

Introduction to sequence alignment

Andrey Prjibelski
Center for Algorithmic Biotechnology
SPbU

Alignment

AACGCTAACGGTAA
AACCGCGAACTAA

Alignment

AACGCTAACGGTAA

AACCGCGAACTAA



AAC - GCTAACGGTAA

AACCGCGAAC - - TAA

Needleman–Wunsch algorithm

ACGTTAG

ACCTAG

Needleman–Wunsch algorithm

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

$S(x, x) = 1$ (match)

$S(x, y) = -1$ (mismatch)

$S(x, -) = S(-, x) = -1$ (Indel)

$\text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = 3$

Dynamic programming

Solving a complex problem by breaking it into collection of simpler problems and storing intermediate results in specific data structures

Dynamic programming

A C G T T A G

A C C T A G

Dynamic programming

Case 1

A C G T T A G

A C C T A G

Dynamic programming

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

$$\text{Score}(\text{ACGTTA}, \text{ACCTA}) = 2$$

Dynamic programming

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

$$\begin{aligned} & \text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = \\ & \text{Score}(\text{ACGTTA}, \text{ACCTA}) + S(\text{G}, \text{G}) = 2 + 1 = 3 \end{aligned}$$

Dynamic programming

Case 2

A C G T T A G

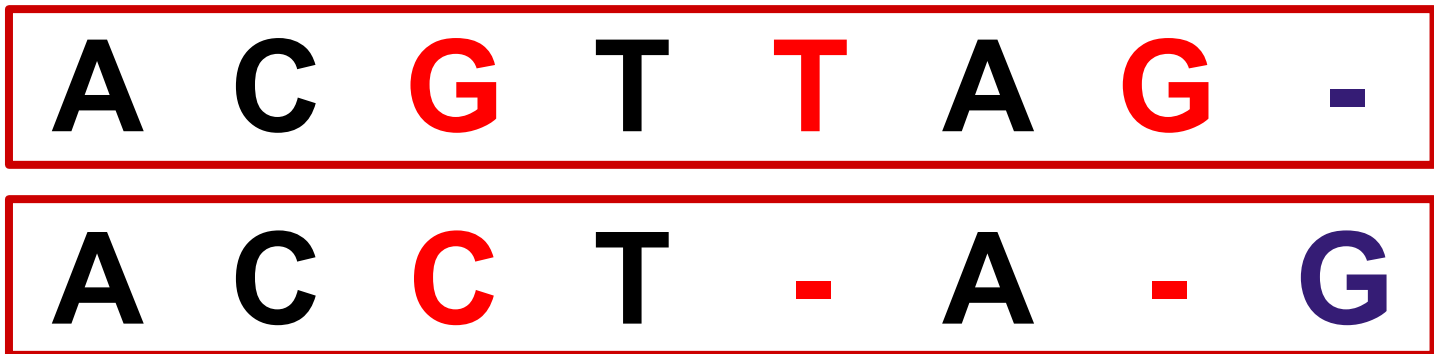
A C C T A G

Dynamic programming

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

Score(ACGTTAG, ACCTA) = 1

Dynamic programming



$$\begin{aligned} & \text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = \\ & \text{Score}(\text{ACGTTAG}, \text{ACCTA}) + S(-, \text{G}) = 1 + (-1) = 0 \end{aligned}$$

Dynamic programming

Case 3

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

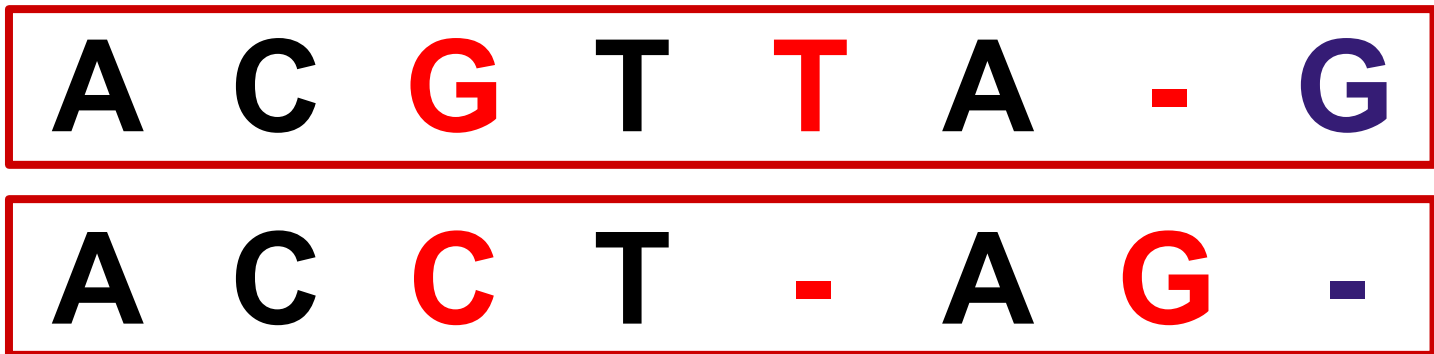
A C C T A G

Dynamic programming

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

$$\text{Score}(\text{ACGTTA}, \text{ACCTAG}) = 1$$

Dynamic programming



$$\begin{aligned} & \text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = \\ & \text{Score}(\text{ACGTTA}, \text{ACCTAG}) + S(\text{G}, \text{-}) = 1 + -1 = 0 \end{aligned}$$

Dynamic programming

- Case 1

$$\underline{\text{Score(ACGTTAG, ACCTAG) = 3}}$$

- Case 2

$$\text{Score(ACGTTAG, ACCTAG) = 0}$$

- Case 3

$$\text{Score(ACGTTAG, ACCTAG) = 0}$$

Dynamic programming

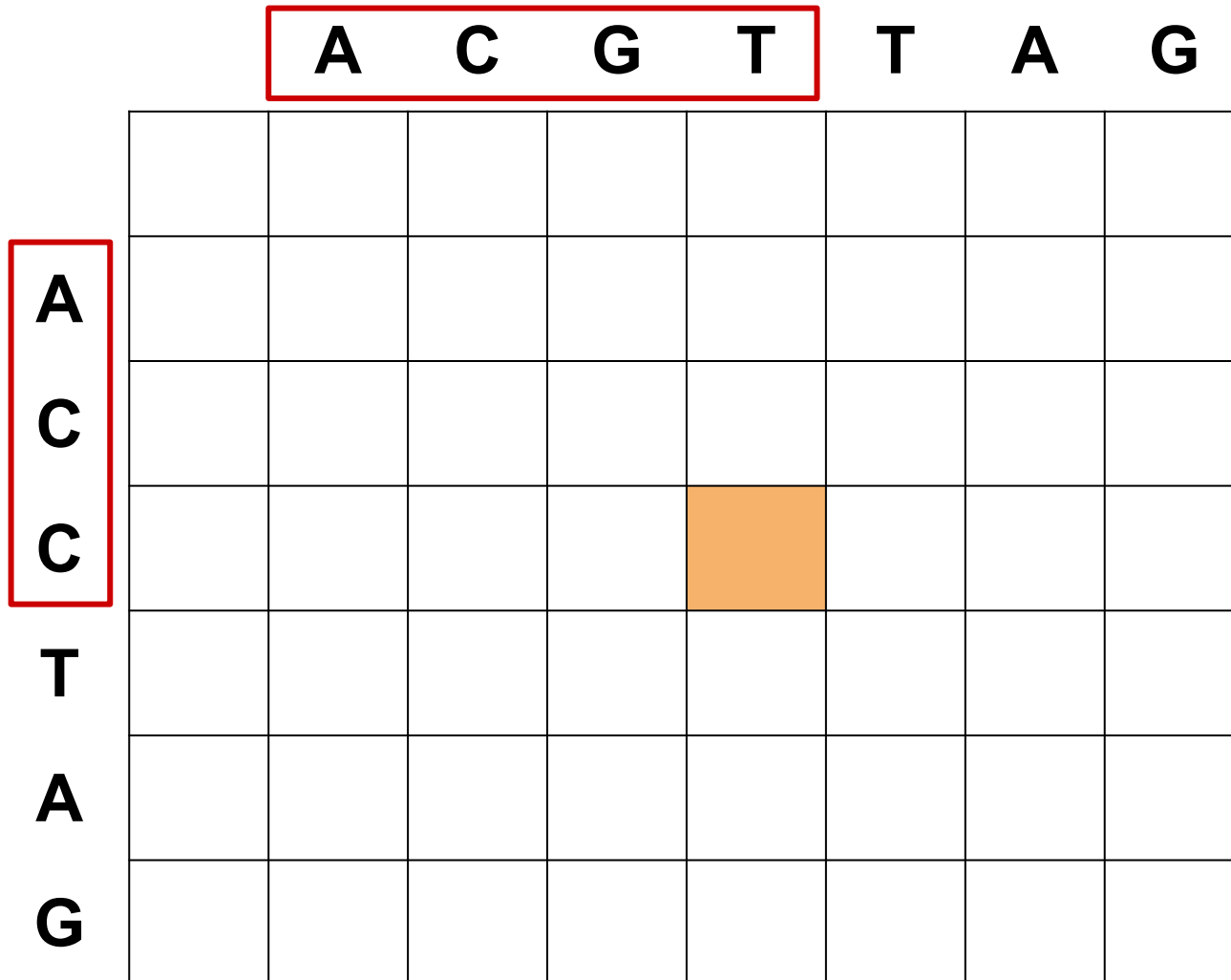
A	C	G	T	T	A	G
A	C	C	T	-	A	G

$$\begin{aligned} & \text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = \\ & \text{Score}(\text{ACGTTA}, \text{ACCTA}) + S(\text{G}, \text{G}) = 2 + 1 = 3 \end{aligned}$$

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

	A	C	G	T	T	A	G
0							
A							
C							
C							
T							
A							
G							

Needleman–Wunsch algorithm

		A	C	G	T	T	A	G
	0	-1						
A								
C								
C								
T								
A								
G								

Needleman–Wunsch algorithm

		A	C	G	T	T	A	G
	0	-1	-2					
A								
C								
C								
T								
A								
G								

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

- - - -
A C C T

Score(ACGT, "") = -4

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

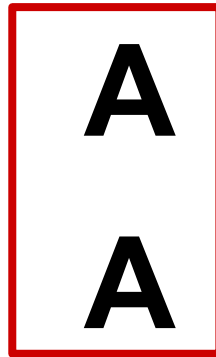
Needleman–Wunsch algorithm



A diagram illustrating a step in the Needleman–Wunsch algorithm. It features a vertical red rectangle on the left, representing a gap in a sequence. To the right of the rectangle, the letter 'A' is written twice, stacked vertically, representing the characters in the other sequence.

