Bioinformatics
CS300
Chap 3
Sequence Alignment:
The Needleman-Wunsch algorithm

Spring 2021
Oliver BONHAM-CARTER

Needleman-Wunsch Algorithm Background

- Global Alignment: Used to determine which parts of a sequence (inside the sequence) are shared (common) with another sequence.
- Developed by Saul B. Needleman and Christian D. Wunsch in 1970.
- Dynamic programming to find optimal solution for matching the characters of the two sequences.

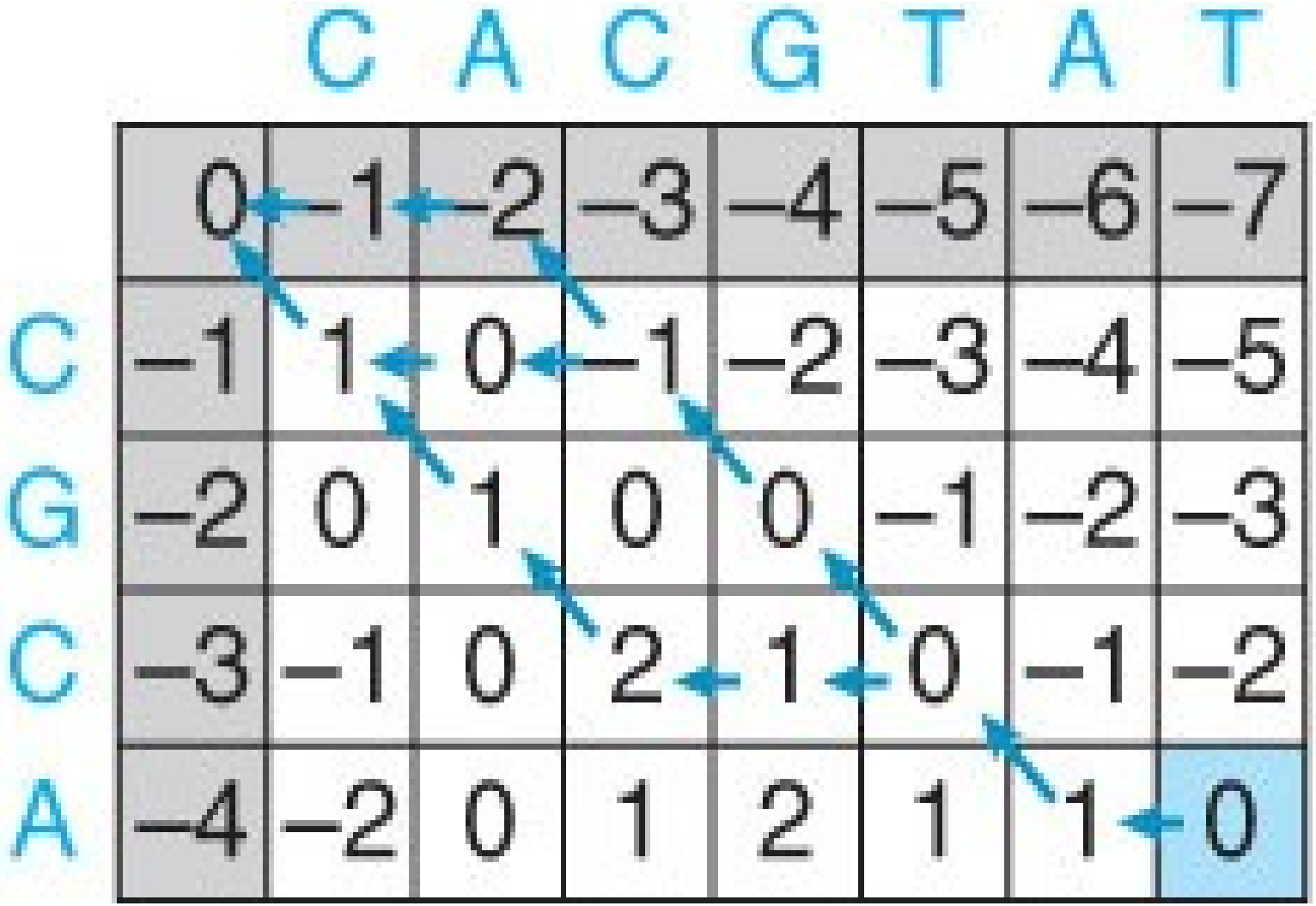
Terms

- Alignment is divided up unto sub problems
- Solutions are scored; the best solutions for char by char comparison are kept in the overall solution.
- **Match** – bases of each sequence at position ARE same
- **Mismatch** – bases of each sequence at position are NOT same
- **Gap** – bases are not the same, some insert or deletion may have occurred.

AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGTCGATTTGCCCGAC

-AGGCTATCACCT**GACCTC**CA**GGCCGA--TGCCC---**
TAG-CTATCAC--GACCGC**--GG**TCGA**TTGCCC**GAC

Global Pairwise Alignment:
Needleman-Wunsch



Algorithm

		C	A	C	G	T	A	T
	0	−1	−2	−3	−4	−5	−6	−7
C	−1	1	0	−1	−2	−3	−4	−5
G	−2	0	1	0	0	−1	−2	−3
C	−3	−1	0	2	1	0	−1	−2
A	−4	−2	0	1	2	1	1	0

- Create N x M matrix and place each sequence along one axis
- Place score 0 at the up-left corner
- Fill in 1st row & column with gap penalty multiples (here, gap penalty is -1)
- Fill in the matrix with max value of 3 possible moves:
  - Vertical move: Score + gap penalty
  - Horizontal move: Score + gap penalty
  - Diagonal move: Score + match/mismatch score
- To reconstruct the optimal alignment, trace back where the max at each step came from, stop when hit the origin.

For each Element: Three Calculations

