Comparison of sequences

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Dr. Jaspreet Kaur Dhanjal
Assistant Professor, Center for Computational Biology
Email ID: jaspreet@iiitd.ac.in

August 18, 2025

What and Why?

```
ACCGGTATCCTAGGAC
|||  |||| ||||||
ACC--TATCTTAGGAC
```

- Are these two sequences related?
- How similar (or dissimilar) are they?
- Matching the two sequences as closely as possible = alignment
- Comparing alignment needs a score

- DNA and Proteins are based on linear sequences
- Homology based gene prediction
- Similar sequence have similar structure & function
- Protein function annotation
- Protein structure modeling
- Assembly of genomes
- Searching for mutations or polymorphism
- Phylogenetic analysis

Pairwise sequence alignment

Types of alignment

Global alignment

To compare sequences of similar sizes

- Compare closely related genes
- Search for mutations or polymorphisms in a sequence compared to a reference.

```
ACCGGTATCCTAGGAC
|||| |||| ||||||
ACCG-TATCTTAGGAC
```

Local alignment

To find shared subsequences

- Search for protein domains
- Find gene regulatory elements
- Locate a similar gene in a genome sequence.

```
ATGCGCTACCGTATCCTAGGAC
       |||||||| ||
-------ACCGTATC-TA----
```

End free alignment

To find joins/overlaps

- Align the sequences from adjacent sequencing primers

```
CGCTACC   TCCTAGGAC
    |||   ||||
    ACCGTATCCT
         ↓
CGCTACCGTATCCTAGGAC
```

DOT PLOT

Dot plot **is a graphical method that allows the comparison of two biological sequences and identify regions of close similarity between them.**

Seq1: TWILIGHTZONE
Seq2: MIDNIGHTZONE

Put a dot, or 1,
where ever there is identity

	T	W	I	L	I	G	H	T	Z	O	N	E
M												
I												
D												
N												
I												
G												
H												
T												
Z												
O												
N												
E												

	T	W	I	L	I	G	H	T	Z	O	N	E
M												
I			●		●							
D												
N											●	
I			●		●							
G						●						
H							●					
T	●							●				
Z									●			
O										●		
N											●	
E												●

DOT PLOT

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

Noisy

Word size: size of sequence block used for comparison. Here, the window size is 1.

Stringency: Number of matches required to score positive. In this example stringency is 1 (requires exact match).

Word size= 4; Stringency= 2

```
GATCGTACCATGGAATCGTCCAGATCA
GATC
 GATC
  GATC
   GATC
```

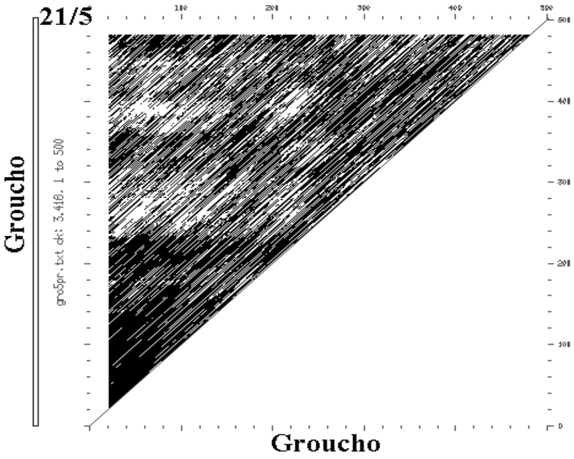
+(4/4)

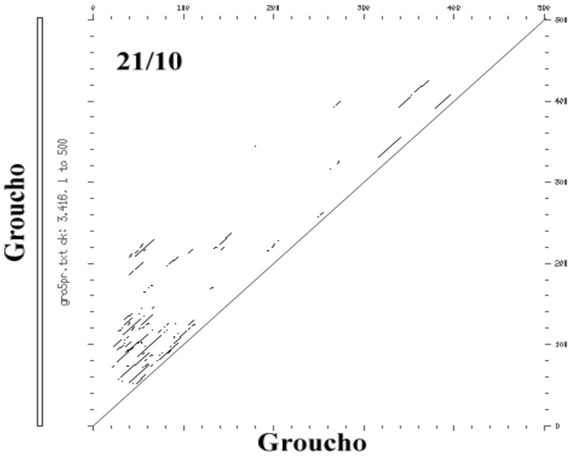
- (0/4)

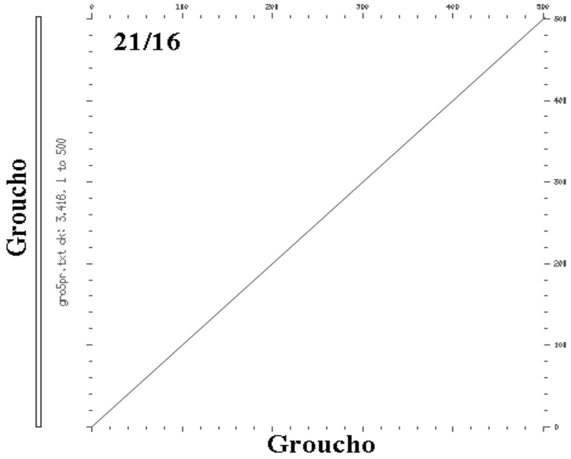
- (0/4)

- (1/4)

