a	x[5]
a	a	x[6]
a	a	x[7]
b	b	x[8]
b	a	x[9]
b	b	x[10]
b	a	x[11]
b	a	x[12]
b	a	x[13]

LF mapping of entire T takes too long
Reverse search 'ab'
where is F(b)? 8-13
which are preceded by a? 9-13
Use LF mapping to find F(a): 4-7

$abaabaabbaabb
aabaabbaabb$ab
aabb$abaabaabb
aabbaabb$abaab
abaabaabbaabb$
abaabbaabb$aba
abb$abaabaabba
abbaabb$abaaba
b$abaabaabbaab
baabaabbaabb$a
baabb$abaabaab
baabbaabb$abaa
bb$abaabaabbaa
bbaabb$abaabaa

ab	baa	aab
x[4]	x[9]	x[1]
x[5]	x[10]	x[2]
x[6]	x[11]	x[3]
x[7]		

Exercises

3) Find positions of F for 'baa', 'aab' and 'ab' using FM index query

F	L	Position of F
$	b	x[0]
a	b	x[1]
a	b	x[2]
a	b	x[3]
a	$	x[4]
a	a	x[5]
a	a	x[6]
a	a	x[7]
b	b	x[8]
b	a	x[9]
b	b	x[10]
b	a	x[11]
b	a	x[12]
b	a	x[13]

LF mapping of entire T takes too long
Reverse search 'ab'
where is F(b)? 8-13
which are preceded by a? 9-13
Use LF mapping to find F(a): 4-7

$abaabaabbaabb
aabaabbaabb$ab
aabb$abaabaabb
aabbaabb$abaab
abaabaabbaabb$
abaabbaabb$aba
abb$abaabaabba
abbaabb$abaaba
b$abaabaabbaab
baabaabbaabb$a
baabb$abaabaab
baabbaabb$abaa
bb$abaabaabbaa
bbaabb$abaabaa

ab	baa	aab
x[4]	x[9]	x[1]
x[5]	x[10]	x[2]
x[6]	x[11]	x[3]
x[7]		

Exercises

3) Find positions of F for 'baa', 'aab' and 'ab' using FM index query

F	L	Position of F
$	b	x[0]
a	b	x[1]
a	b	x[2]
a	b	x[3]
a	$	x[4]
a	a	x[5]
a	a	x[6]
a	a	x[7]
b	b	x[8]
b	a	x[9]
b	b	x[10]
b	a	x[11]
b	a	x[12]
b	a	x[13]

LF mapping of entire T takes too long
Reverse search 'ab'
where is F(b)? 8-13
which are preceded by a? 9-13
Use LF mapping to find F(a): 4-7

$abaabaabbaabb
aabaabbaabb$ab
aabb$abaabaabb
aabbaabb$abaab
abaabaabbaabb$
abaabbaabb$aba
abb$abaabaabba
abbaabb$abaaba
b$abaabaabbaab
baabaabbaabb$a
baabb$abaabaab
baabbaabb$abaa
bb$abaabaabbaa
bbaabb$abaabaa

ab	baa	aab
x[4]	x[9]	x[1]
x[5]	x[10]	x[2]
x[6]	x[11]	x[3]
x[7]		

Exercises

3) Find positions of F for 'baa', 'aab' and 'ab' using FM index query

F	L	Position of F
$	b	x[0]
a	b	x[1]
a	b	x[2]
a	b	x[3]
a	$	x[4]
a	a	x[5]
a	a	x[6]
a	a	x[7]
b	b	x[8]
b	a	x[9]
b	b	x[10]
b	a	x[11]
b	a	x[12]
b	a	x[13]

LF mapping of entire T takes too long
Reverse search 'baa'
where is F(a)? 1-7
which are preceded by a? 5-7
Use LF mapping to find F(a): 1-3

$abaabaabbaabb
aabaabbaabb$ab
aabb$abaabaabb
aabbaabb$abaab
abaabaabbaabb$
abaabbaabb$aba
abb$abaabaabba
abbaabb$abaaba
b$abaabaabbaab
baabaabbaabb$a
baabb$abaabaab
baabbaabb$abaa
bb$abaabaabbaa
bbaabb$abaabaa

Which are preceded by 'b': 1-3
What are positions of 'b': 9-11

ab	baa	aab
x[4]	x[9]	x[1]
x[5]	x[10]	x[2]
x[6]	x[11]	x[3]
x[7]		

Exercises

3) Find positions of F for 'baa', 'aab' and 'ab' using FM index query

F	L	Position of F
$	b	x[0]
a	b	x[1]
a	b	x[2]
a	b	x[3]
a	$	x[4]
a	a	x[5]
a	a	x[6]
a	a	x[7]
b	b	x[8]
b	a	x[9]
b	b	x[10]
b	a	x[11]
b	a	x[12]
b	a	x[13]

LF mapping of entire T takes too long
Reverse search 'aab'

$abaabaabbaabb
aabaabbaabb$ab
aabb$abaabaabb
aabbaabb$abaab
abaabaabbaabb$
abaabbaabb$aba
abb$abaabaabba
abbaabb$abaaba
b$abaabaabbaab
baabaabbaabb$a
baabb$abaabaab
baabbaabb$abaa
bb$abaabaabbaa
bbaabb$abaabaa

ab	baa	aab
x[4]	x[9]	x[1]
x[5]	x[10]	x[2]
x[6]	x[11]	x[3]
x[7]		

Today's objectives

- Types of alignments and how they are scored
- Needleman-Wunsch algorithm
- Smith-Waterman algorithm

What is an alignment?

```
GATTGTATCTAACTA

GTTCTATTCTAAC
```

```
GATTGTAT-CTAACTA
|  ||.|||  |||||
G-TTCTATTCTAAC--
```

Substitution (G to C)

Insertion/deletion (T to -)

A gapped arrangement of coordinates between sequences.

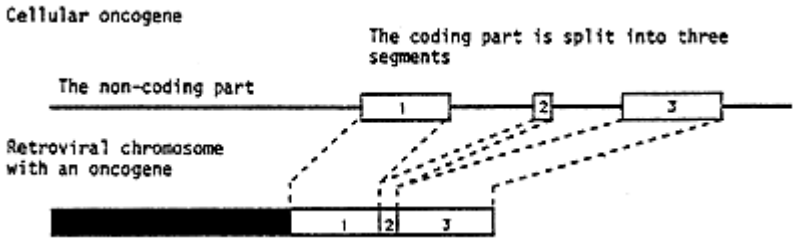
Essentially a question of where to put gaps, start and stop.

Minimize edit distance (mutation) between two sequences

Alignment Problems

1) Generate the best alignment between v-src and c-src

2) Find the best alignment of v-src to the human genome




```
-GACCCATTCGGTTCAATTATAGCCCAACCTGAGAACTTGAATAAGACTGA---TATCACTGACTACAAA-
 || ..||||||||||||||.||||||.||||||||.||||.||||  .||||||   ||||| .|| ||..||
TGA-ATATTCGGTTCAATTTTAGCCAAACCTGAAAACTCGAAT--CACTGATGTTATCA-AGA-TATTAAA
```

Problem:

- Word/substring only works if sequences are highly similar, need to handle mismatches
- Words/substrings don't handle gaps, need to handle gaps
- Method to extend seeds with mismatch & gaps

Alignments covered & history

Classes

Pairwise vs multiple alignment

Local vs global alignment

Exhaustive vs approximate

Global, exhaustive, pairwise alignment

– Needleman S.B. and Wunsch C.D. (1970) J. Mol. Biol. 48, 443-453

Local, exhaustive, pairwise alignment

– Smith T.F. and Waterman M.S. (1981) J. Mol. Biol.147, 195-197

Local, approximate, pairwise alignment

BLAST: Basic Local Alignment Search Tool (Altschul et al. 1990).