$$\text{Score}(\text{“ ”}, \text{“ ”}) = 0$$

Needleman–Wunsch algorithm



A

A

$$\text{Score}(A, A) = \text{Score}("", "") + S(A, A) = 0 + 1 = 1$$

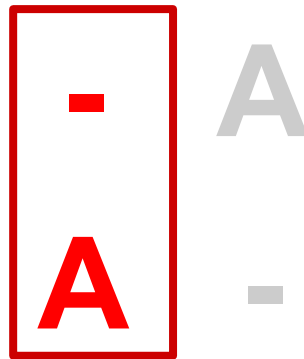
Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

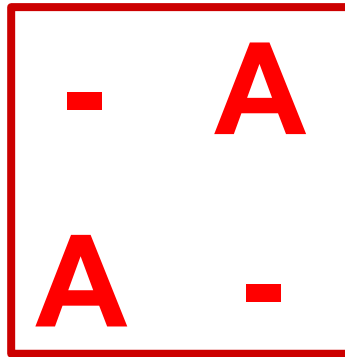
		A	C	G	T	T	A	G
	0	-1	-2	-3	-4	-5	-6	-7
A	-1	1						
C	-2							
C	-3							
T	-4							
A	-5							
G	-6							

Needleman–Wunsch algorithm



$$\text{Score}(\text{" "}, A) = -1$$

Needleman–Wunsch algorithm



$$\text{Score}(A, A) = \text{Score}(\text{“ ”}, A) + S(A, -) = -1 + -1 = -2$$

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

$$M(i, j) = \text{MAX} \begin{cases} M(i - 1, j - 1) + S(X(i), Y(j)) \\ M(i - 1, j) + S(X(i), '-') \\ M(i, j - 1) + S('-', Y(j)) \end{cases}$$

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

The image shows a Needleman–Wunsch algorithm matrix. The columns are labeled A, C, G, T, T, A, G. The rows are labeled A, C, C, T, A, G. The matrix contains numerical values representing the alignment score. Red arrows indicate the path of the optimal alignment, starting from the bottom-left cell (row G, column A) and moving up and right to the top-right cell (row A, column G). The cells (row A, column T) and (row A, column A) are highlighted in green, and the cell (row A, column A) is highlighted in orange.

Needleman–Wunsch algorithm

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

The diagram illustrates the Needleman–Wunsch algorithm for sequence alignment. The table shows the scores for aligning the sequence 'ACCTAAG' (rows) with the sequence 'ACCTAAG' (columns). The scores are calculated based on the alignment of characters, with a match score of 1 and a mismatch score of -1. The cell (1,2) is highlighted in orange, indicating a match between 'C' and 'C'. Red arrows show the path of maximum alignment, starting from (0,0) and ending at (1,7).

Needleman–Wunsch algorithm

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

Diagram illustrating the Needleman–Wunsch algorithm. The table shows the alignment of two sequences: A (top row) and C (left column). The sequence A is A C G T T A G, and the sequence C is C C T A G. The table contains numerical values representing the alignment score. The cell containing '1' (A aligned with A) is highlighted in green. The cell containing '2' (C aligned with C) is highlighted in orange. Red arrows indicate the path of the alignment, starting from the top-left cell (0) and moving through the cells containing '1', '0', and '2'.

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

The image shows a dynamic programming table for the Needleman–Wunsch algorithm. The table is a grid with 8 columns and 8 rows. The columns are labeled A, C, G, T, T, A, G and the rows are labeled A, C, C, T, A, G. The top-left cell (0,0) is 0. The first row contains values from -1 to -7. The first column contains values from -1 to -6. The cell at row A, column C (1,2) is highlighted in green and contains 0. The cell at row C, column C (2,3) is highlighted in orange and contains 2/-1/-1. Red arrows point from the green cell to its left, top-left, and top-right neighbors, and from the orange cell to its top neighbor.

Needleman–Wunsch algorithm

$$M(i, j) = \text{MAX} \begin{cases} M(i - 1, j - 1) + S(X(i), Y(j)) \\ M(i - 1, j) + S(X(i), '-') \\ M(i, j - 1) + S('-', Y(j)) \end{cases}$$

Needleman–Wunsch algorithm

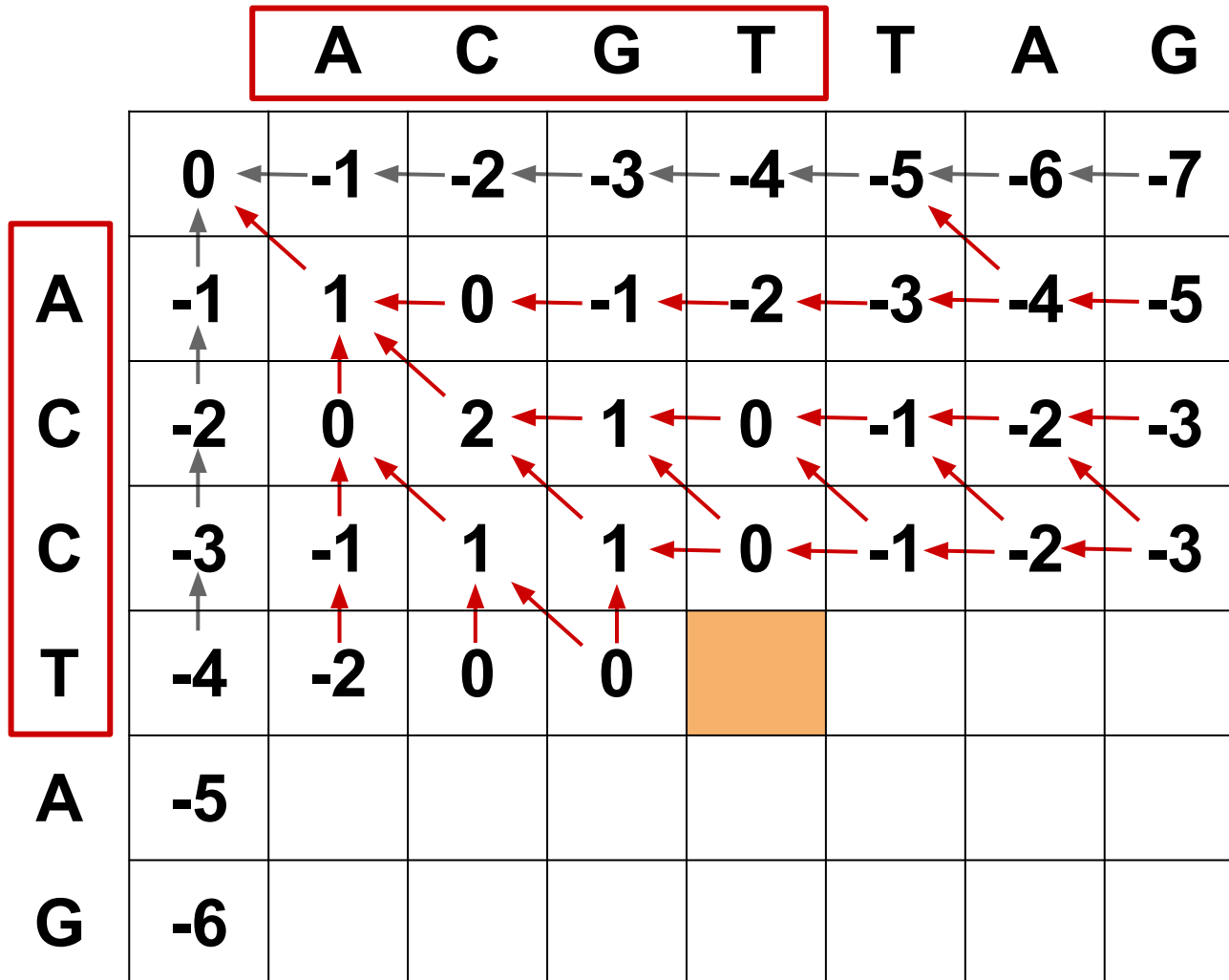
	A	C	G	T	T	A	G	
	0	-1	-2	-3	-4	-5	-6	-7
A	-1	1	0	-1	-2	-3	-4	-5
C	-2	0	2					
C	-3							
T	-4							
A	-5							
G	-6							

Needleman–Wunsch algorithm

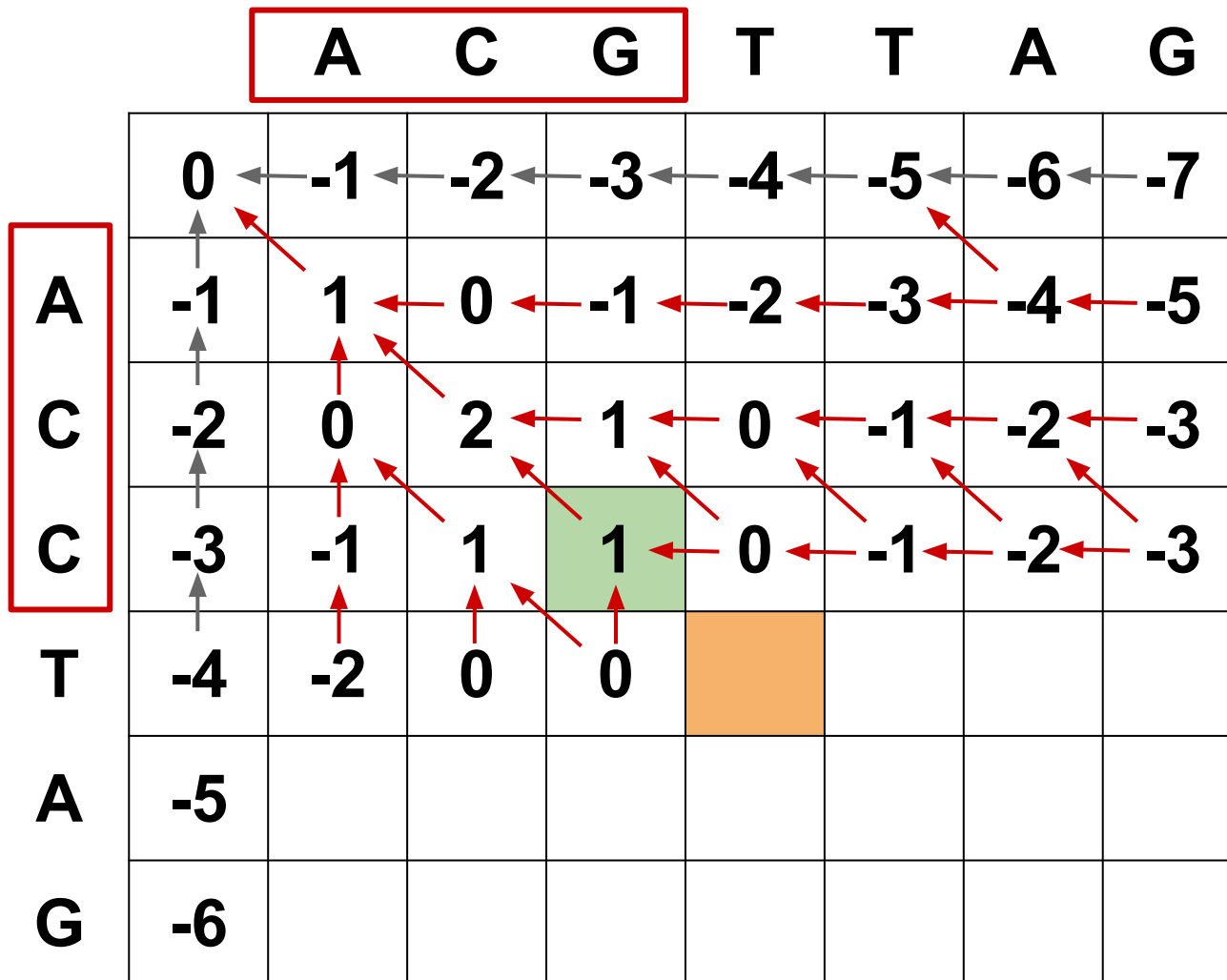
	A	C	G	T	T	A	G	
	0	-1	-2	-3	-4	-5	-6	-7
A	-1	1	0	-1	-2	-3	-4	-5
C	-2	0	2	1	0	-1	-2	-3
C	-3	-1	1	1	0	-1	-2	-3
T	-4	-2	0	0				
A	-5							
G	-6							