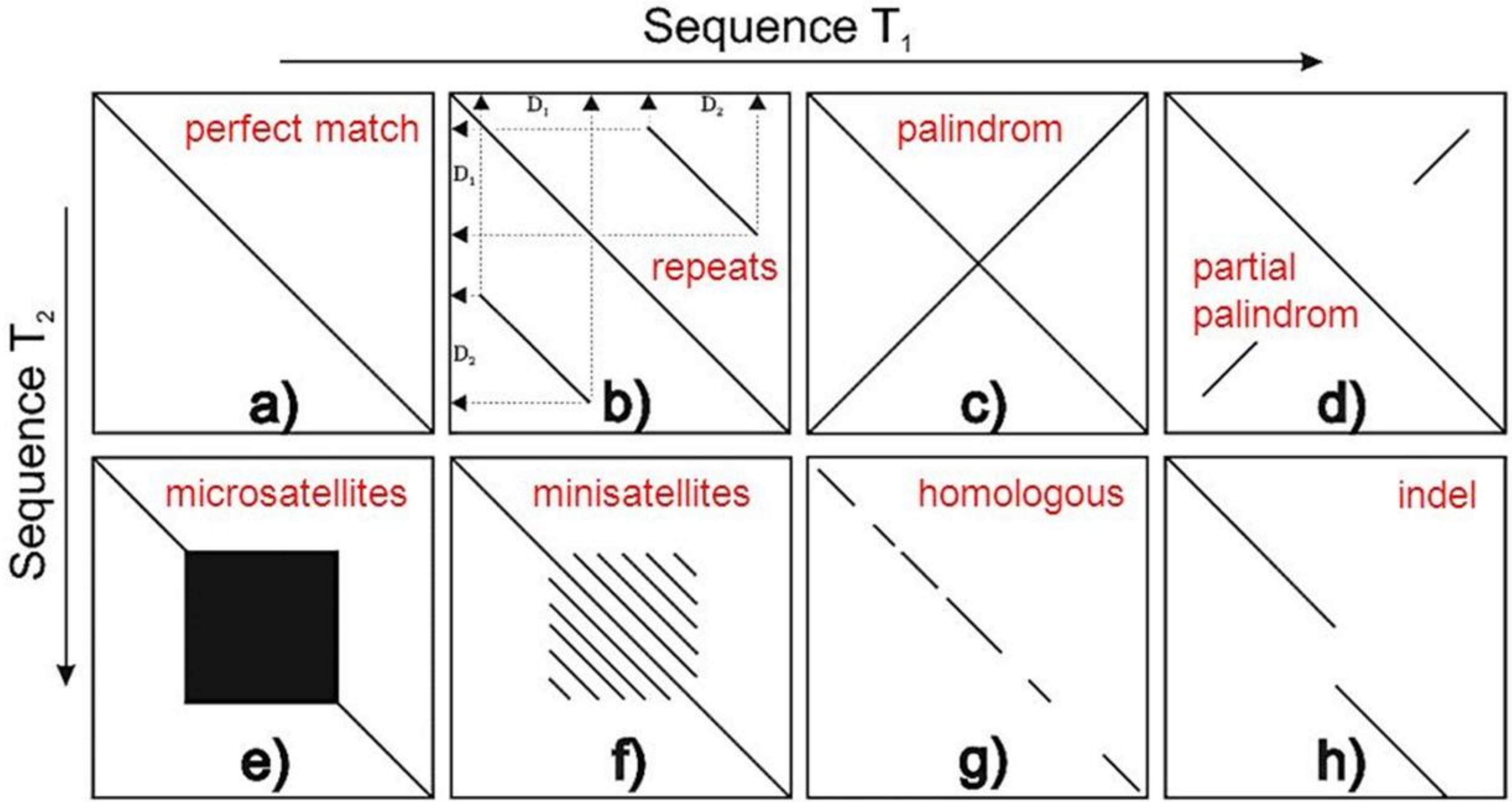
DOT PLOT


21/5
Groucho
gro5pr.txt ck: 3,416, 1 to 500
Groucho
100
200
300
400
500
0
100
200
300
400
500


21/10
Groucho
gro5pr.txt ck: 3,416, 1 to 500
Groucho
0
100
200
300
400
500
0
100
200
300
400
500


21/16
Groucho
gro5pr.txt ck: 3,416, 1 to 500
Groucho
0
100
200
300
400
500
0
100
200
300
400
500

Interpretation of DOT PLOT



Limitations of DOT PLOT

- Rely on visual analysis
- Difficult to find optimal alignments
- Difficult to estimate significance of alignments
- Insensitive to conserved substitutions (e.g. L ↔ I or S ↔T)
- Compares only two sequences (vs. multiple alignment)
- Time consuming (1,000 bp vs. 1,000 bp = 10^6 operations,
  1,000,000 bp vs. 1,000,000 bp = 10^{12} operations)

Simple Alignment

```
ACCGGTATCCTAGGAC             ACCGGTATCCTAGGAC
                    ──────▶  |||  |||| ||||||
ACCTATCTTAGGAC               ACC--TATCTTAGGAC
```

Score: a match (+1), a mismatch (-1), a gap (-1)

```
ACCGGTATCCTAGGAC
|||  |||| ||||||
ACC--TATCTTAGGAC
```

Total Score: (13x1) + (1x-1) +(2x-1) = 10

***<u>Limitation</u>*: number of alignments between two sequences is exponential**
very slow algorithm

***<u>Solution</u>*: Dynamic programming**

Optimal alignment

The optimal alignment of two similar sequences usually

- maximizes the number of matches and
- minimizes the number of gaps.

Permitting the insertion of arbitrarily many gaps might lead to high scoring alignments of non-homologous sequences.

Penalizing gaps forces alignments to have relatively few gaps.

Gap penalties increase the quality of an alignment – non-homologous sequences are not aligned.

Global Alignment

```
ACCGGTATCCTAGGAC
|||  |||| ||||||
ACC--TATCTTAGGAC
```

- To compare sequences of similar sizes
- Gaps are inserted into, or at the ends of each sequence
- The sequence length (bases+gaps) are identical for each sequence
- Every base or gap in each sequence is aligned with a base or a gap in the other sequence

Let's start by trying out a simple example of alignment:

S = `ACCGGTAT`

T = `ACCTATC`

Simple scoring scheme (2 for match, -1 for mismatch, -1 for gap)

Substitution/Match matrix for a simple alignment

An alignment between base *i* in S and base *j* in T is represented: (S*i*, T*j*)

The score for this occurring is represented: σ(S*i*, ,T*j*)