Global, approximate, multiple alignment

ClustalW: Thompson,J.D. et al. (1994) Nucleic Acids Res., 22, 4673–4680.

Local, approximate, pairwise alignment

Bowtie/BWA: time/memory solution to short read mapping (2009/2009)

Sequence Alignment

Homology is existence of shared ancestry between sequences. Homology can be inferred from sequence similarity.

For newly identified sequence, identification of related sequences helps ‘annotate’ that sequence with various attributes, like Function Origin, Structure

Alignment is needed to establish evolutionary relationships among organisms.

Alignment is needed to know where in a genome a sequence comes from.

Global versus local alignment

```
GACCCATTCGGTTCAATTATAGCCCAACCTGAGAACTTGAATAAGACTGATATCACTGACTACAAA

TGAATATTCGGTTCAATTTTAGCCAAACCTGAAAACTCGAATCACTGATGTTATCAAGATATTAAA
```

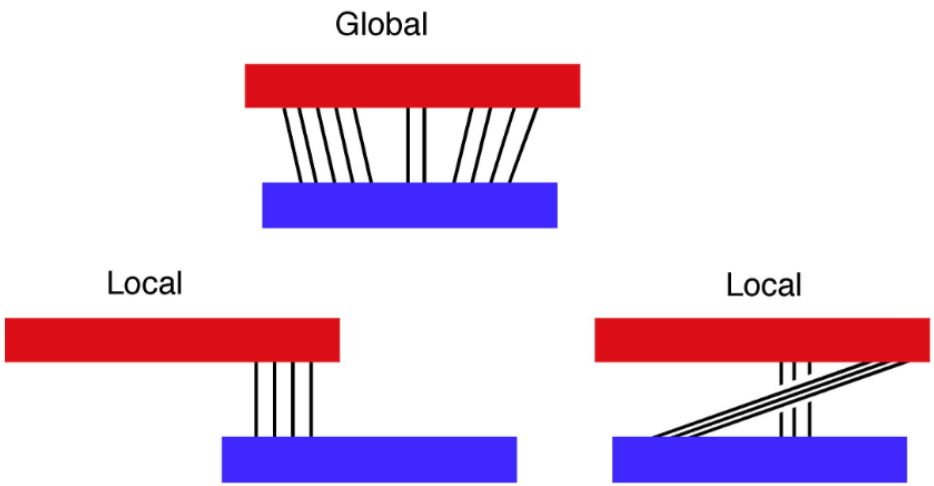
Global alignment: end to end arrangement of both sequences

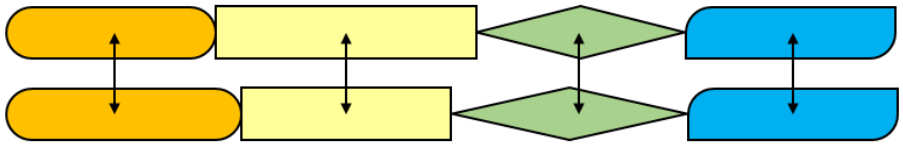
```
-GACCCATTCGGTTCAATTATAGCCCAACCTGAGAACTTGAATAAGACTGA---TATCACTGACTACAAA-
 || ..||||||||||||||.|||||.||||||||.||||.||||  .|||||   ||||| .|| ||..||
TGA-ATATTCGGTTCAATTTTAGCCAAACCTGAAAACTCGAAT--CACTGATGTTATCA-AGA-TATTAAA
```

```
ATTCGGTTCAATTATAGCCCAACCTGAGAACTTGAATAAGACTGA---TATCA
||||||||||||||.|||||.||||||||.||||.||||  .|||||   |||||
ATTCGGTTCAATTTTAGCCAAACCTGAAAACTCGAAT--CACTGATGTTATCA
```

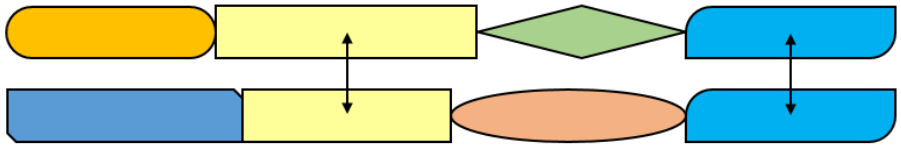
Local alignment: arrangement of subsequences

Global versus local alignment





Global Alignment



Local Alignment

Global

- end to end
- sequence can only be represented once
- forced homology

Local

- subsequences
- multiple alignments for one query
- handles rearrangements

Exercises

1) Describe these alignments as global/local and exhaustive/approximate: BLAST, Needleman-Wunsch, Smith-Waterman, Bowtie

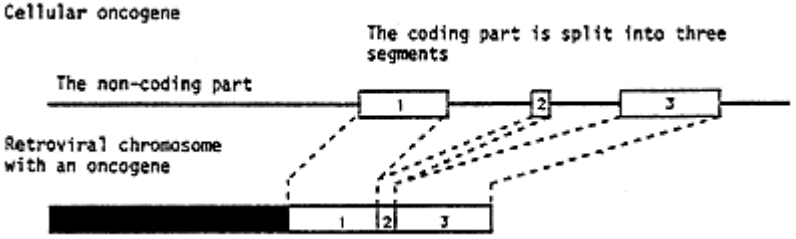
2) Which alignment method would you use: BLAST, Needleman-Wunsch, Smith-Waterman?

a) Generate an alignment between a cDNA and the human genome (e.g. v-src to human)

b) Find homolog of human cDNA in chicken (e.g. human and chicken src)

c) Find best alignment of human and chicken homologs (e.g. human and chicken src)

d) Align v-src to c-src



Who wore it better?

What is the better fit

```
-GACCCATACGGTTCAA
 ||  ..||.|||||||||
TGA-ATATTCGGTTCAA
```

vs

```
-GACCCATA--CGGTTCAA
 ||    |||   |||||||||
TGA---ATATTCGGTTCAA
```

Problem:

- how do we compare (score)
- there are too many to choose from

Scoring an alignment

```
-GACCCATACGGTTCAA
 || ..||.|||||||
TGA-ATATTCGGTTCAA
```

vs

```
-GACCCATA--CGGTTCAA
 ||   |||  ||||||||
TGA---ATATTCGGTTCAA
```

Scoring Alignment
Gap = -2
Mismatch = -1
Match = 1

Score = Match+Mismatch+GAP

Which has the higher score?

```
-GACCCATACGGTTCAA
 || ..||.|||||||
TGA-ATATTCGGTTCAA
```

vs

```
-GACCCATA--CGGTTCAA
 ||   |||  |||||||
TGA---ATATTCGGTTCAA
```

Alignment #1
12-3-4 = 5
higher score

Scoring Alignment
Gap = -2
Mismatch = -1
Match = 1

Alignment #2
13-0-12 = 1

Score = Match+Mismatch+GAP

Score criteria matters

```
-GACCCATACGGTTCAA
 || ..||.|||||||
TGA-ATATTCGGTTCAA
```

vs

```
-GACCCATA--CGGTTCAA
 ||   |||  ||||||||
TGA---ATATTCGGTTCAA
```

Alignment #1
12-3-4 = 5
24-3-2 = 19

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 2

Alignment #2
13-0-12 = 1
26-0-6 = 20

Score = Match+Mismatch+GAP

Scoring is easy: How do we find best scoring alignments

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

Matrix of sequence comparisons

- represents all matches/mismatches
- directional (reverse)
- doesn't include complement

Dot plot representation

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

Matrix of sequence comparisons

- represents all matches/mismatches
- directional (reverse)
- doesn't include complement