- Recursion, based on the principle of optimality:

$$F_{ij} = \max(F_{i-1,j-1} + S(A_i, B_j),\ F_{i,j-1} + d,\ F_{i-1,j} + d)$$

The pseudo-code for the algorithm to compute the F matrix therefore looks like this:

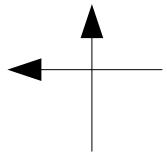
```
for i=0 to length(A)
  F(i,0) ← d*i
for j=0 to length(B)
  F(0,j) ← d*j
for i=1 to length(A)
  for j=1 to length(B)
  {
    Match ← F(i-1,j-1) + S(Ai, Bj)
    Delete ← F(i-1, j) + d
    Insert ← F(i, j-1) + d
    F(i,j) ← max(Match, Insert, Delete)
  }
```

https://en.wikipedia.org/wiki/Needleman%E2%80%93Wunsch_algorithm

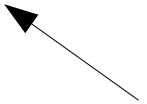
Let's Calculate!

		A	T	C	G
_					
T					
C					
A					

- Gap: -1

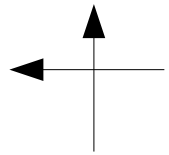


- Match: 1
- Mismatch: 0

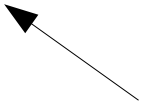


Add The Outer Values

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1				
C	-2				
A	-3				

- Gap: -1 

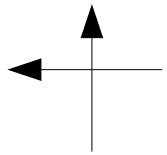
- Match: 1
- Mismatch: 0



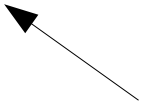
Mismatch: A != T

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1				
C	-2				
A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



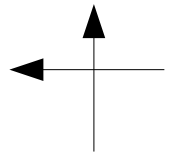
Upperbox:
-1 – 1 = -2

Sidebox:
-1 – 1 = -2

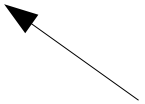
Diag:
0 – 0 = 0

Mismatch: A != T

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0			
C	-2				
A	-3				

- Gap: -1 

- Match: 1
- Mismatch: 0



Upperbox:
-1 – 1 = -2

Sidebox:
-1 – 1 = -2

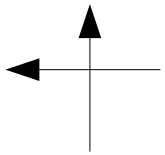
Diag:
0 – 0 = 0

Max value

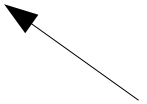
Match: T = T

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	**0**		
C	-2				
A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
-2 - 1 = -3