The diagram illustrates the Needleman–Wunsch algorithm for sequence alignment. It shows a dynamic programming matrix for aligning the sequence 'ACCTAA' (rows) with 'ACCTAA' (columns). The matrix contains scores for each cell, and red arrows indicate the path of maximum score from the bottom-right cell (row 'G', column 'G') to the top-left cell (row 'A', column 'A'). The path starts at (G,G) with a score of -6, moves up to (A,G) with a score of -5, then left to (A,A) with a score of -1, and finally up to (A,A) with a score of 1. The path ends at (A,A) with a score of 1.

Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

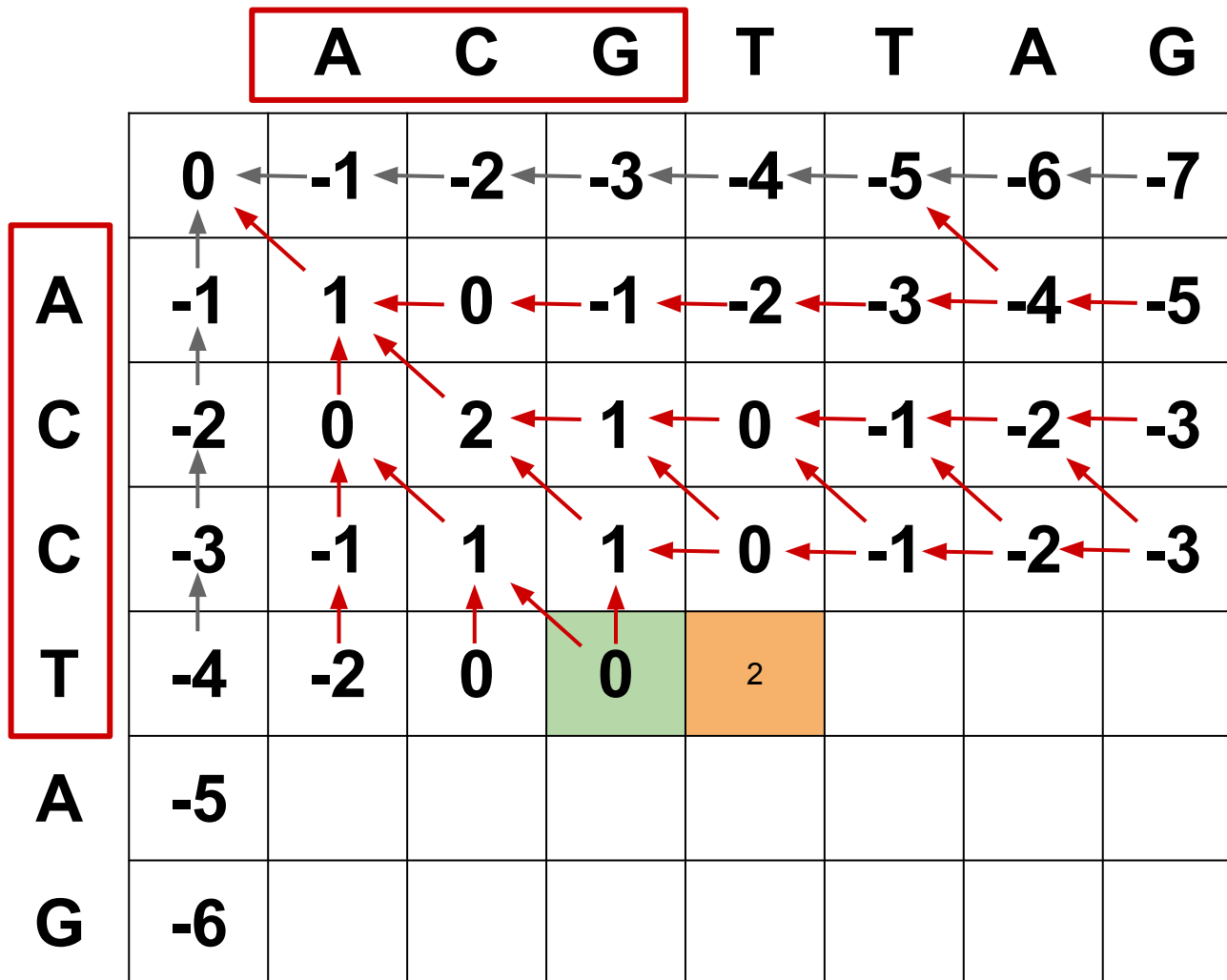


Dynamic programming

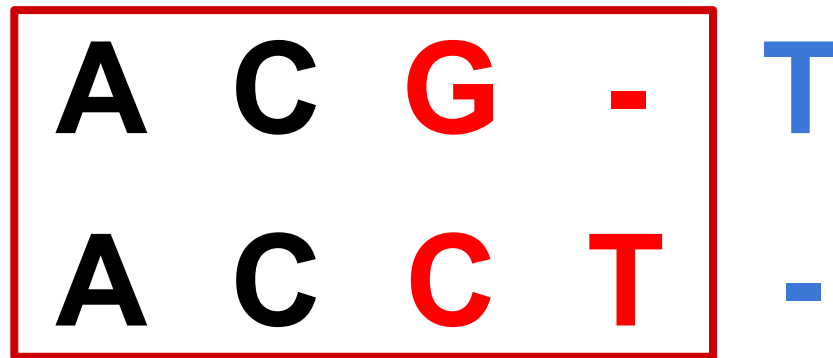
A	C	G	T
A	C	C	T

$$\begin{aligned} \text{Score(ACGT, ACCT)} &= \\ \text{Score(ACG, ACC)} + S(\text{T, T}) &= 1 + 1 = 2 \end{aligned}$$

Needleman–Wunsch algorithm

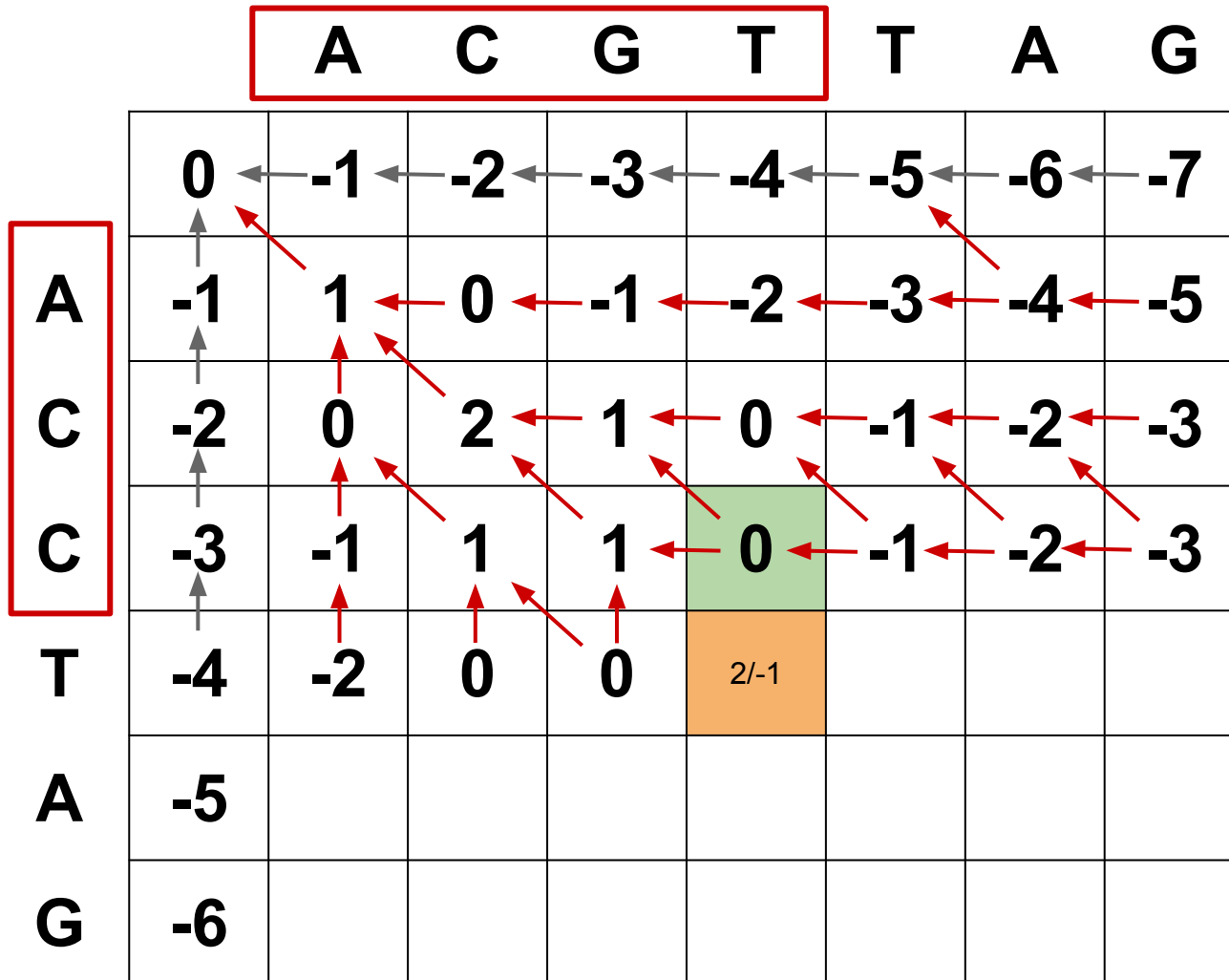


Dynamic programming



$$\begin{aligned} \text{Score(ACGT, ACCT)} &= \\ \text{Score(ACG, ACCT)} + S(\text{T, -}) &= 0 + -1 = -1 \end{aligned}$$