Forward
Reverse

Dot plot representation of alignment

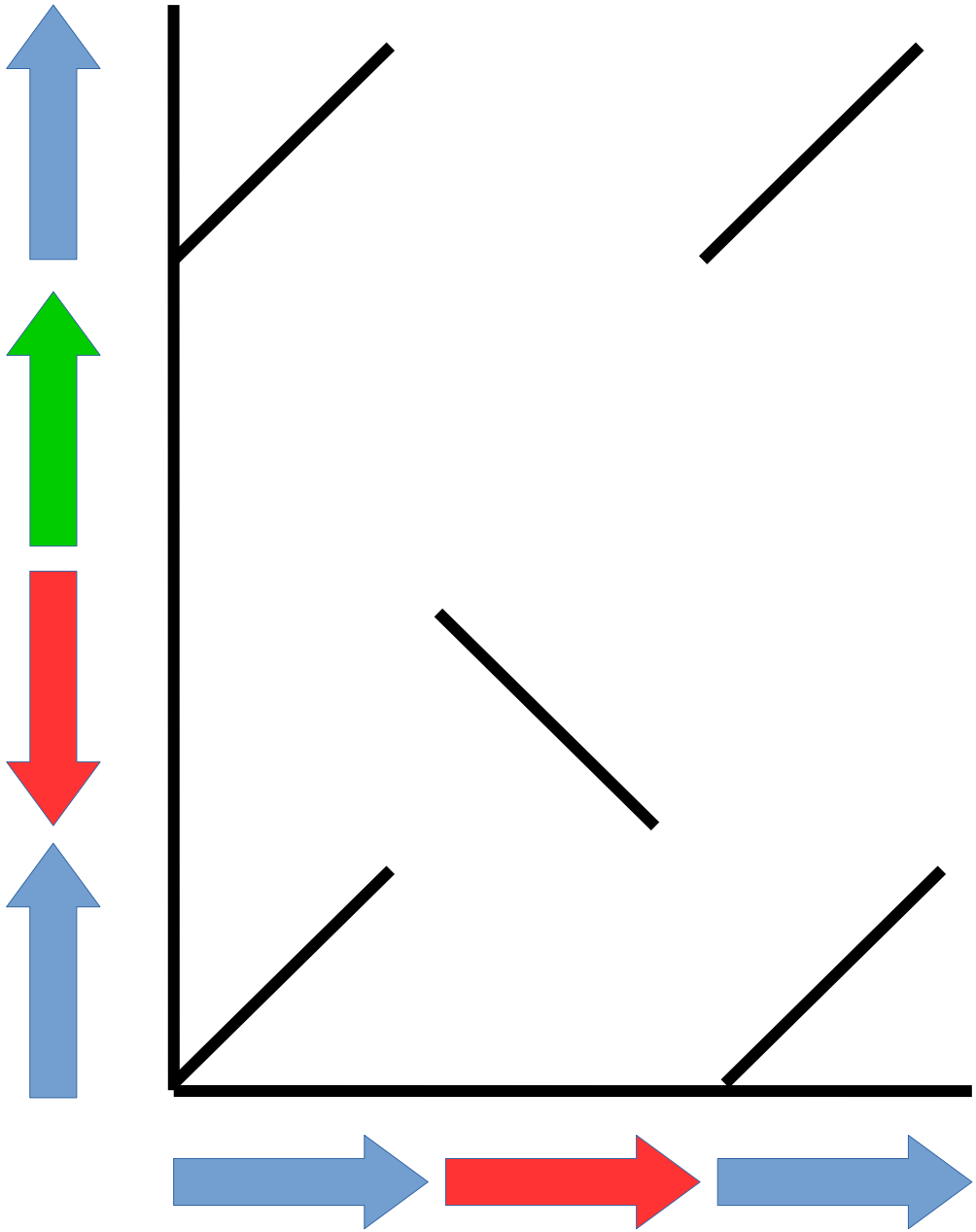
	G	A	C	C	C	A	T	A	C	G	G	T	T	C	A	A
T							1					1	1			
G	1									1	1					
A		1				1		1							1	1
A		1				1		1							1	1
T							1					1	1			
A		1				1		1							1	1
T							1					1	1			
T							1					1	1			
C			1	1	1				1					1		
G	1									1	1					
G	1									1	1					
T							1					1	1			
T							1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

Diagonals represent all ungapped strings of matches

```
-GACCCATACGGTTCAA
 || ..||.||||||||
TGA-ATATTCGGTTCAA
```

```
-GACCCATA--CGGTTCAA
 ||   |||  ||||||||
TGA---ATATTCGGTTCAA
```

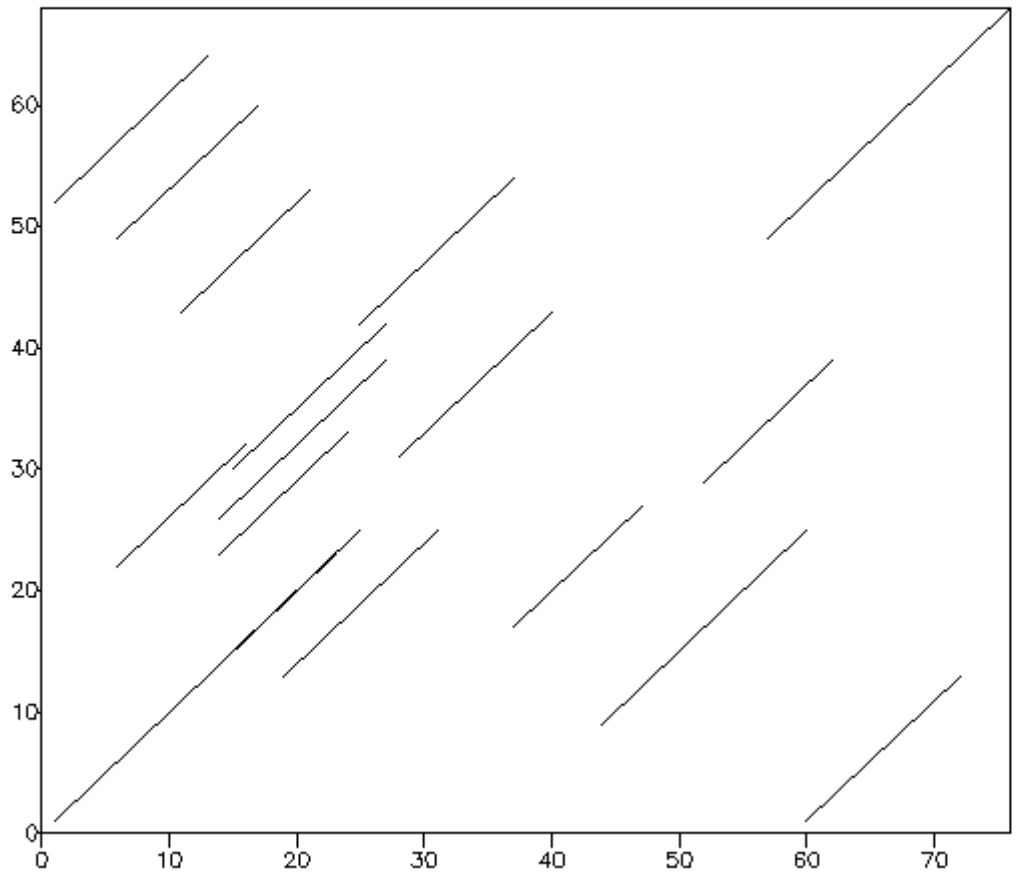
Gaps, repeats and inverted repeats



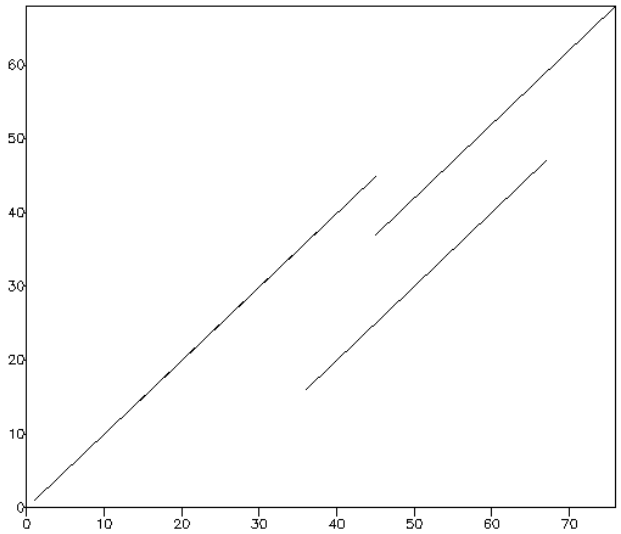
A dotplot represents all possible ungapped alignments

```
************ * *** ***  *     ****   *  ****    * * *          ********* * **
CTTCCAAGACTCTTGGGTATTGAAACTGCTGGTGGTGTCATGACCAAGTTGAT---------TCCAAGAAACTCTAC
CTTCCAAGACTCCTAGGTGTTGGTATGCAAGGTGACATTTTGACATTTTCGGTATTGTTGTCCCAAGAAACACAAC
```