Sidebox:
0 – 1 = -1

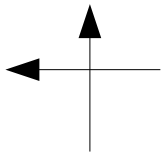
Diag:
-1 + 1 = 0

Max value

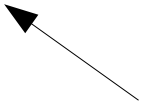
Mismatch: T != C

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	
C	-2				
A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
-3 – 1 = -4

Sidebox:
0 – 1 = -1

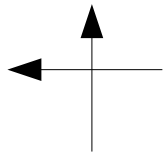
Max value

Diag:
-2 + 0 = -2

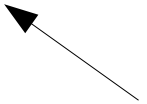
Mismatch: T != G

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2				
A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
-4 – 1 = -5

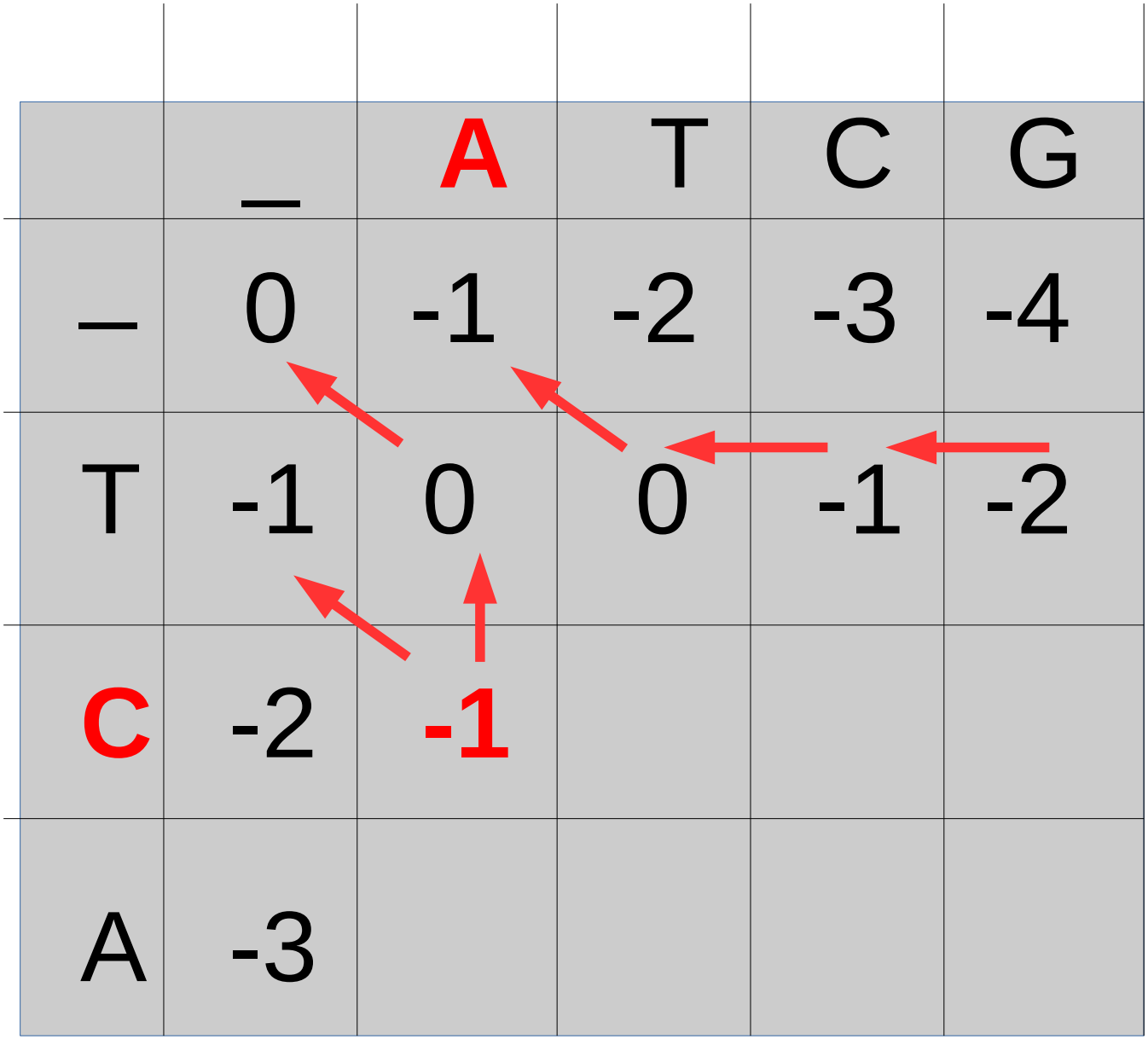
Sidebox:
-1 - 1 = -2

Max value

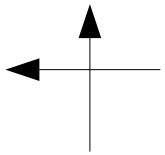
Diag:
-3 - 0 = -3

Mismatch: C != A

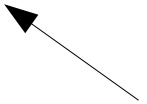
	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2	-1			
A	-3				



- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
0 - 1 = -1

Max value

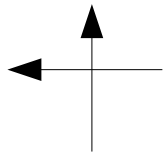
Sidebox:
-2 - 1 = -3

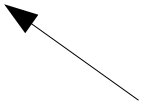
Diag:
-1 + 0 = -1

Max value

Mismatch: C != A