Global Alignment

Length of S = *m*= 8

Length of T = *n* = 7

OK for Global alignment

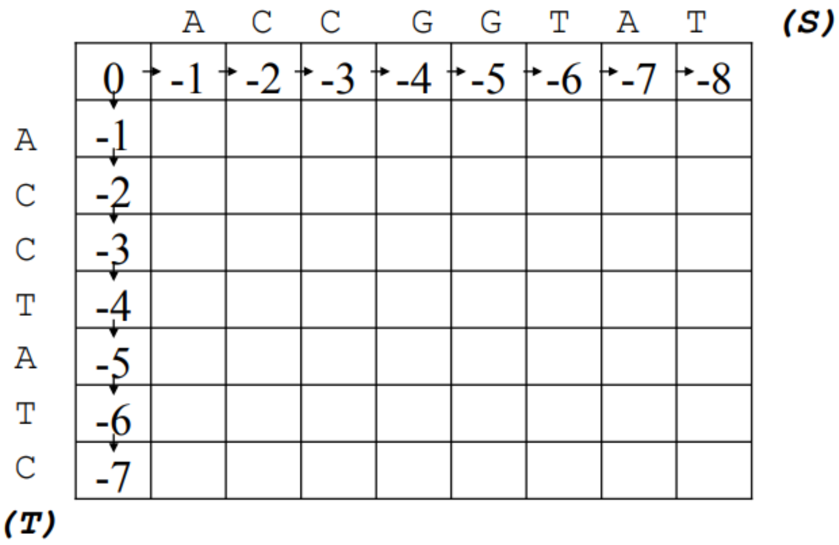
		A	C	C	G	G	T	A	T
A									
C									
C									
T									
A									
T									
C									

(S)

(T)

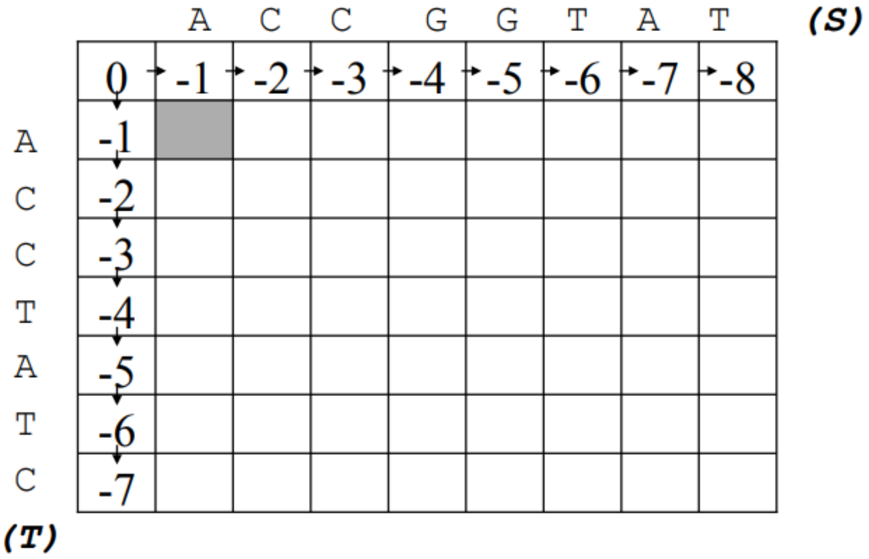
Global Alignment

Represent scores for gaps in row/col 0



		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1									
C	-2									
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

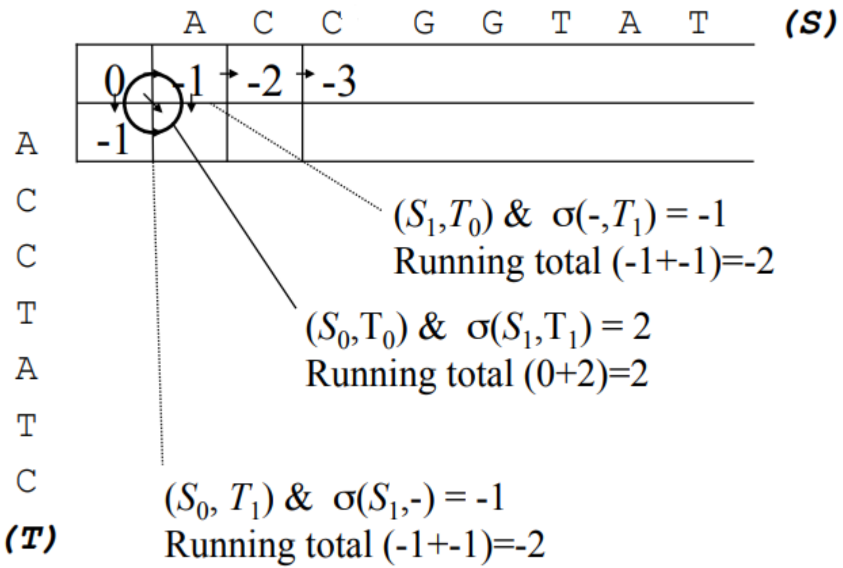
For each cell consider the 'best' path



		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	▨								
C	-2									
C	-3									
T	-4									
A	-5									
T	-6									
C	-7									
(T)										