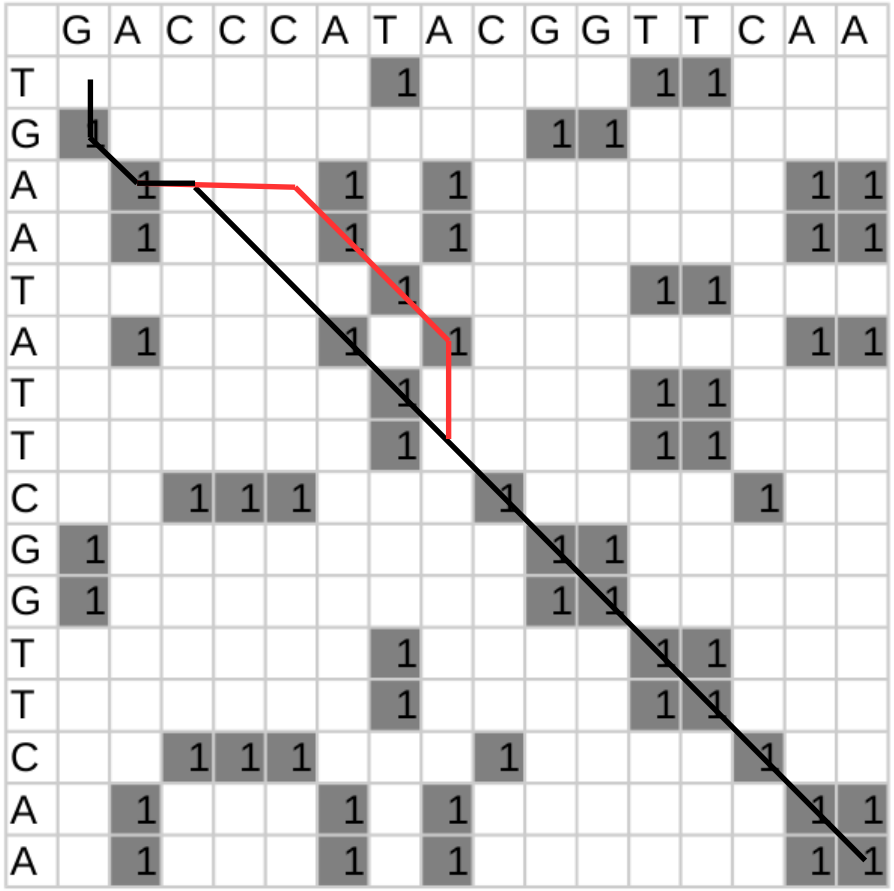
repeat



- Lines show windows (size) above some threshold (score % mismatch)
- Below shows increased window with lower score



Alignment: what path represents the best scoring alignment

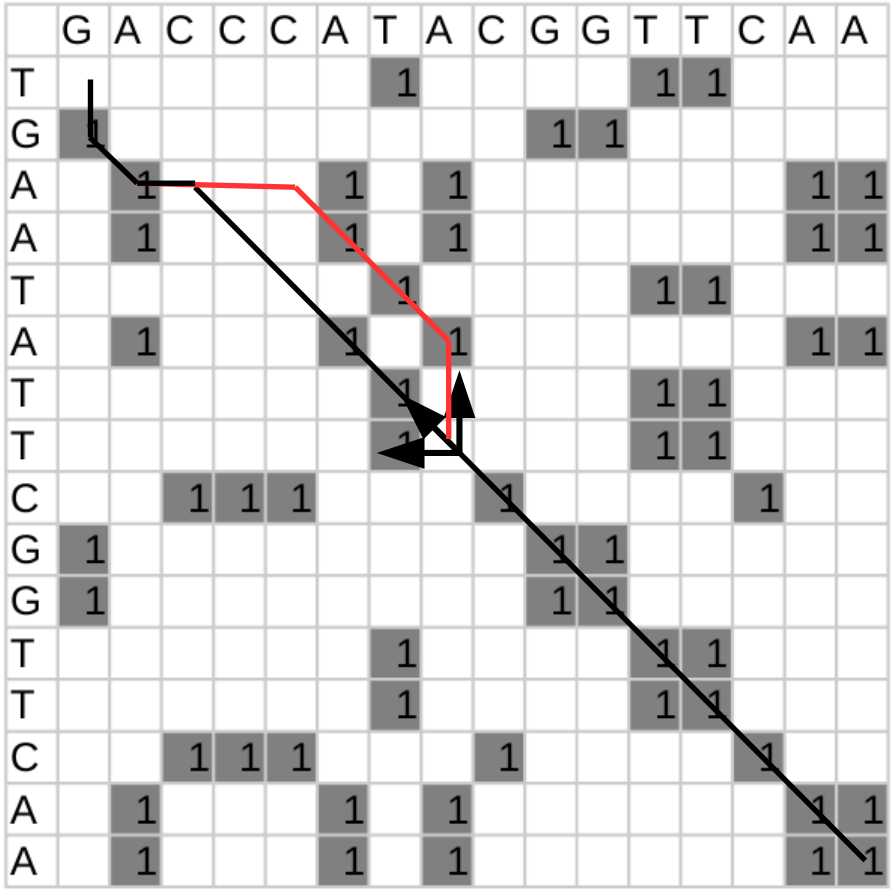


Gaps are represented by vertical and horizontal lines.

```
-GACCCATACGGTTCAA
 || ..||.|||||||
TGA-ATATTCGGTTCAA
```

```
-GACCCATA--CGGTTCAA
 ||   |||  |||||||
TGA---ATATTCGGTTCAA
```

Alignment: what path represents the best scoring alignment



All possible alignments represented by three possible moves (diagonal, left, up)

Diagonal: pair two bases

Left: insert gap in left sequence

Up: insert gap in top sequence

```
-GACCCATACGGTTCAA
 || ..||.|||||||
TGA-ATATTCGGTTCAA
```

```
-GACCCATA--CGGTTCAA
 ||   |||  |||||||
TGA---ATATTCGGTTCAA
```

Problem: Number of possible alignments: (m+n)!/m!*n! (n=m=16: 601 million)

Dynamic Programming

Global Alignments:

– Needleman S.B. and Wunsch C.D. (1970) J. Mol. Biol. 48, 443-453

Local Alignments:

– Smith T.F. and Waterman M.S. (1981) J. Mol. Biol.147, 195-197

– One simple modification of Needleman/Wunsch: when a value in the score matrix becomes negative, reset it to zero (begin a new alignment)

Both are guaranteed to be mathematically optimal:

- Given two sequences (and a scoring system) these algorithms are guaranteed to find the very best scoring alignment between the two sequences!
- Slow O(nm) algorithm
- Performed in 2 stages
    - Prepare a scoring matrix using recursive function
    - Scan matrix diagonally using traceback protocol