Needleman–Wunsch algorithm



Dynamic programming

A	C	G	T	-
A	C	C	-	T

$$\begin{aligned} \text{Score}(\text{ACGT}, \text{ACCT}) &= \\ \text{Score}(\text{ACGT}, \text{ACC}) + S(-, \text{T}) &= 0 + -1 = -1 \end{aligned}$$

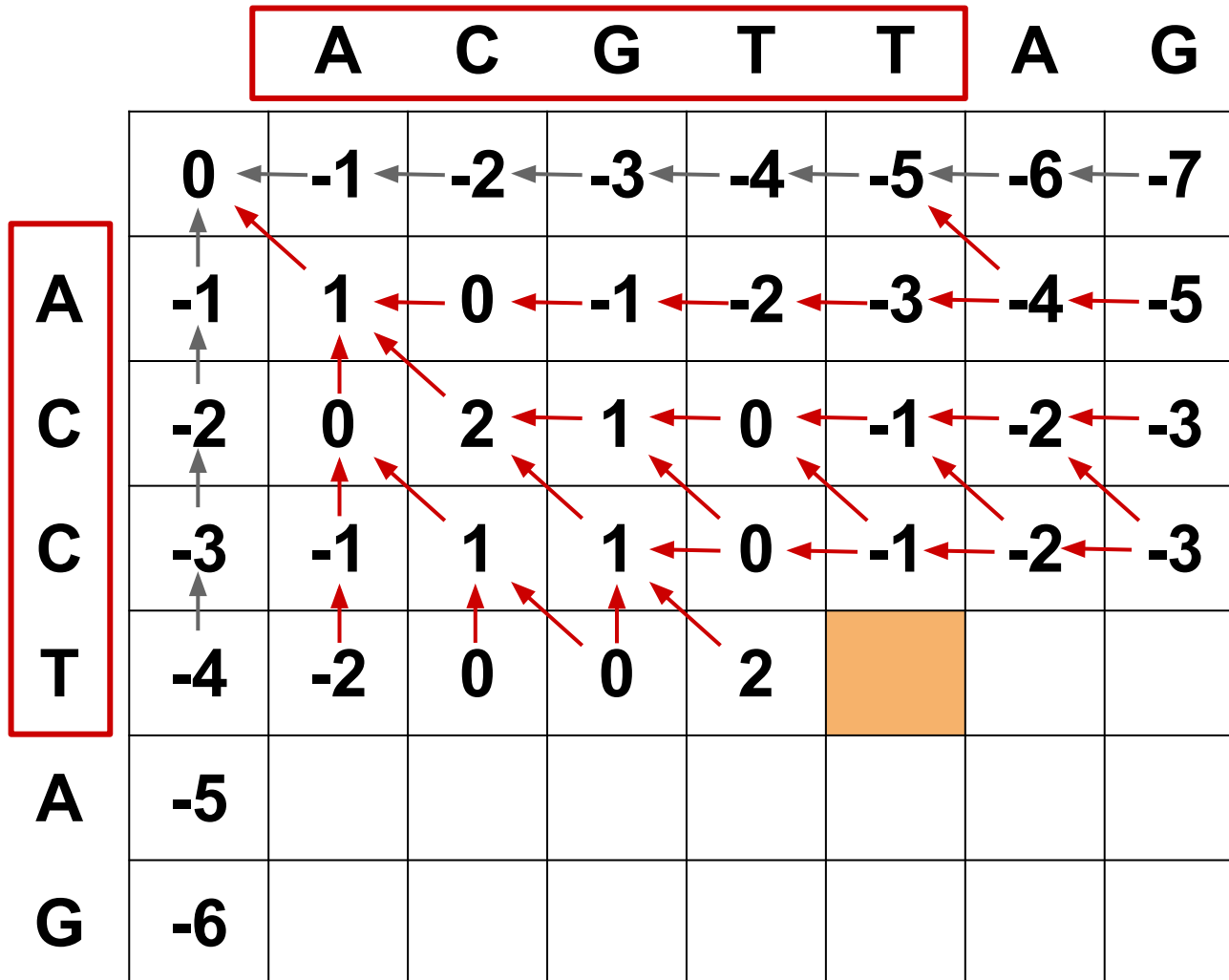
Needleman–Wunsch algorithm

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

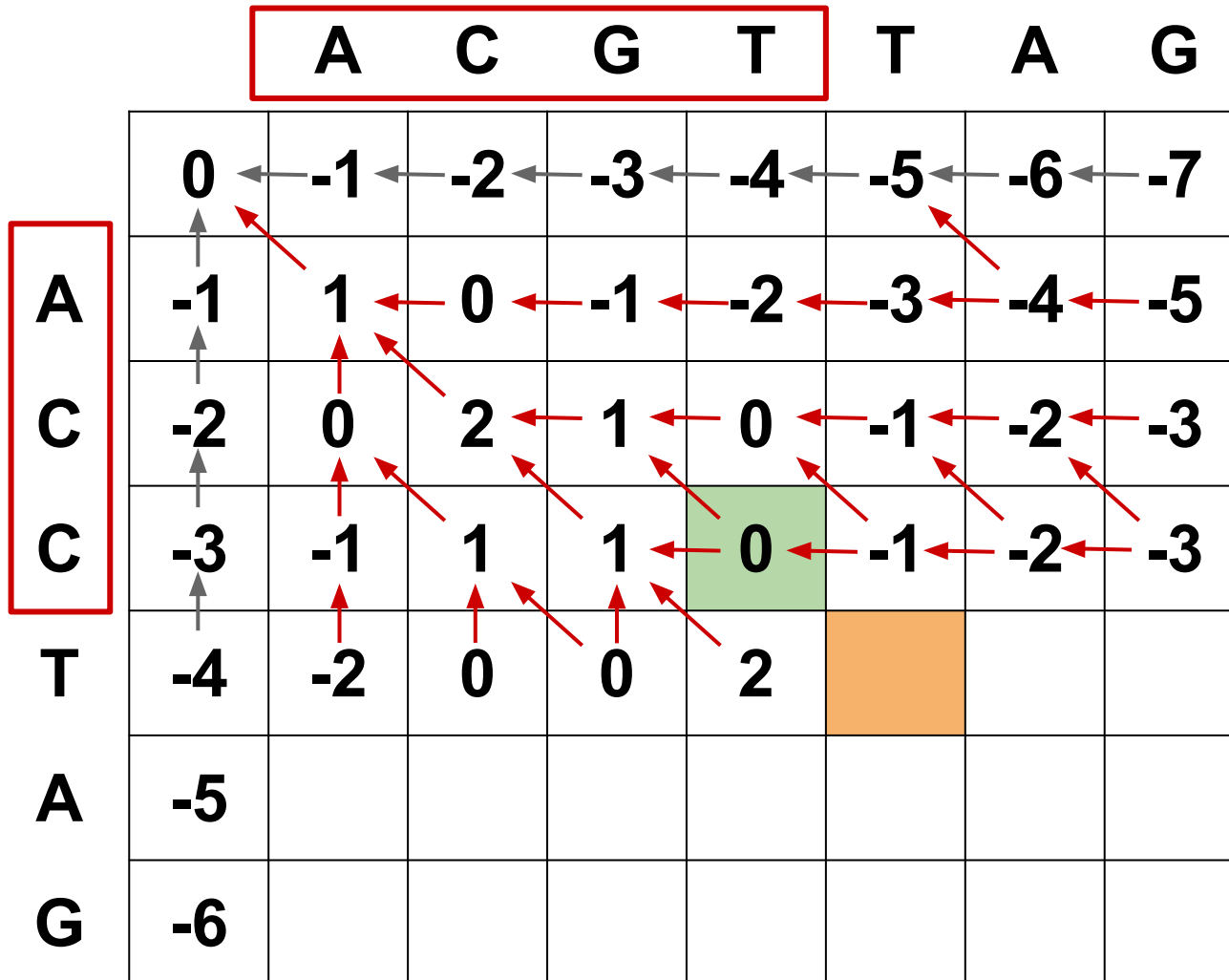
Needleman–Wunsch algorithm

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

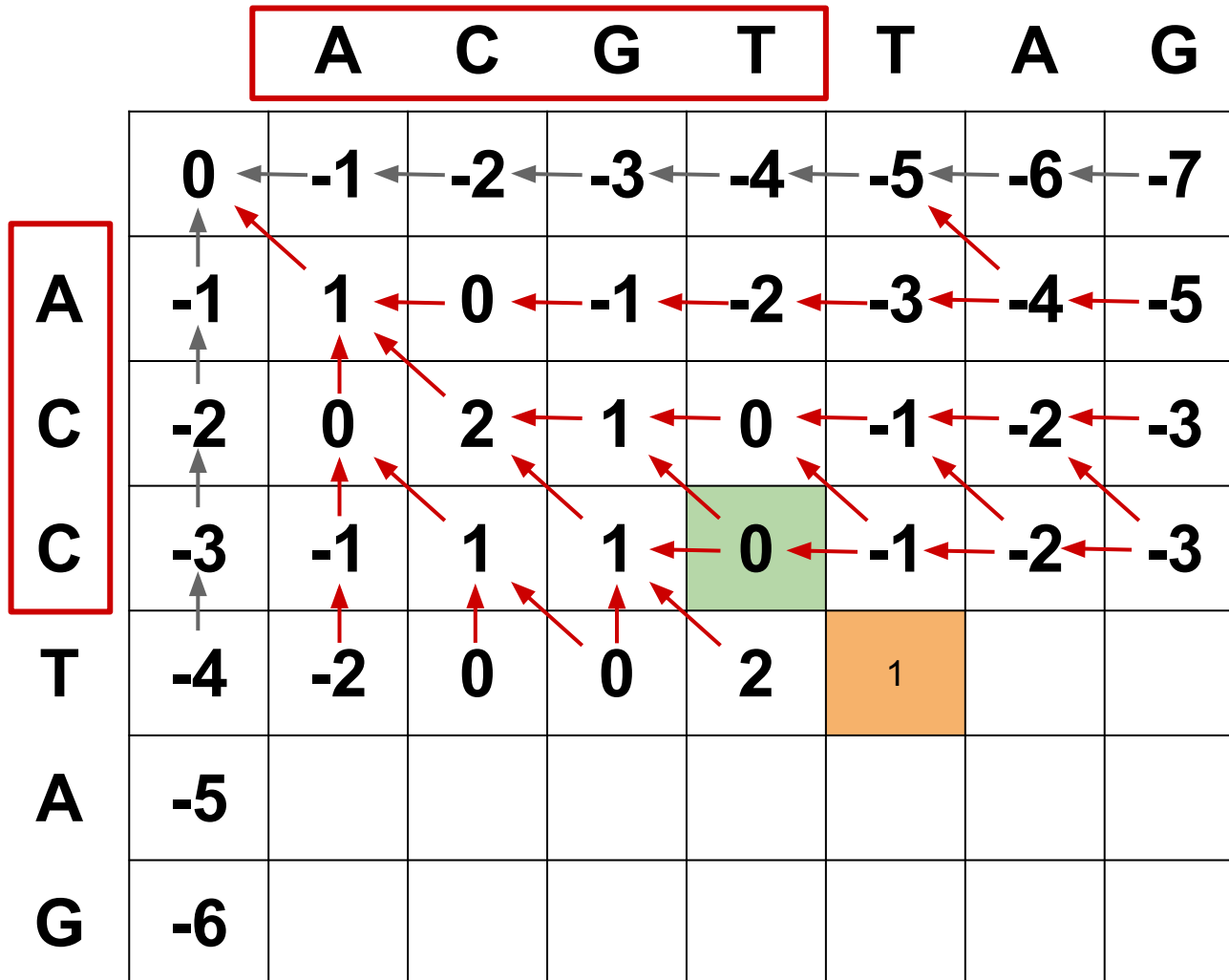
Needleman–Wunsch algorithm



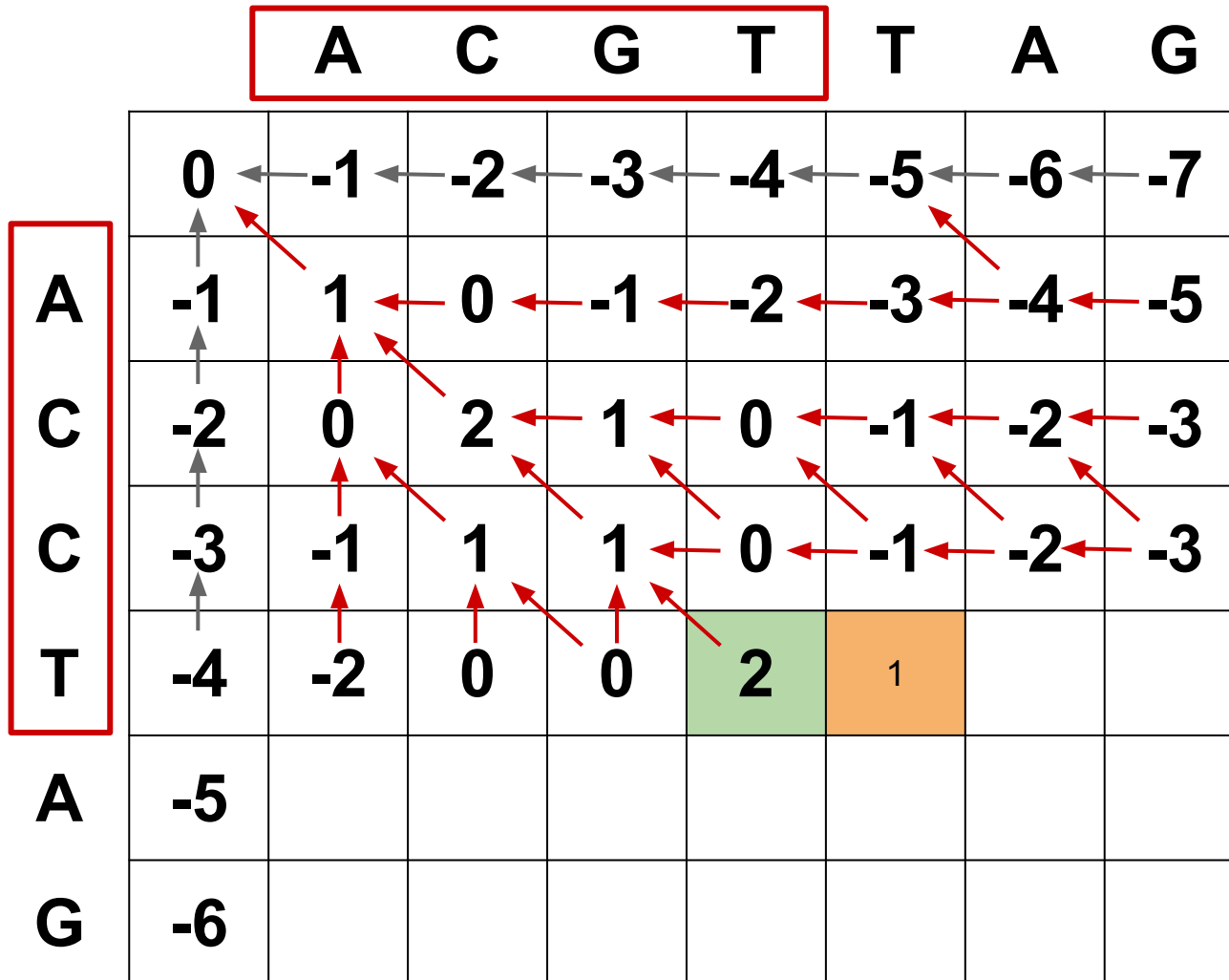
Needleman–Wunsch algorithm



Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