Global Alignment

For each cell consider the 'best' path

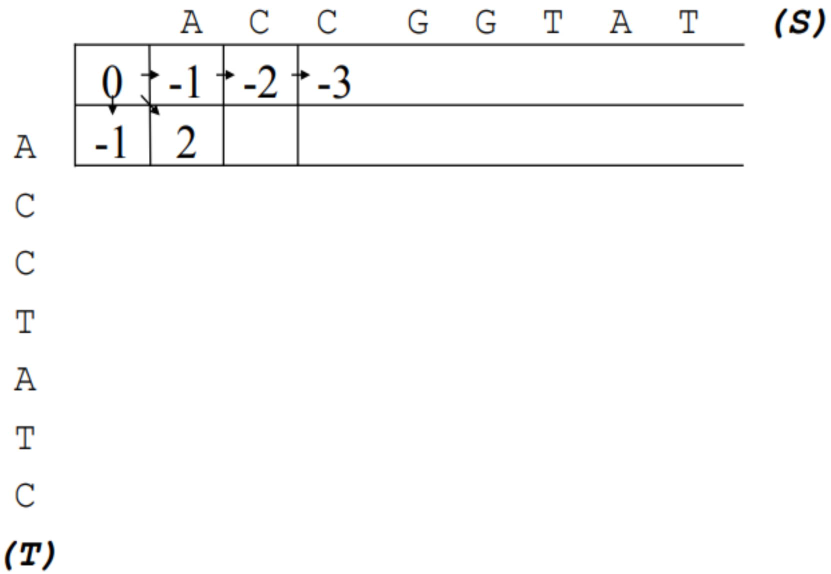


(S_1,T_0) & $\sigma(-,T_1) = -1$
Running total (-1+-1)=-2

(S_0,T_0) & $\sigma(S_1,T_1) = 2$
Running total (0+2)=2

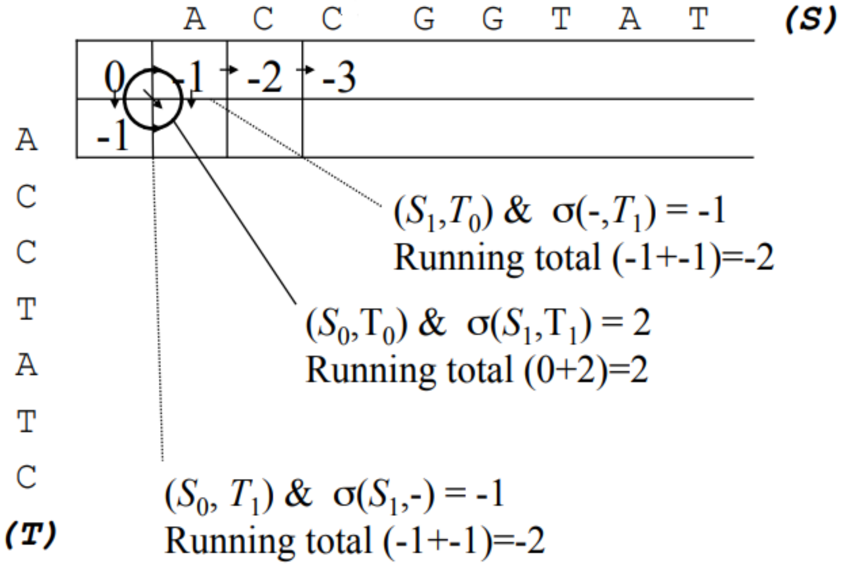
(S_0, T_1) & $\sigma(S_1,-) = -1$
Running total (-1+-1)=-2

Record the 'best' path



Global Alignment

For each cell consider the 'best' path

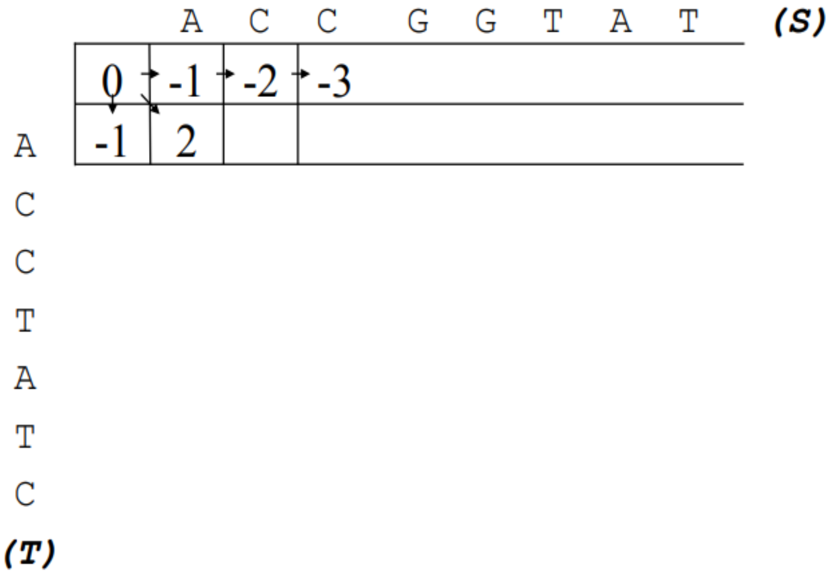


(S_1,T_0) & $\sigma(-,T_1) = -1$
Running total (-1+-1)=-2

(S_0,T_0) & $\sigma(S_1,T_1) = 2$
Running total (0+2)=2

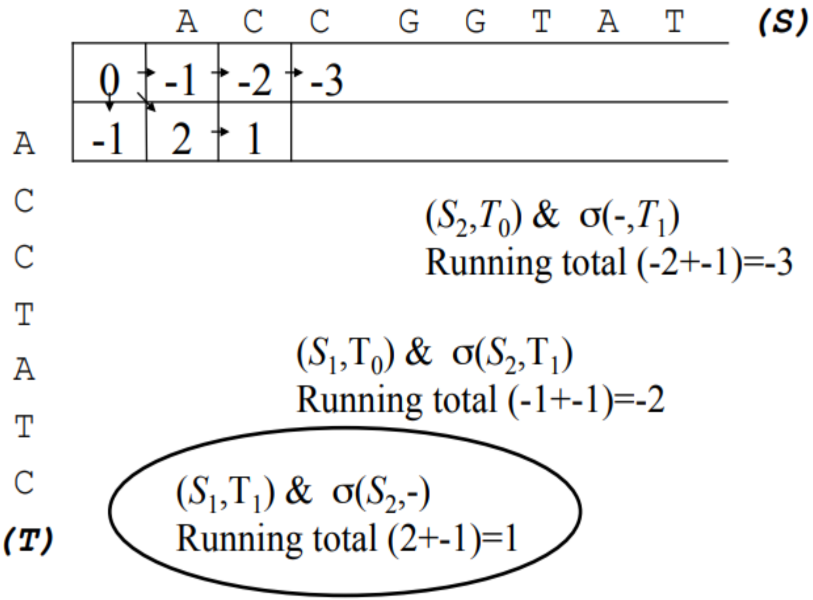
(S_0, T_1) & $\sigma(S_1,-) = -1$
Running total (-1+-1)=-2

Record the 'best' path



Global Alignment

For each cell consider the 'best' path



		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3						
A	-1	2	1							
C										
C										
T										
A										
T										
C										
(T)										