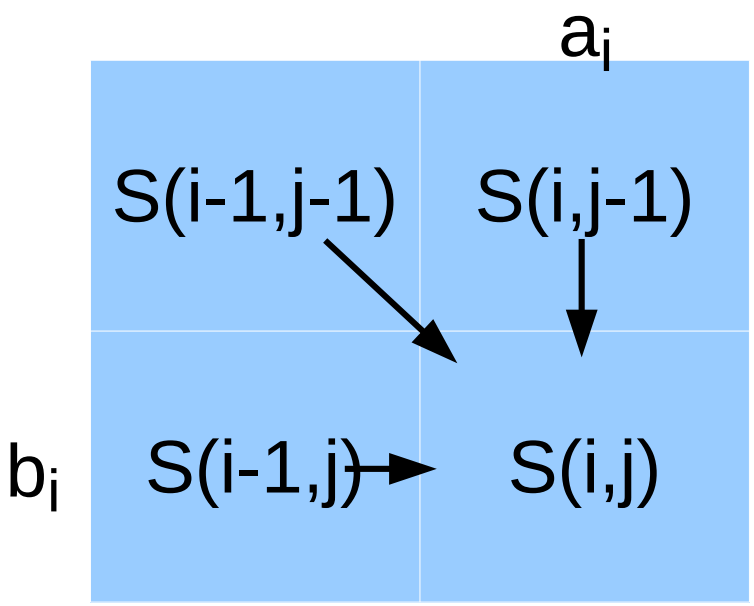
What is a dynamic programming algorithm

- Algorithms that break a problem into smaller sub-problems and use the solutions of those sub-problems to construct the solution of a larger one.

- Number of sub-problems may become very large, so DP organizes computations to avoid computing values that you already know

Problem: Number of possible alignments: (m+n)!/m!*n! (n=m=16: 601 million)

Scoring alignments with a subproblem



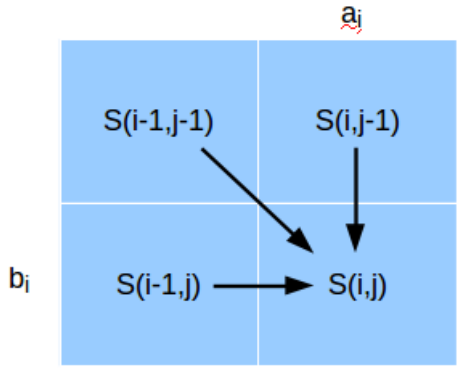
$$S(i,j) = \max \begin{cases} S(i-1,j-1)+score(a_i,b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

Scoring matrix example

- Fill in gapped edges
- Fill in scores left to right, top to bottom

	-	A	A	G
-	0	-1	-2	-3
A	-1			
G	-2			
C	-3			

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 1



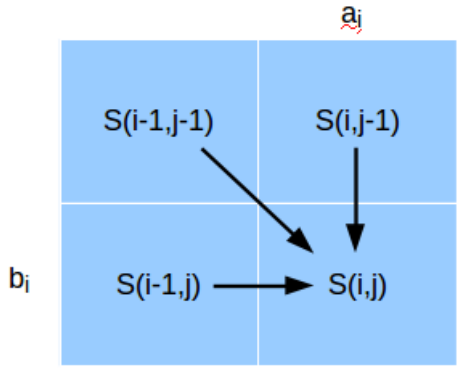
$$S(i,j) = \max \begin{cases} S(i-1,j-1) + score(a_j, b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

Scoring matrix example

- Fill in gapped edges
- Fill in scores left to right, top to bottom

	-	A	A	G
-	0	-1	-2	-3
A	-1	{-2,1,-2}		
G	-2			
C	-3			

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 1



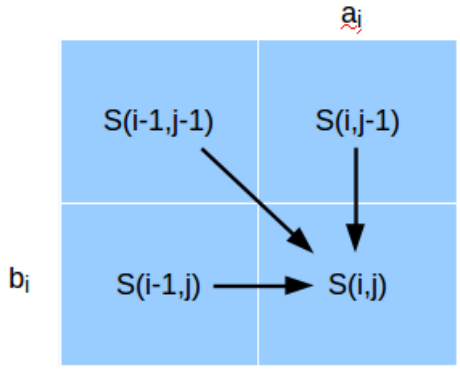
$$S(i,j) = \max \begin{cases} S(i-1,j-1) + score(a_j, b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

Scoring matrix example

- Fill in gapped edges
- Fill in scores left to right, top to bottom

	-	A	A	G
-	0	-1	-2	-3
A	-1	1	?	
G	-2			
C	-3			

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 1



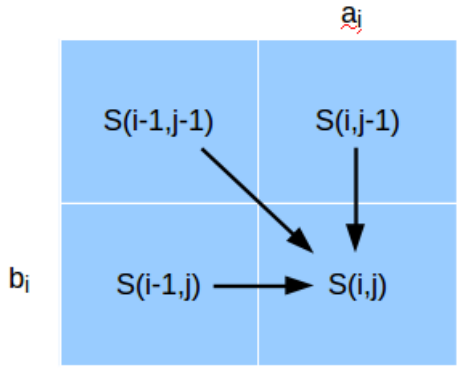
$$S(i,j) = \max \begin{cases} S(i-1,j-1) + score(a_j, b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

Scoring matrix example

- Fill in gapped edges
- Fill in scores left to right, top to bottom

	-	A	A	G
-	0	-1	-2	-3
A	-1	1	0	
G	-2			
C	-3			

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 1



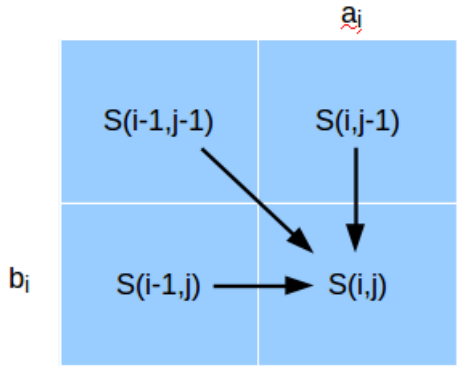
$$S(i,j) = \max \begin{cases} S(i-1,j-1) + score(a_j, b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

Scoring matrix example

- Fill in gapped edges
- Fill in scores left to right, top to bottom
- Traceback start at lower right, follow the arrows

	-	A	A	G
-	0	-1	-2	-3
A	-1	1	0	-1
G	-2	0	0	1
C	-3	-1	-1	0

Scoring Alignment
Gap = -1
Mismatch = -1
Match = 1



$$S(i,j) = \max \begin{cases} S(i-1,j-1) + score(a_j, b_i) \\ S(i,j-1) + gap \\ S(i-1,j) + gap \end{cases}$$

The lower right entry has the maximum scoring global alignment. This alignment can be found using a traceback.

Traceback: start from lower right and move back through pointers until the start. Guarantee is for best scoring alignment, there may be more than one.