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2	-1	0		
A	-3				

- Gap: -1 

- Match: 1
- Mismatch: 0 

Upperbox:
0 - 1 = -1

Sidebox:
-1 - 1 = -2

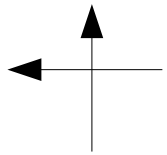
Diag:
0 + 0 = 0

Max value

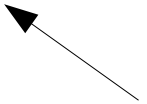
Match: C == C

		_	A	T	C	G
	_	0	-1	-2	-3	-4
	T	-1	0	0	-1	-2
	C	-2	-1	0	1	
	A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
-1 - 1 = -2

Sidebox:
0 - 1 = -1

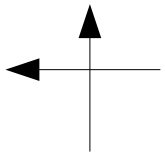
Diag:
0 + 1 = 1

Max value

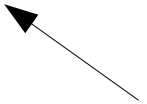
Mismatch: C != G

	_	A	T	C	**G**
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2	-1	0	1	**0**
A	-3				

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:
-2 - 1 = -3

Sidebox:
1 - 1 = 0

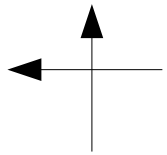
Max value

Diag:
-1 - 0 = -1

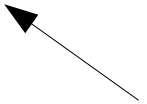
Completed With Arrows

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2	-1	0	1	0
A	-3	-1	-1	0	1

- Gap: -1



- Match: 1
- Mismatch: 0



Upperbox:

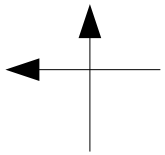
Sidebox:

Diag:

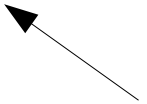
Follow the Arrows Back to Start

	_	A	T	C	G
_	0	-1	-2	-3	-4
T	-1	0	0	-1	-2
C	-2	-1	0	1	0
A	-3	-1	-1	0	1

- Gap: -1



- Match: 1
- Mismatch: 0

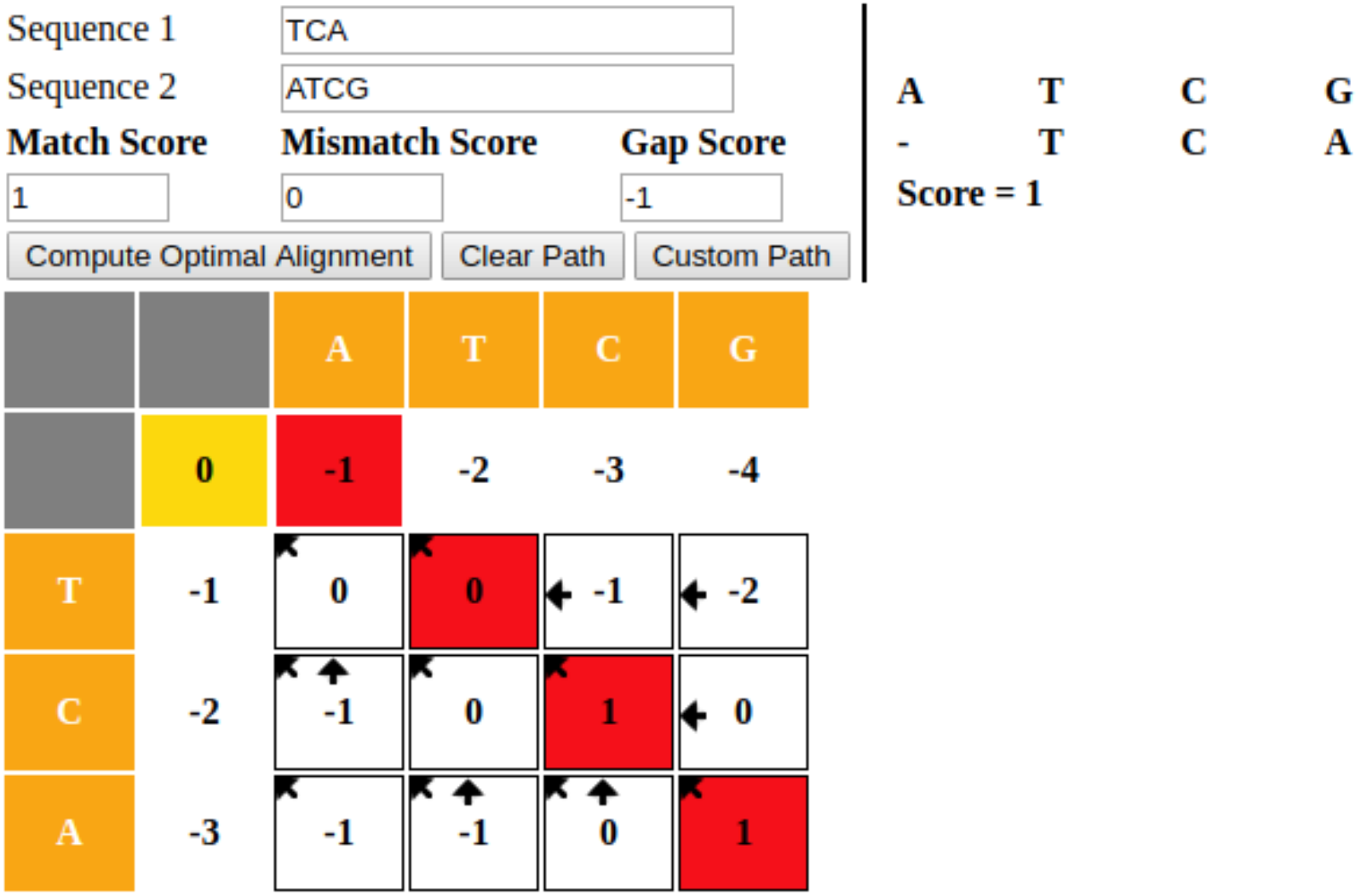


Alignment:

A T C G

_ T C A

Follow the Arrows Back To Find the Sequence Alignment



http://experiments.mostafa.io/public/needleman-wunsch/

To Get the Alignment

- With each calculation, we placed an arrow to show how the score was calculated and to give us the actual alignment.