(S_2,T_0) & $\sigma(-,T_1)$
Running total (-2+-1)=-3

(S_1,T_0) & $\sigma(S_2,\mathrm{T}_1)$
Running total (-1+-1)=-2

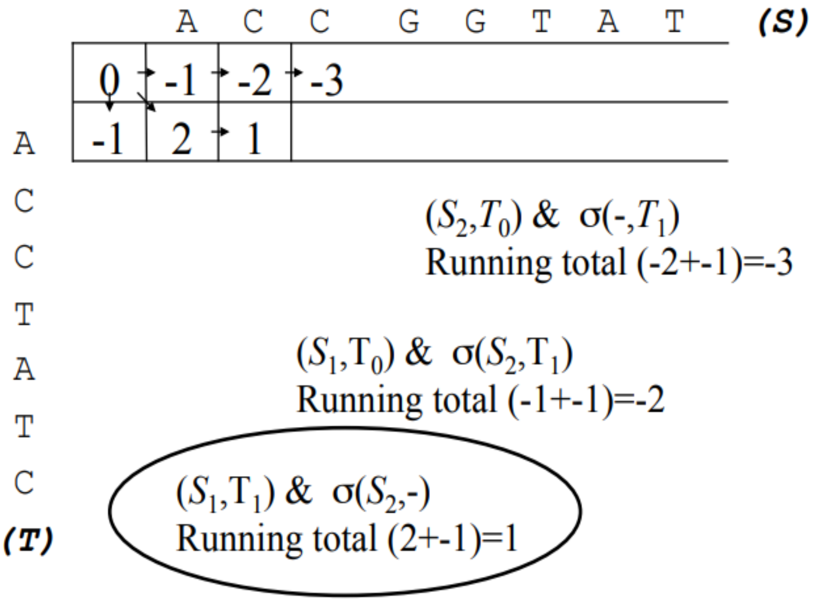
(S_1,T_1) & $\sigma(S_2,-)$
Running total (2+-1)=1

Final Matrix

		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	**9**	**= Score**
(T)										

Global Alignment

For each cell consider the 'best' path



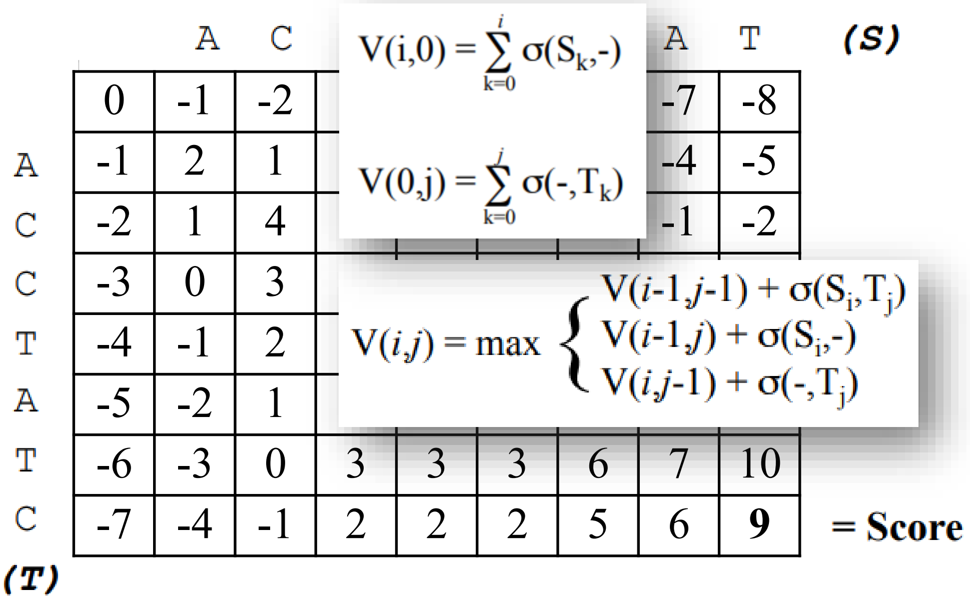
		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3						
A	-1	2	1							
C										
C										
T										
A										
T										
C										
(T)										

(S_2,T_0) & $\sigma(-,T_1)$
Running total (-2+-1)=-3

(S_1,T_0) & $\sigma(S_2,T_1)$
Running total (-1+-1)=-2

(S_1,T_1) & $\sigma(S_2,-)$
Running total (2+-1)=1

Final Matrix



		A	C					A	T	*(S)*
	0	-1	-2					-7	-8	
A	-1	2	1					-4	-5	
C	-2	1	4					-1	-2	
C	-3	0	3							
T	-4	-1	2							
A	-5	-2	1							
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	**9**	**= Score**
(T)										

$$V(i,0) = \sum_{k=0}^{i} \sigma(S_k,-)$$

$$V(0,j) = \sum_{k=0}^{j} \sigma(-,T_k)$$

$$V(i,j) = \max \begin{cases} V(i-1,j-1) + \sigma(S_i,T_j) \\ V(i-1,j) + \sigma(S_i,-) \\ V(i,j-1) + \sigma(-,T_j) \end{cases}$$

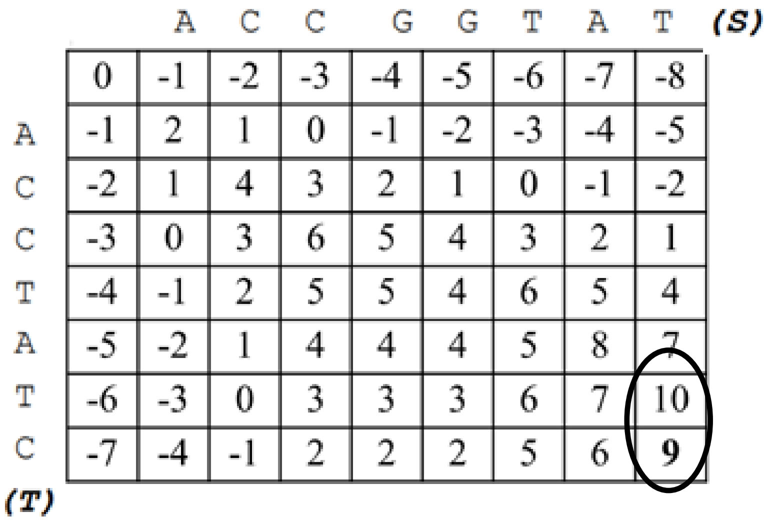
Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

- (S)

C (T)