Diagonal move: take 1 bp from each
Up: insert gap in top and 1 bp from left
Left: insert gap in left and 1 bp from top

```
AAG-
A-GC

AAG-
-AGC
```

	-	A	A	G
-	0	-1	-2	-3
A	-1	1	0	-1
G	-2	0	0	1
C	-3	-1	-1	0

Two equal 'best' alignments. Algorithm only reports one picked arbitrarily

Scoring matrix example

Needleman–Wunsch

match = 1 mismatch = –1 gap = –2

		G	A	T	T	A	G	T	A	C
	0	–2	–4	–6	–8	–10	–12	–14	–16	–18
G	–2									
A	–4									
C	–6									
C	–8									
T	–10									
A	–12									
T	–14									
A	–16									

Scoring matrix example

Needleman–Wunsch

match = 1 mismatch = –1 gap = –2

		G	A	T	T	A	G	T	A	C
	0	–2	–4	–6	–8	–10	–12	–14	–16	–18
G	–2	1	–1	–3	–5	–7	–9	–11	–13	–15
A	–4									
C	–6									
C	–8									
T	–10									
A	–12									
T	–14									
A	–16									

Scoring matrix example

Needleman–Wunsch

match = 1 mismatch = –1 gap = –2

		G	A	T	T	A	G	T	A	C
	0	–2	–4	–6	–8	–10	–12	–14	–16	–18
G	–2	1	–1	–3	–5	–7	–9	–11	–13	–15
A	–4	–1	2	0	–2	–4	–6	–8	–10	–12
C	–6	–3	0	1	–1	–3	–5	–7	–9	–9
C	–8	–5	–2	–1	0	–2	–4	–6	–8	–8
T	–10	–7	–4	–1	0	–1	–3	–3	–5	–7
A	–12	–9	–6	–3	–2	1	–1	–3	–2	–4
T	–14	–11	–8	–5	–2	–1	0	0	–2	–3
A	–16	–13	–10	–7	–4	–1	–2	–1	1	–1

What is the other alignment?

```
GA-TTAGTAC
||  .||  ||
GACCTA-TA-
```

Scoring matrix example

Needleman–Wunsch

match = 1 mismatch = –1 gap = –2

		G	A	T	T	A	G	T	A	C
	0	–2	–4	–6	–8	–10	–12	–14	–16	–18
G	–2	1	–1	–3	–5	–7	–9	–11	–13	–15
A	–4	–1	2	0	–2	–4	–6	–8	–10	–12
C	–6	–3	0	1	–1	–3	–5	–7	–9	–9
C	–8	–5	–2	–1	0	–2	–4	–6	–8	–8
T	–10	–7	–4	–1	0	–1	–3	–3	–5	–7
A	–12	–9	–6	–3	–2	1	–1	–3	–2	–4
T	–14	–11	–8	–5	–2	–1	0	0	–2	–3
A	–16	–13	–10	–7	–4	–1	–2	–1	1	–1

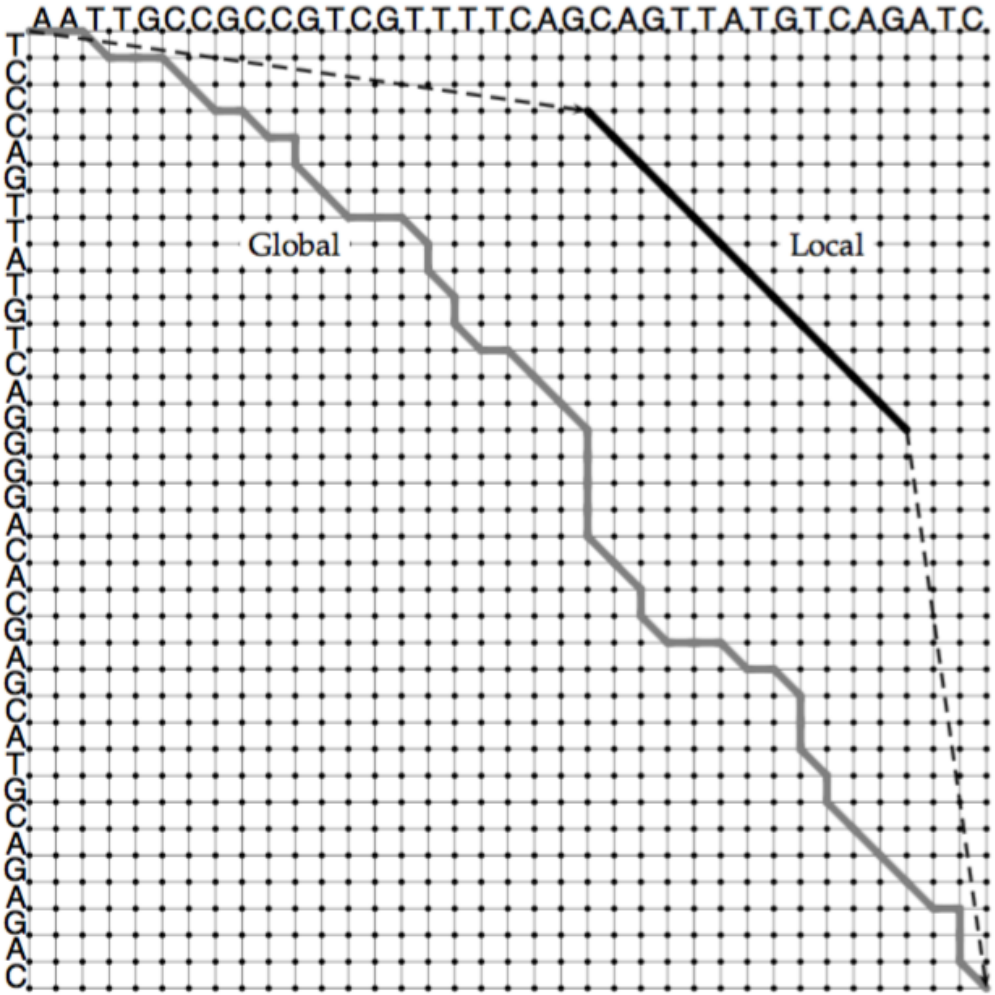
What is the other alignment?

```
GAT-TAGTAC
||. || ||
GACCTA-TA-

GA-TTAGTAC
|| .|| ||
GACCTA-TA-
```

Differences between NW and SM

	Smith–Waterman local algorithm	**Needleman–Wunsch global algorithm**
Initialization	First row and first column are set to 0	First row and first column are subject to gap penalty
Scoring	Negative score is set to 0	Score can be negative
Traceback	Begin with the highest score, end when 0 is encountered	Begin with the cell at the lower right of the matrix, end at top left cell

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C

                    tccCAGTTATGTCAGgggacacgagcatgcagagac
                       ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```



Example showing when global and local alignments really differ

Smith-Waterman Local alignment

match = 1 mismatch = -1 gap = -1

		A	A	C	G	T	A	C	A	G	T	C	T	A
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	1	1	0	0	0	1	0	1	0	0	0	0	1
A	0	1	2	1	0	0	1	0	1	0	0	0	0	1
G	0	0	1	1	2	1	0	0	0	2	1	0	0	0
T	0	0	0	0	1	3	2	1	0	1	3	2	1	0
C	0	0	0	1	0	2	2	3	2	1	2	4	3	2
T	0	0	0	0	0	1	1	2	2	1	2	3	5	4
A	0	1	1	0	0	0	2	1	3	2	1	2	4	6
G	0	0	0	0	1	0	1	1	2	4	3	2	3	5
C	0	0	0	1	0	0	0	2	1	3	3	4	3	4
C	0	0	0	1	0	0	0	1	1	2	2	4	3	3
G	0	0	0	0	2	1	0	0	0	2	1	3	3	2
C	0	0	0	1	1	1	0	1	0	1	1	2	2	2
G	0	0	0	0	2	1	0	0	0	1	0	1	1	1
C	0	0	0	1	1	1	0	1	0	0	0	1	0	0
A	0	1	1	0	0	0	2	1	2	1	0	0	0	1

```
Global
AACGTAC-AGTCT---A
|| || | || |     |
AA-GT-CTAGCCGCGCA
Local
AGTCTA
||||||
AGTCTA
```

Negative scoring matrix cells are set to zero. Traceback procedure starts at the highest scoring matrix cell and proceeds until a cell with score zero is encountered.