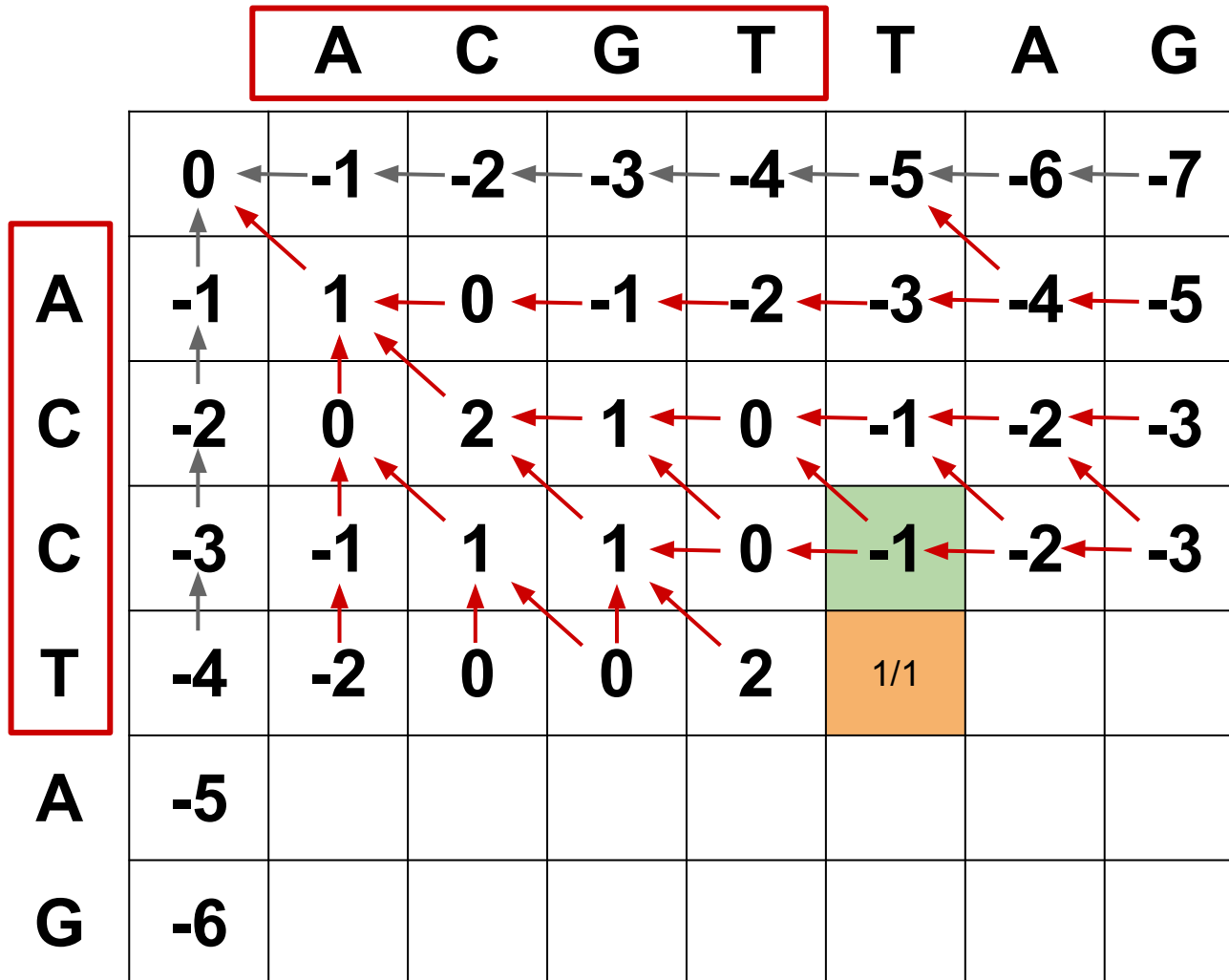


Needleman–Wunsch algorithm

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

The image shows a Needleman–Wunsch algorithm matrix for sequence alignment. The top sequence is A C G T T A G and the bottom sequence is A C C T A G. The matrix contains numerical values representing the alignment score at each position. Red arrows indicate the path of the optimal alignment, starting from the bottom-right cell (T, T) and moving towards the top-left cell (A, A). The cell containing '1/1' is highlighted in orange, and the cell containing '2' is highlighted in green.

Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

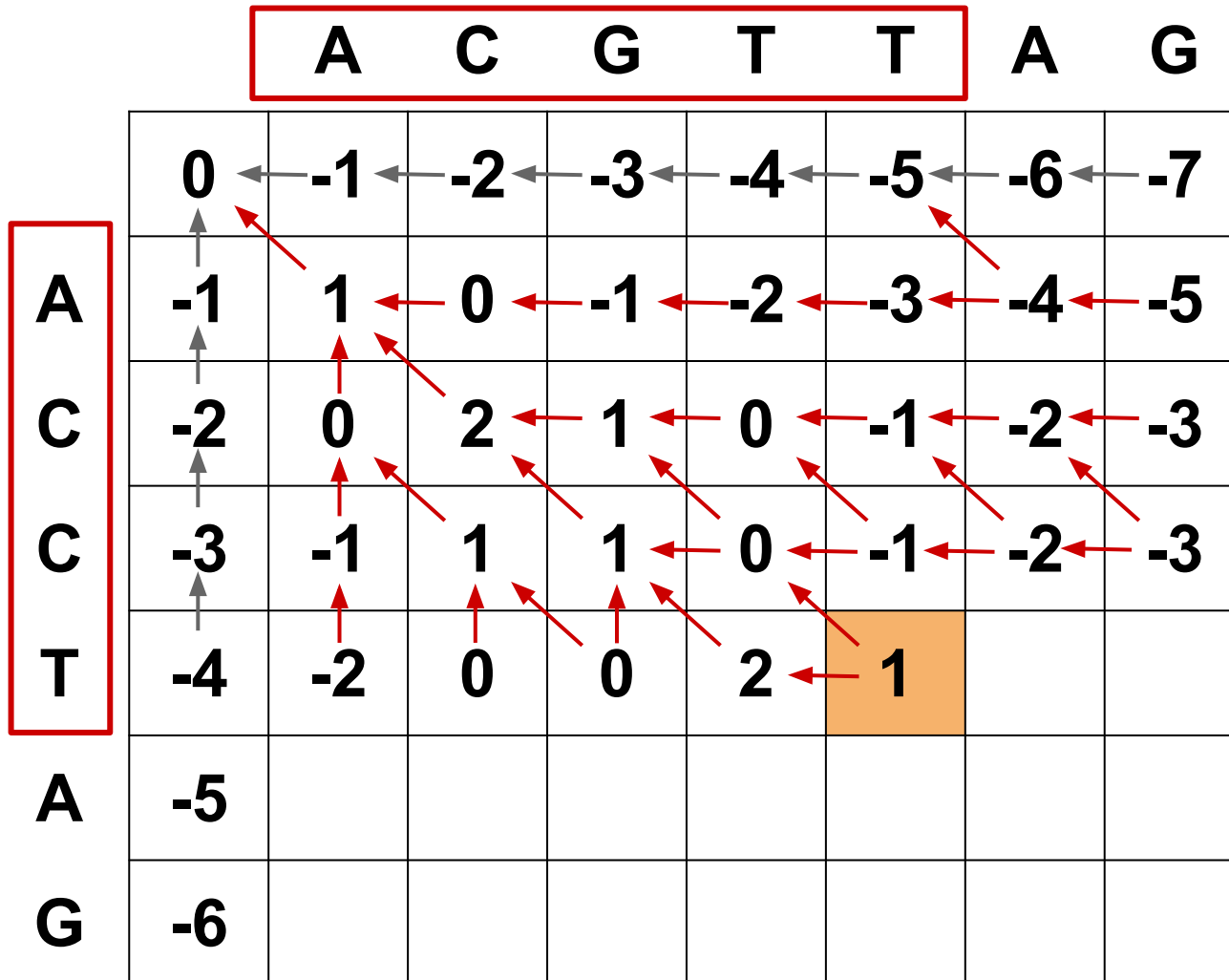
	A C G T				T	A	G	
	0	-1	-2	-3	-4	-5	-6	-7
A	-1	1	0	-1	-2	-3	-4	-5
C	-2	0	2	1	0	-1	-2	-3
C	-3	-1	1	1	0	-1	-2	-3
T	-4	-2	0	0	2	1/1/-2		
A	-5							
G	-6							

Needleman–Wunsch algorithm

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

The diagram illustrates the Needleman–Wunsch algorithm for sequence alignment. The top sequence is A C G T T A G and the bottom sequence is A C C T A G. The alignment path is highlighted with red arrows, starting from the bottom-right cell (T, T) and moving towards the top-left cell (A, A). The value 1/1/-2 in the cell (T, T) indicates a match with a score of 1, a gap penalty of 1, and a mismatch penalty of -2.

Needleman–Wunsch algorithm



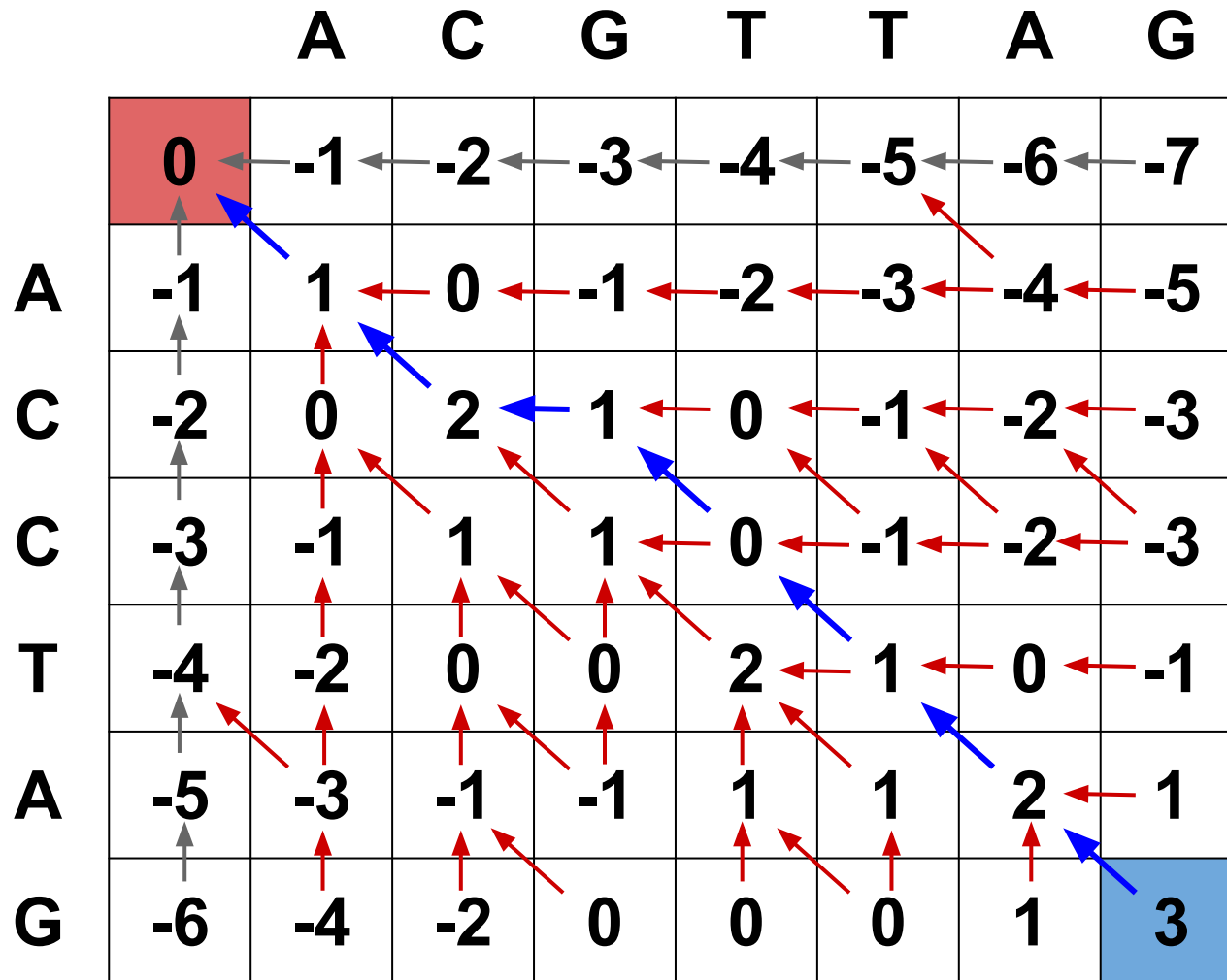
Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm

G	G

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

Needleman–Wunsch algorithm

A	A
G	G

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

Needleman–Wunsch algorithm

T	T
A	A
G	G

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

Needleman–Wunsch algorithm

C	T
T	T
A	A
G	G

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

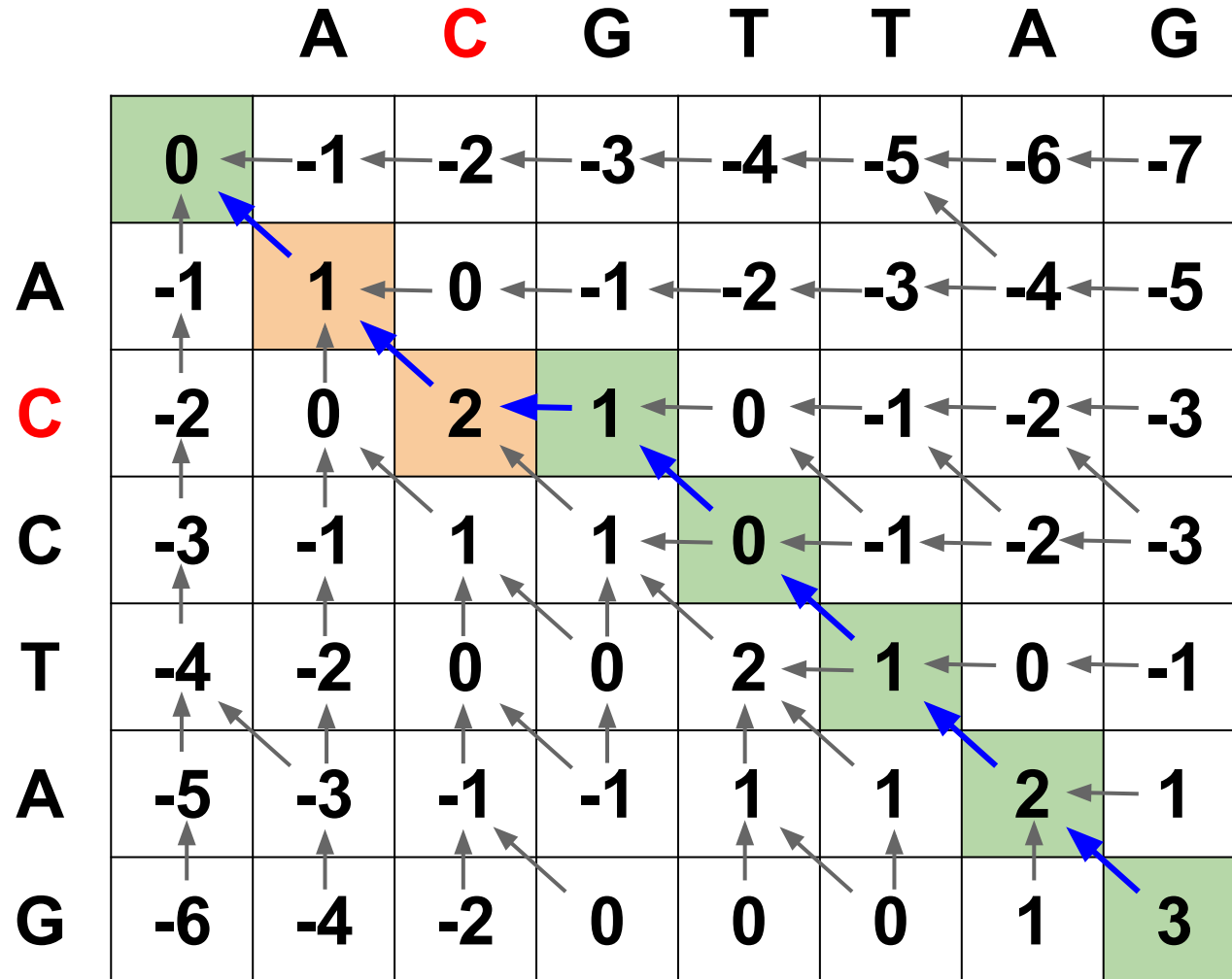
Needleman–Wunsch algorithm

-	G
C	T
T	T
A	A
G	G

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

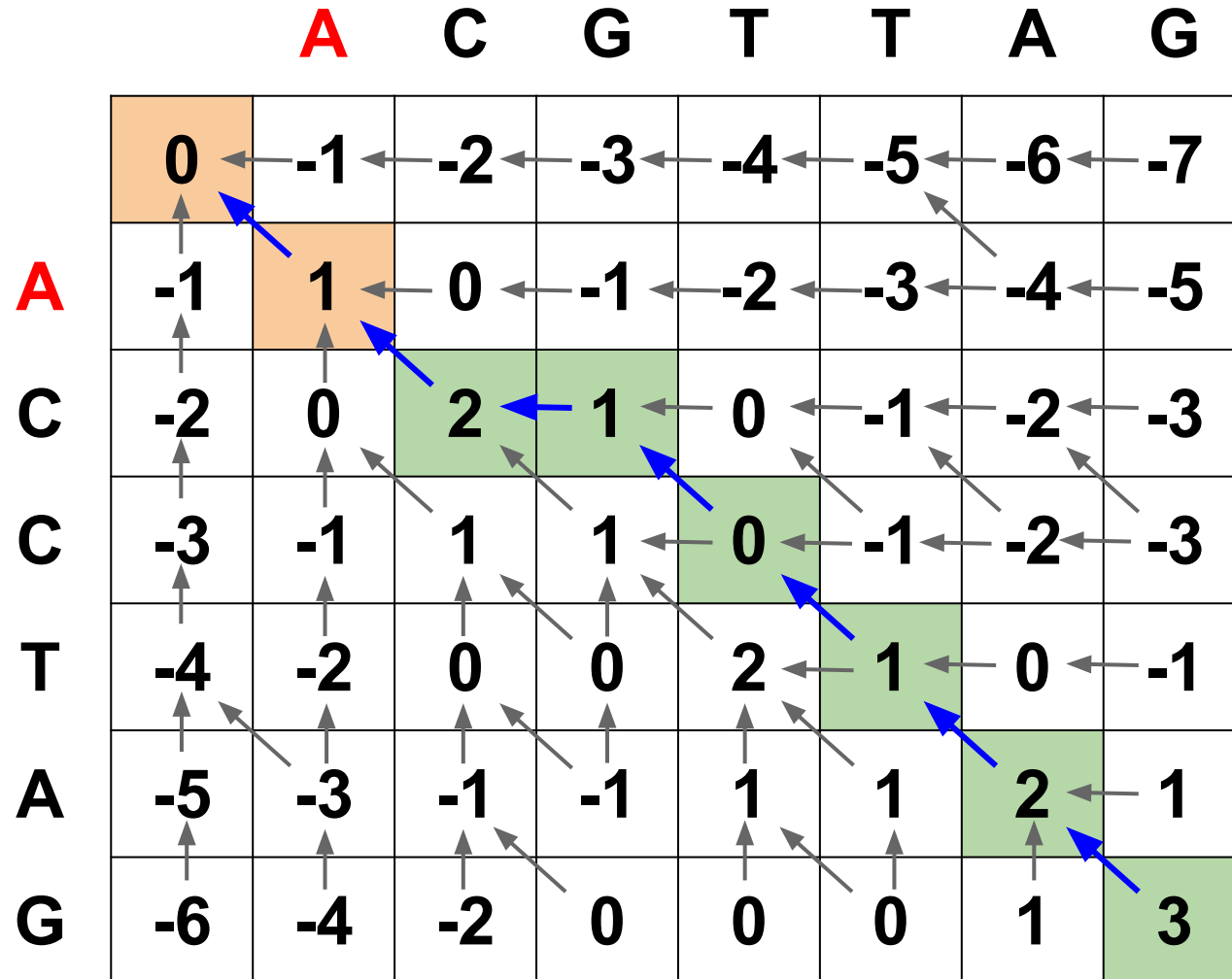
Needleman–Wunsch algorithm

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



Needleman–Wunsch algorithm

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



Needleman–Wunsch algorithm

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

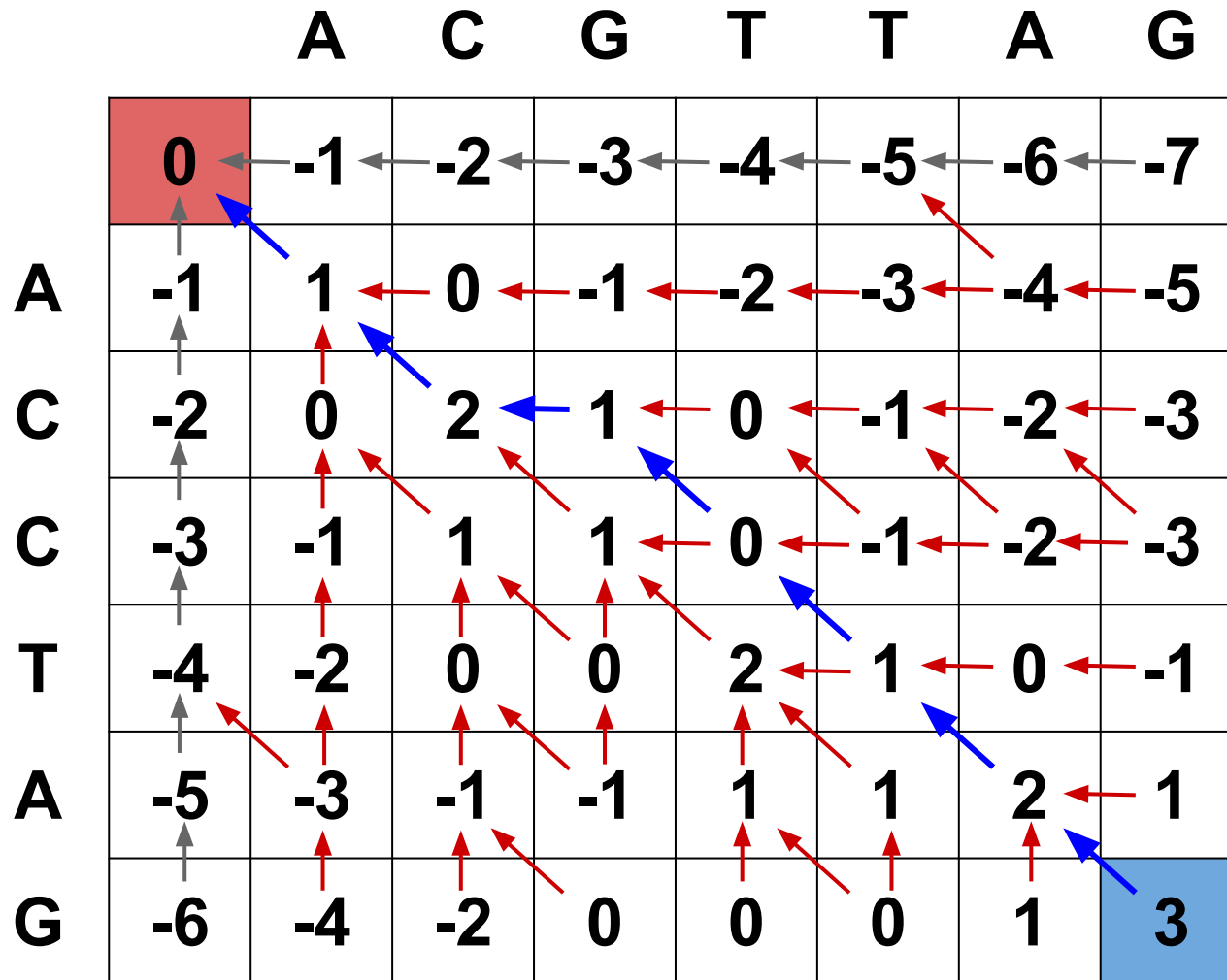
A C **C** T - A G

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

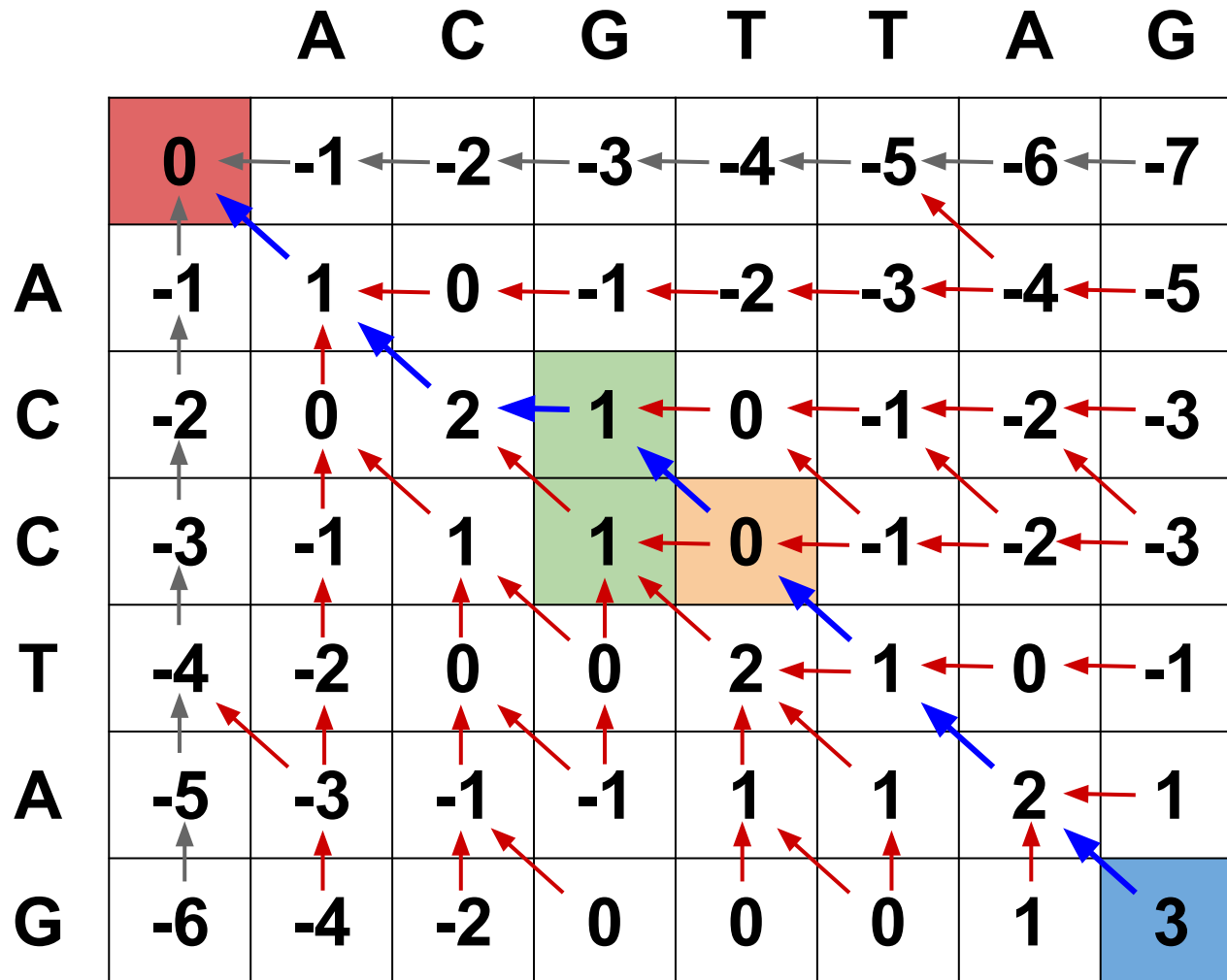
A C - **C** T A G

$$\text{Score}(\text{ACGTTAG}, \text{ACCTAG}) = 3$$

Needleman–Wunsch algorithm



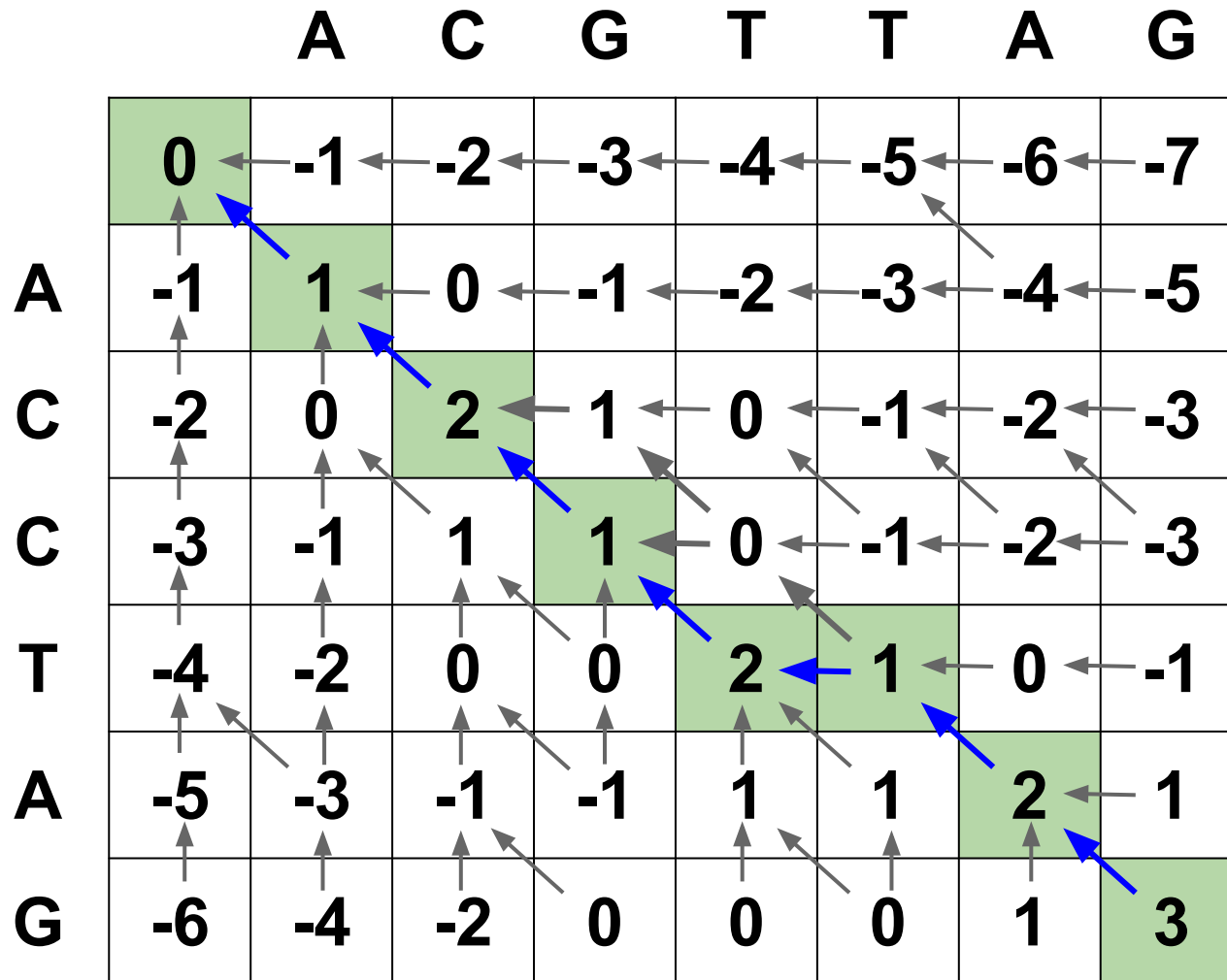
Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

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

Needleman–Wunsch algorithm



Needleman–Wunsch algorithm

Time =

Memory =

Needleman–Wunsch algorithm

$$\text{Time} = O(n^2)$$

$$\text{Memory} = O(n^2)$$

Problem solving

stepik.org/56393/

Problem 0

Parse FASTA format from stdin

>Header_1

ACGGCTTGACGGT

>Header_2

ATACAATCCGCGAACTAAGCGGGA

....

Report total number of sequences and bases to stdout

Hint: use BioPython `from Bio import SeqIO`

Problem 1a

Implement Needleman-Wunsch algorithm

- Input: FASTA with 2 sequences
- Output: alignment score

Problem 1b

Implement Needleman-Wunsch algorithm

- Input: FASTA with 2 sequences
- Output: alignment score and alignment itself

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

Problem 1b*

Implement Needleman-Wunsch algorithm

- Input: FASTA with 2 sequences
- Output: alignment score and alignment itself;
if multiple alignments exist, output all

Global alignment

ATCTGATCGCCA

ATACAATCCGCGAACTAAGCGGGA