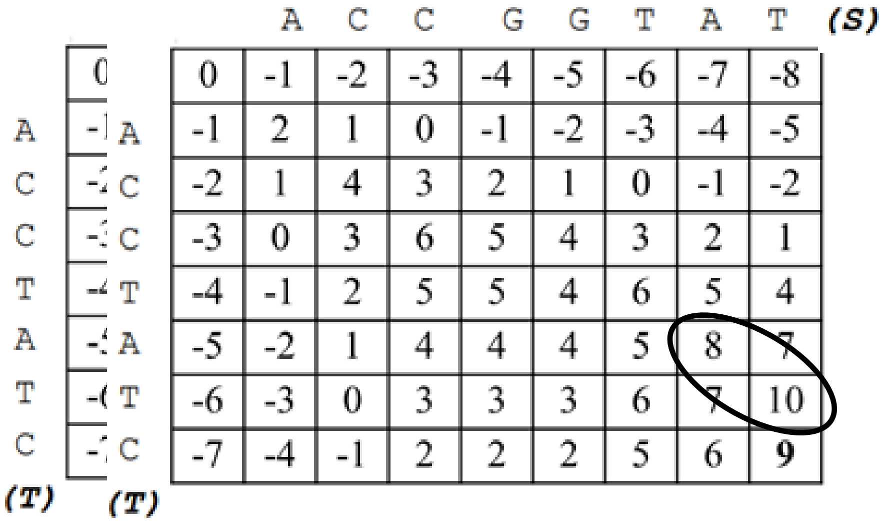
		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	**9**	
(T)										

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
T - (S)
|
T C (T)
```



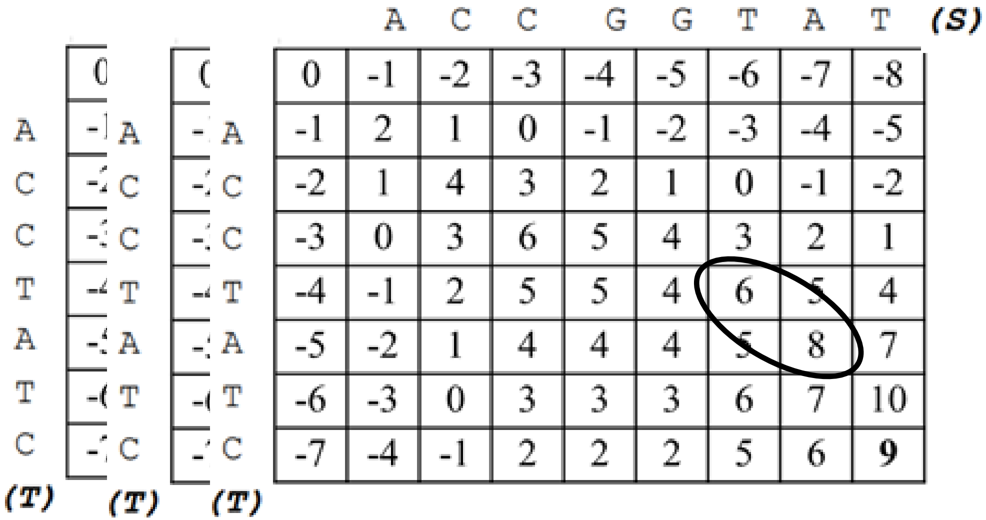
		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	9	
(T)										

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
A T - (S)
| |
A T C (T)
```



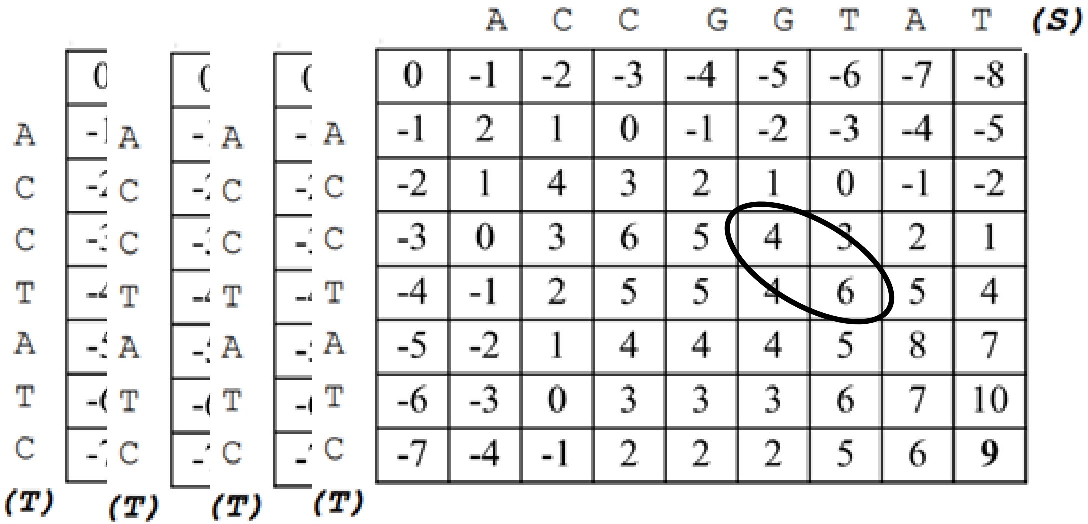
		A	C	C	G	G	T	A	T	*(S)*
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	9	
(T)										

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
T A T - (S)
| | |
T A T C (T)
```



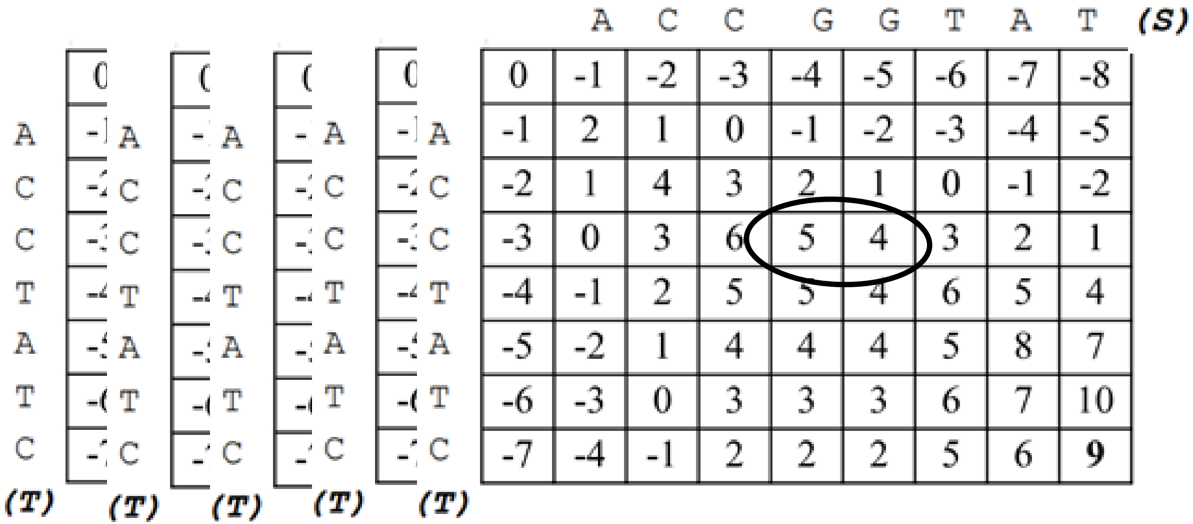
(T) \ (S)		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	5	4	6	5	4
A	-5	-2	1	4	4	4	5	8	7
T	-6	-3	0	3	3	3	6	7	10
C	-7	-4	-1	2	2	2	5	6	9

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
G T A T - (S)
  | | |
- T A T C (T)
```