Alignment of any letter system given a scoring matrix

(A)

		I	S	A	L	I	G	N	E	D
	0	0	0	0	0	0	0	0	0	0
T	0	-1	1	0	-1	-1	-2	0	-1	-1
H	0	-3	-2	-1	-3	-4	-3	-1	0	-2
I	0	-4	-5	-3	1	1	-7	-6	-4	-3
S	0	-2	8	0	-5	-1	1	-6	-6	-4
L	0	2	0	7	4	-3	-5	-2	-9	-10
I	0	4	0	-1	9	8	0	-8	-5	-12
N	0	-3	5	-2	1	6	8	6	-2	-4
E	0	-3	-3	4	-4	-2	4	8	11	3

(B)

```
THIS-LI-NE-
  ||  ||  ||
--ISALIGNED
```

Scoring matrix

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	11																			
S	1	2																		
T	-1	1	2																	
P	-2	1	1	6																
A	-1	1	2	1	2															
G	-1	1	-1	-1	1	5														
N	-1	1	1	-1	0	0	3													
D	-3	0	-1	-2	0	1	2	5												
E	-4	-1	-1	-2	-1	0	1	4	5											
Q	-3	-1	-1	0	-1	-1	0	1	2	5										
H	0	-1	-1	0	-2	-2	1	0	0	2	6									
R	-1	-1	-1	-1	-1	0	0	-1	0	2	2	5								
K	-3	-1	-1	-2	-1	-1	1	0	1	2	1	4	5							
M	-2	-1	0	-2	-1	-3	-2	-3	-3	-2	-2	-2	-2	6						
I	-2	-1	1	-2	0	-3	-2	-3	-3	-3	-3	-3	-3	3	4					
L	-3	-2	-1	0	-1	-4	-3	-4	-4	-2	-2	-3	-3	3	2	5				
V	-2	-1	0	-1	1	-2	-2	-2	-2	-3	-3	-3	-3	2	4	2	4			
F	0	-2	-2	-3	-3	-5	-3	-5	-5	-4	0	-4	-5	0	0	2	0	8		
Y	2	-1	-3	-3	-3	-4	-1	-2	-4	-2	4	-2	-3	-2	-2	-1	-3	5	9	
W	1	-3	-4	-4	-4	-2	-5	-5	-5	-3	-3	0	-3	-3	-4	-2	-3	-1	0	15

DNA and protein scoring matrices

DNA: BLAST scoring matrix

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

BLAST

Protein: PAM 250 scoring matrix

What is the score:
CNE
CQE
12+1+4 = 17

Log-odds PAM 250 matrix

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	W	Y
X	0																			
C	12																			
S	0	2																		
T	-2	1	3																	
P	-3	1	0	6																
A	-2	1	1	1	2															
G	-3	1	0	-1	1	5														
N	-4	1	0	-1	0	0	2													
D	-5	0	0	-1	0	1	2	4												
E	-5	0	0	-1	0	0	1	3	4											
Q	-5	-1	-1	0	0	-1	1	2	2	4										
H	-3	-1	-1	0	-1	-2	2	1	1	3	6									
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6								
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5							
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6						
I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5					
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6				
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4			
F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9		
W	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10	
Y	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17

How do we chose a good scoring matrix?

Amino acid substitution matrices

1) PAM (point accepted mutation) matrix
- introduced by Margaret Dayhoff in 1978
- based on 1572 observed mutations in the phylogenetic trees of 71 families of closely related proteins
- based on log likelihood ratios of amino acid substitution between homologs versus random alignment
- PAMX matrix represents the number of substitutions (X) per 100 amino acids, extrapolated from close seqs

2) BLOSSUM (Block Substitution Matrices)
- Henikoff and Henikoff 1992
- score alignments of distantly related proteins, where PAM did not work well
- based on log likelihood ratios within blocks of conserved sequences
- BLOSUMX uses sequences less than X% identical.

BLOSUM62 Substitution matrix (e.g., used in sequence alignment scoring)

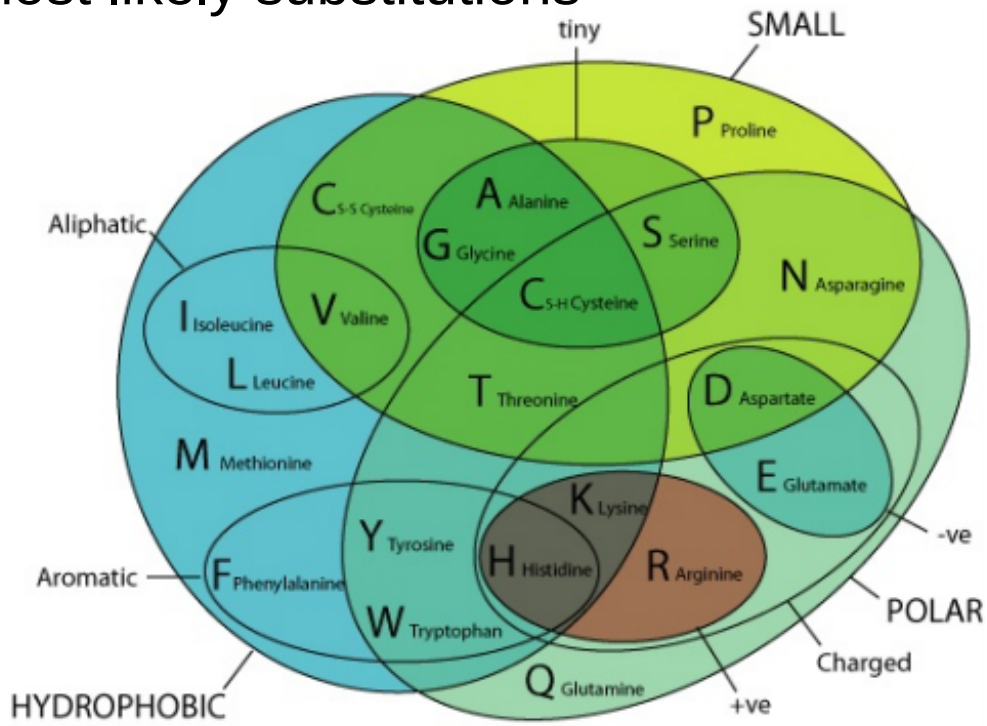
Table shows bonus or penalty score for substituting one amino acid for another

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W

Chemical-physical properties indicate most likely substitutions

Aliphatic: nonpolar and hybrophobic
Aromatic: ring

PAM matrix	Equivalent BLOSUM matrix
PAM100	Blosum90
PAM120	Blosum89
PAM160	Blosum60
PAM200	Blosum52
PAM250	Blosum45



Affine Gap Penalty

Scoring Alignment
Gap = -2
Mismatch = -1
Match = 1
Match = 2

```
     -GACCCATACGGTTCAA
#1    ||  ..||.|||||||
     TGA-ATATTCGGTTCAA

     -GACCCATA--CGGTTCAA
#2    ||   |||  |||||||
     TGA---ATATTCGGTTCAA
```

Should we really score this as three gaps. Maybe it should have lower penalty?