AT- C - -T - - G - - A - - T - - - CGCCA

ATACAATCCGCGAACTAAGCGGGA

Local alignment

ATCTGATCGCCA

ATACAATCCGCGAACTAAGCGGGA



AT **CTGAT**CGCCA

ATACAATCCGC-

GAACTAAGCGGGA

Problem 2

Implement Smith-Waterman algorithm

- Input: FASTA with 2 sequences
- Output: alignment score and alignment itself

Smith-Waterman algorithm

0	0	0	0	0	0	0	0
0							
0							
0							
0							
0							
0							

Scoring matrix

In real life different substitutions may have different probability

	A	C	G	T
A	3	-3	-2	-3
C	-3	3	-3	-2
G	-2	-3	3	-3
T	-3	-2	-3	3

Problem 3

Implement SW or NW algorithm with scoring matrix

- Input: Scoring matrix (4 rows with 4 numbers each) and FASTA with 2 sequences
- Output: alignment score and alignment itself

Affine gap penalty

ACGCCTTAACGTGCA
ACGCCTACCGGTGCA

Affine gap penalty

ACGCCTTAACCGGTGCA

ACGCTACGTGCA



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

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

Affine gap penalty

ACGCCTTAACCGGTGCA

ACGCTACGTGCA



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

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

Affine gap penalty

- Scoring function $S(a, b)$
- Gap open score S_{op}
- Gap extension score S_{ex}

Affine gap penalty

- 3 matrices
 - **M**atches/mismatches (diagonal arrows)
 - **I**nsertions (vertical arrows)
 - **D**eletions (horizontal arrows)
- Moving from **M** to **D** or **I** adds gap open penalty S_{op}

Affine gap penalty

- $I[i, j] = \max \{ I[i - 1, j] - S_{ex}, M[i - 1, j] - S_{op} \}$
- $D[i, j] = \max \{ D[i, j - 1] - S_{ex}, M[i, j - 1] - S_{op} \}$
- $M[i, j] = \max \{ D[i, j], I[i, j], M[i - 1, j - 1] + S(A[i], B[j]) \}$

Problem 4

Implement affine gap penalty algorithm

- Input: 1 line with 4 numbers (match, mismatch, gap open and gap extend scores) and FASTA with 2 sequences
- Output: alignment score and alignment itself

Problem 5

Given 2 long sequences ($L > 10^5$) that have less than k errors ($k \ll L$). Quickly compute

- a. Alignment score
- b. Alignment itself

If there are more than k errors, print “NULL”

- Input: integer k and FASTA with 2 sequences
- Output: alignment score and alignment itself