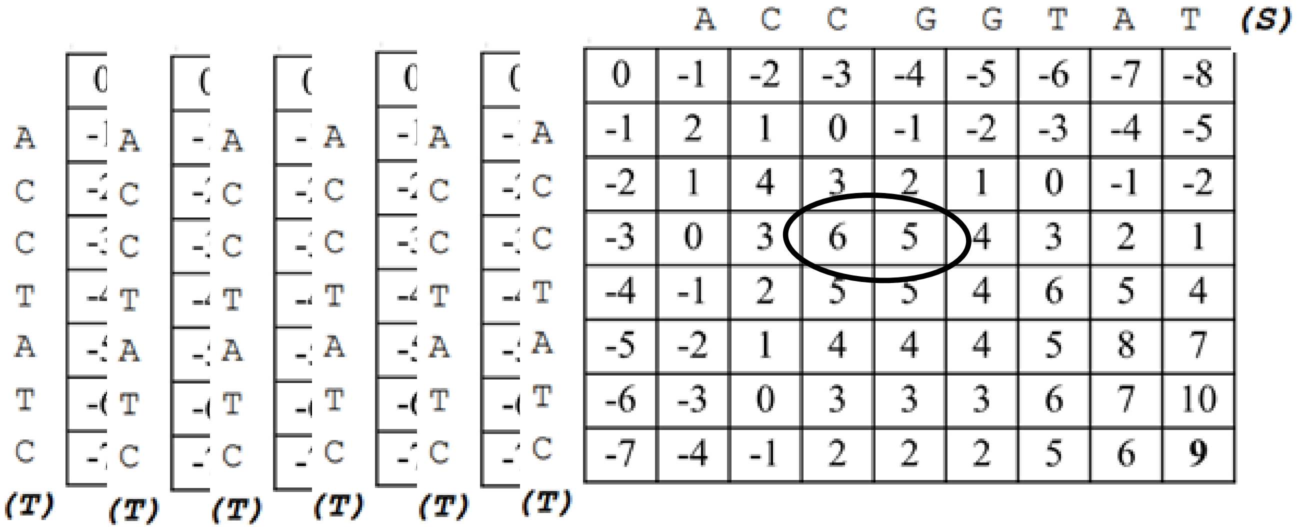
		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	9	
(T)										

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
G G T A T – (S)
    | | |
– – T A T C (T)
```



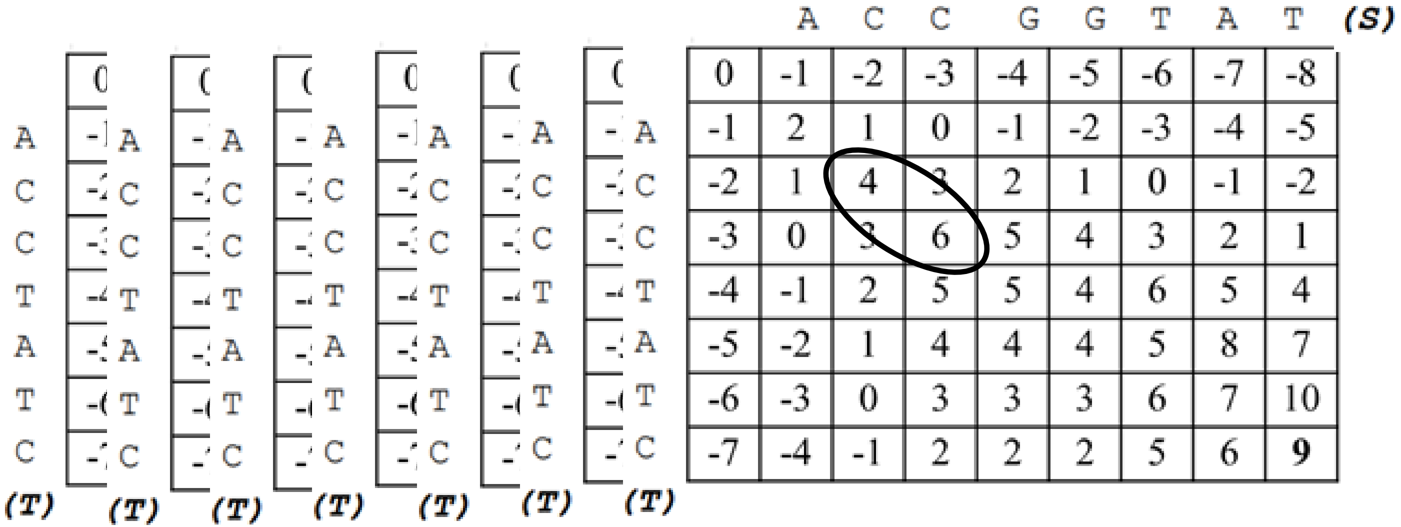
		A	C	C	G	G	T	A	T	(S)
	0	-1	-2	-3	-4	-5	-6	-7	-8	
A	-1	2	1	0	-1	-2	-3	-4	-5	
C	-2	1	4	3	2	1	0	-1	-2	
C	-3	0	3	6	5	4	3	2	1	
T	-4	-1	2	5	5	4	6	5	4	
A	-5	-2	1	4	4	4	5	8	7	
T	-6	-3	0	3	3	3	6	7	10	
C	-7	-4	-1	2	2	2	5	6	9	
(T)										

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
C G G T A T – (S)
|       | | |
C – – T A T C (T)
```