Score = Match+Mismatch+GAP

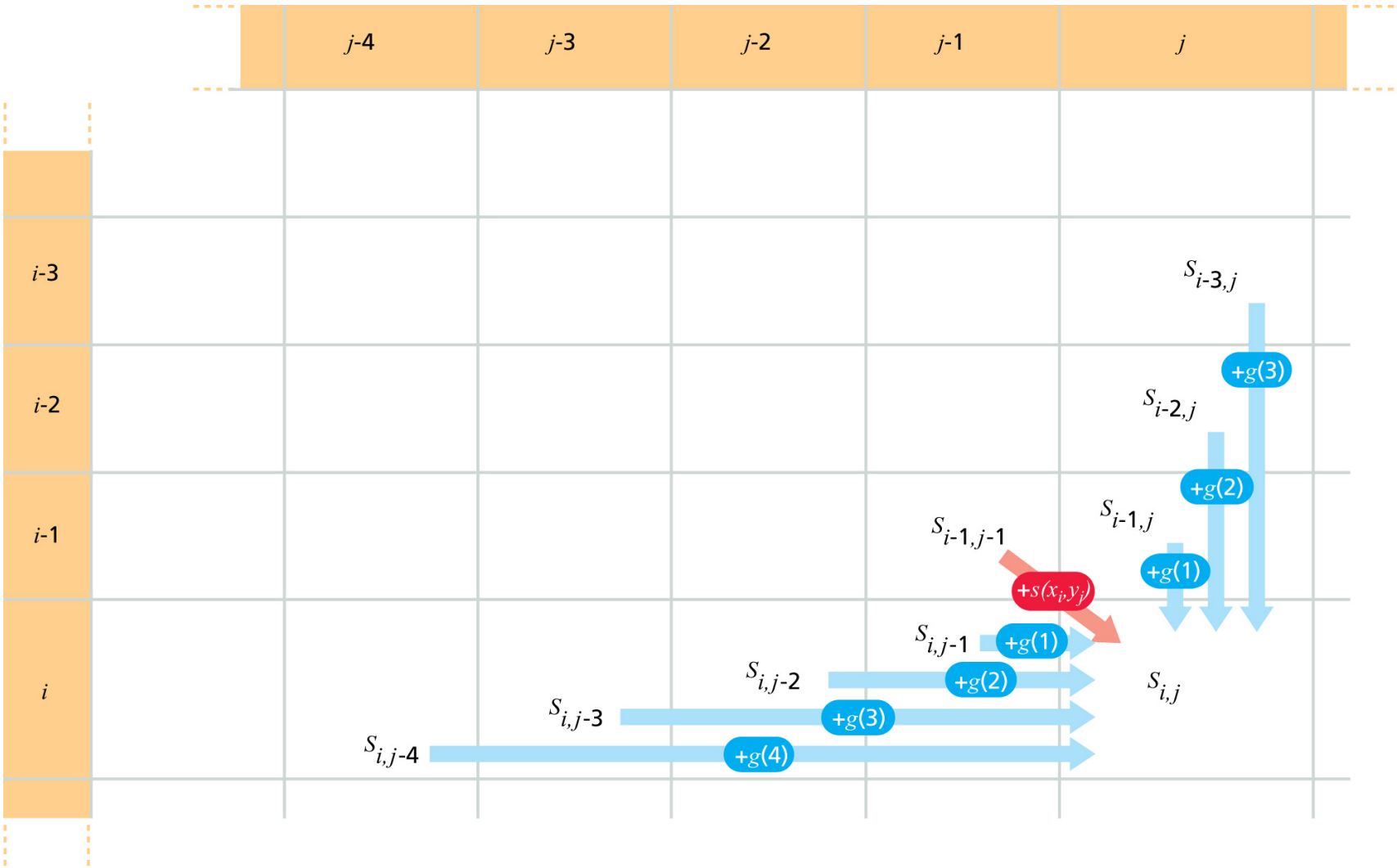
Alignment #1
12-3-4 = 5
24-3-4 = 17
24-3-5 = 16 (affine)
Alignment #2
13-0-12 = 1
26-0-12 = 14
26-0-9 = 17 (affine)
GAP = -2.5 -3.5 -3 = 9

Affine gap penalty = A + B*L
Gap open = A
Gap extension = B
Gap length = L
Example: A = -2, B = -0.5

Using affine gap penalties

$$g(n_{gap}) = -I - (n_{gap} - 1)E$$

I = *gap open*
E = *gap extension*



Time complexity O(n*m) with three matrices:
- start/continue gaps in sequence a
- start/continue gaps in sequence b
- match, mismatch or end gap

Exercises

1) Fill in the scoring matrix using Needleman-Wunsch with match = 1, mismatch = -1 and gap = -2

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

2) How many optimal alignments are there?

3) When using the affine gap penalty, do you get more gaps with B = -1 or B = -2, affine gap penalty = A+B*L?

4) Write down the NW and SW alignments given the following matrices.

	A	I	A	Y	G	L	D	K	K	G	K	E	E	H
A	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80	-88	-96	-104 -112
V	-8	4	-4	-12	-20	-28	-36	-44	-52	-60	-68	-76	-84	-92 -100
N	-16	-4	7	-1	-9	-17	-25	-33	-41	-49	-57	-65	-73	-81 -89
F	-24	-12	-1	5	-3	-9	-17	-24	-32	-40	-48	-56	-64	-72 -80
V	-32	-20	-9	-3	8	0	-8	-16	-24	-32	-40	-48	-56	-64 -72
L	-40	-28	-17	-9	0	5	1	-7	-15	-23	-31	-39	-47	-55 -63
K	-48	-36	-25	-17	-8	-3	9	1	-7	-15	-23	-31	-39	-47 -55
Q	-56	-44	-33	-25	-16	-10	1	8	6	-2	-10	-18	-26	-34 -42
R	-64	-52	-41	-33	-24	-18	-7	1	9	7	-1	-9	-16	-24 -32
Q	-72	-60	-49	-41	-32	-26	-15	-7	3	11	5	1	-7	-15 -23
F	-80	-68	-57	-49	-40	-34	-23	-15	-5	4	9	6	3	-5 -13
P	-88	-76	-65	-57	-46	-42	-31	-23	-13	-4	1	6	3	0 -6
P	-96	-84	-73	-65	-54	-48	-39	-31	-21	-12	-6	0	5	2 -2
G	-104	-92	-81	-73	-62	-56	-47	-39	-29	-20	-14	-7	-1	4 0
E	-112	-100	-89	-81	-70	-56	-55	-47	-37	-28	-14	-15	-9	-3 2
Q	-120	-108	-97	-89	-78	-64	-59	-53	-45	-36	-22	-13	-10	-4 -3
Q	-128	-116	-105	-97	-86	-72	-66	-59	-52	-44	-30	-21	-11	-8 -4
H	-136	-124	-113	-105	-94	-80	-74	-66	-58	-51	-38	-29	-19	-9 -8
	-144	-132	-121	-113	-102	-88	-82	-74	-66	-59	-46	-37	-27	-17 -1

sestoft@itu.dk

	A	I	A	Y	G	L	D	K	K	G	K	E	E	H
A	0	0	0	0	0	0	0	0	0	0	0	0	0	0 0
V	0	4	0	4	0	0	0	0	0	0	0	0	0	0 0
N	0	0	7	0	3	0	1	0	0	0	0	0	0	0 0
F	0	0	0	5	0	3	0	2	0	0	0	0	0	0 1
V	0	0	0	0	8	0	3	0	0	0	0	0	0	0 0
L	0	0	3	0	0	5	1	0	0	0	0	0	0	0 0
K	0	0	2	2	0	0	9	1	0	0	0	0	0	0 0
Q	0	0	0	1	0	0	1	8	6	5	0	5	1	1 0
R	0	0	0	0	0	0	0	1	9	7	3	1	7	3 1
Q	0	0	0	0	0	0	0	0	3	11	5	5	1	7 3
F	0	0	0	0	0	0	0	0	1	4	9	6	7	3 7
P	0	0	0	0	3	0	0	0	0	0	1	6	3	4 2
P	0	0	0	0	0	1	0	0	0	0	0	0	5	2 2
G	0	0	0	0	0	0	0	0	0	0	0	0	0	4 0
E	0	0	0	0	0	6	0	0	0	0	6	0	0	0 2
Q	0	0	0	0	0	0	3	2	1	1	0	7	5	5 0
Q	0	0	0	0	0	0	0	3	3	2	0	1	9	7 5
H	0	0	0	0	0	0	0	0	4	4	0	1	3	11 7
	0	0	0	0	2	0	0	0	0	3	2	0	1	3 19

sestoft@itu.dk

5) What is the complexity of Needleman-Wunsch algorithm and Smith-Waterman?

6) Align these two sequences using Smith-Waterman with a gap penalty of -8 and Blosum62 scoring.

	-	H	E	A	G	A	W	G	H	E	E
-	0	0	0	0	0	0	0	0	0	0	0
P	0										
A	0										
W	0										
H	0										
E	0										
A	0										
E	0										
A	0										

BLOSUM62 Substitution matrix (e.g., used in sequence alignment scoring)

Table shows bonus or penalty score for substituting one amino acid for another

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W

7) Local or global alignments:

a) query can only be represented once

b) handles rearrangements