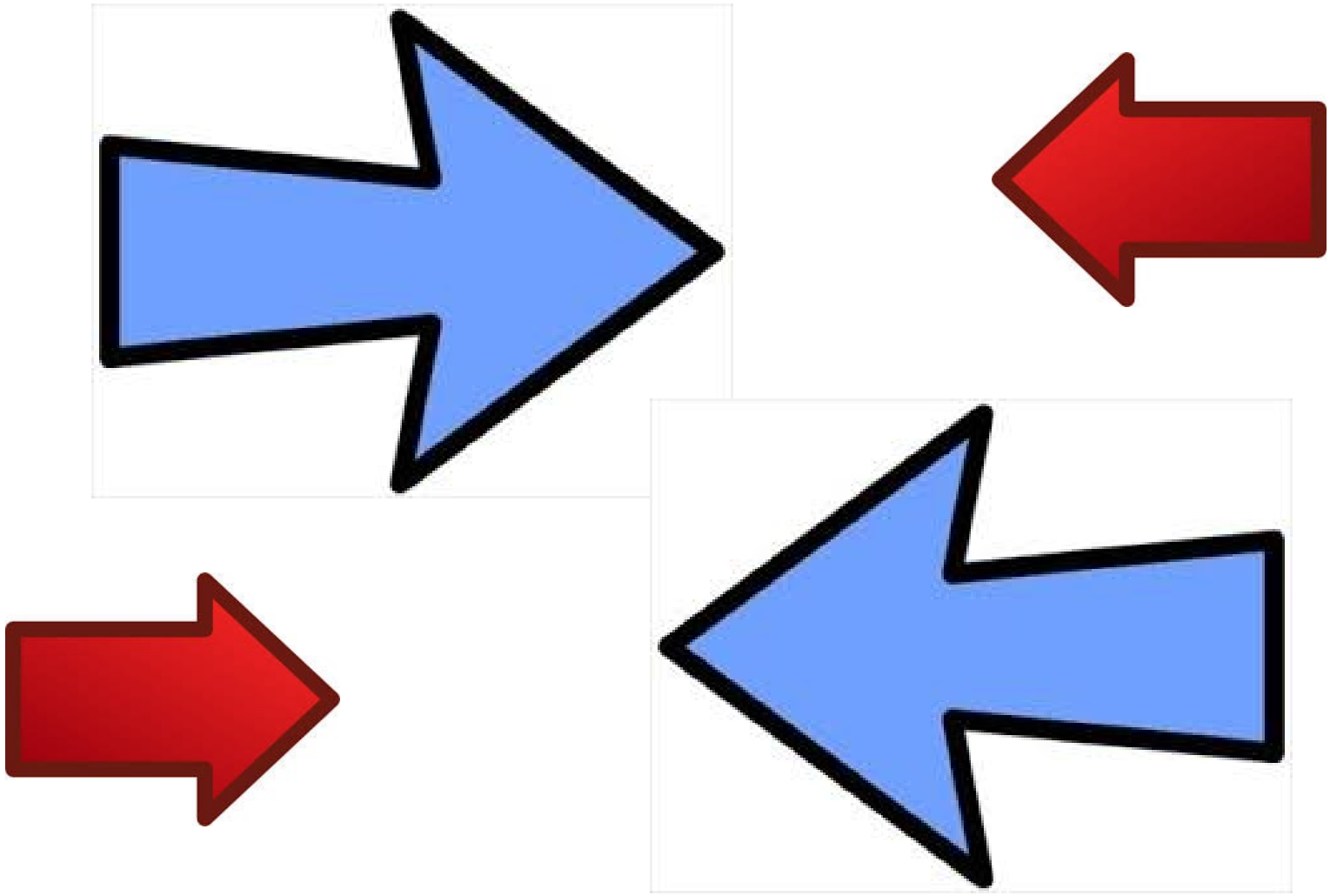
How do we read this output?

Alignment:

A T C G

_ T C A

Up Ahead:
More Examples of Arrows



Follow the Arrows!

Sequence 1	CGCA
Sequence 2	CACGTAT
Match Score	1
Mismatch Score	0
Gap Score	-1

Compute Optimal Alignment | Clear Path | Custom Path

C A C G T A
C G C - - A
Score = 0

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

- S1 = CGCA
- S2 = CACGTAT

Example Alignment score = 0

Let:
Match = +1
Mismatch = 0
Gap = -1

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

Example Alignment score = 0

Let:
Match = +1
Mismatch = 0
Gap = -1

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

CACGTAT
--CGCA-

Example Alignment score = 0

Let:
Match = +1
Mismatch = 0
Gap = -1

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

CACGTAT
C--GCA-

Example

Alignment score = 0

Let:

Match = +1

Mismatch = 0

Gap = -1

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

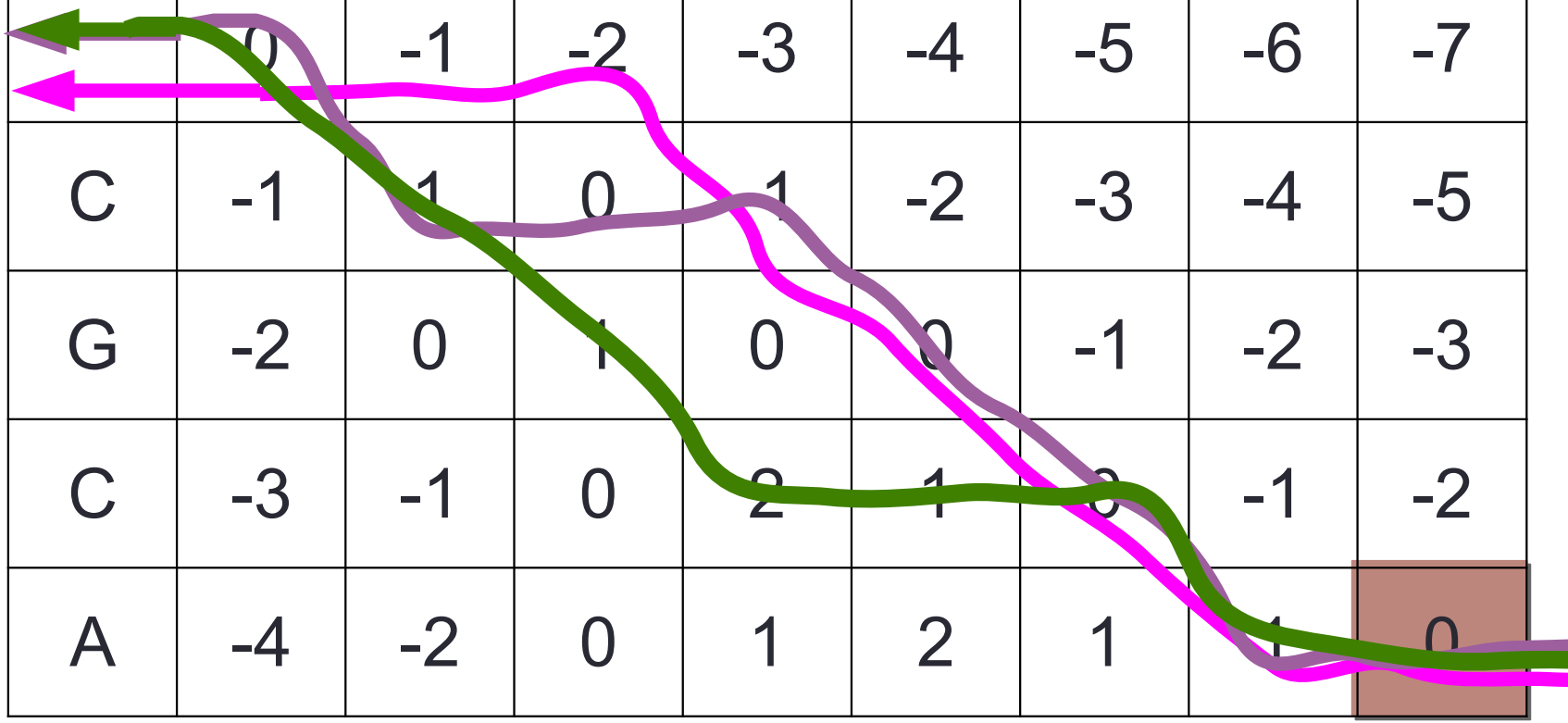
CACGTAT

CGC--A-

Example Alignment score = 0

Let:
Match = +1
Mismatch = 0
Gap = -1

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



```
CACGTAT
--CGCA-

CACGTAT
C--GCA-

CACGTAT
CGC--A-
```