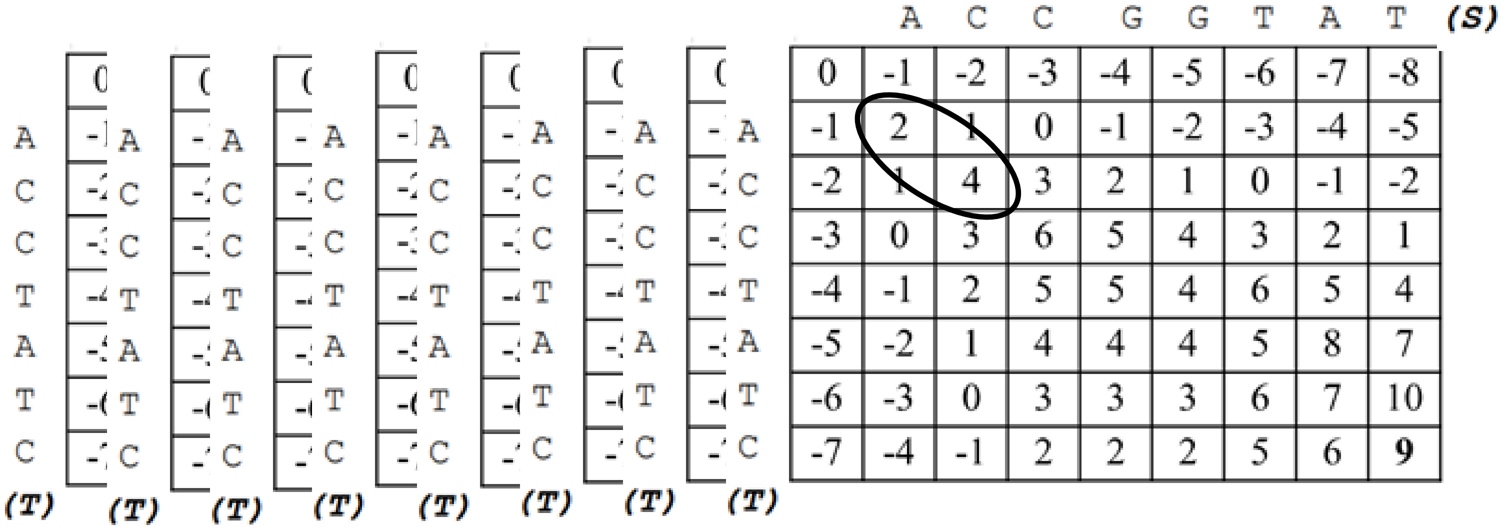
		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	5	4	6	5	4
A	-5	-2	1	4	4	4	5	8	7
T	-6	-3	0	3	3	3	6	7	10
C	-7	-4	-1	2	2	2	5	6	9

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
C C G G T A T - (S)
| |     | | |
C C - - T A T C (T)
```



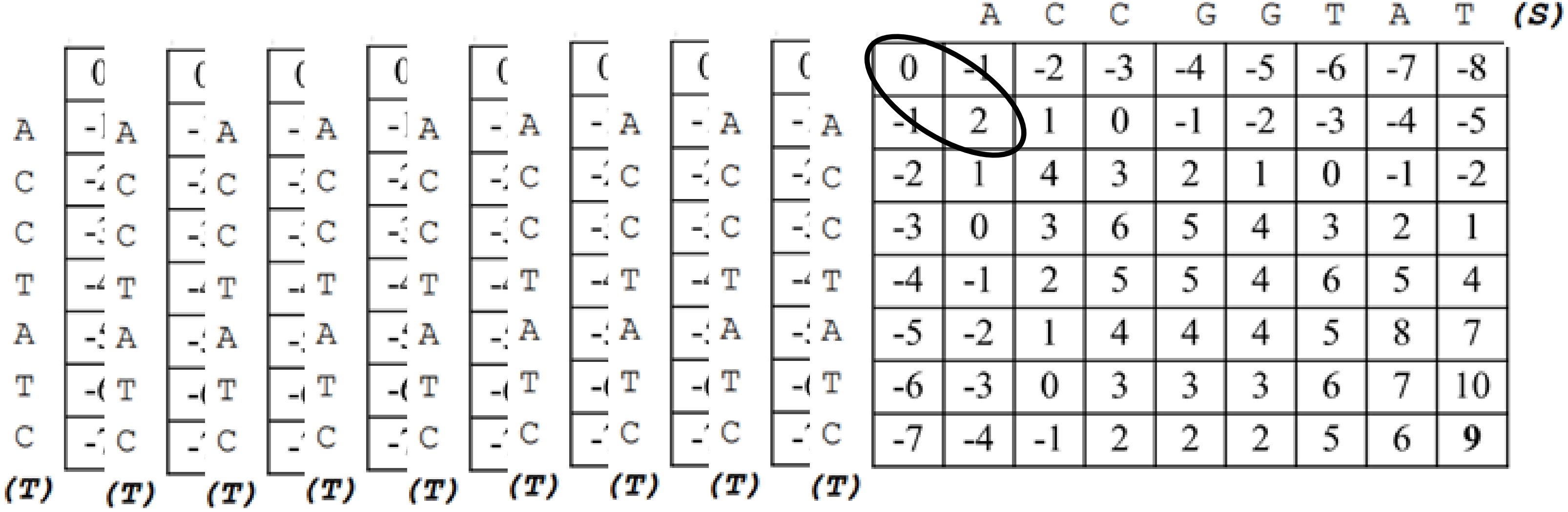
(T) \ (S)		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	5	4	6	5	4
A	-5	-2	1	4	4	4	5	8	7
T	-6	-3	0	3	3	3	6	7	10
C	-7	-4	-1	2	2	2	5	6	9

Global Alignment

Recreate the alignment by following the pointers back through the array to the origin

6 matches = 12 points, 3 gaps = -3 points
Score = 9 confirmed

```
A C C G G T A T – (S)
| | |     | | |
A C C – – T A T C (T)
```



(T) \ (S)		A	C	C	G	G	T	A	T
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
C	-2	1	4	3	2	1	0	-1	-2
C	-3	0	3	6	5	4	3	2	1
T	-4	-1	2	5	5	4	6	5	4
A	-5	-2	1	4	4	4	5	8	7
T	-6	-3	0	3	3	3	6	7	10
C	-7	-4	-1	2	2	2	5	6	9

This is also known as Needleman–Wunsch algorithm