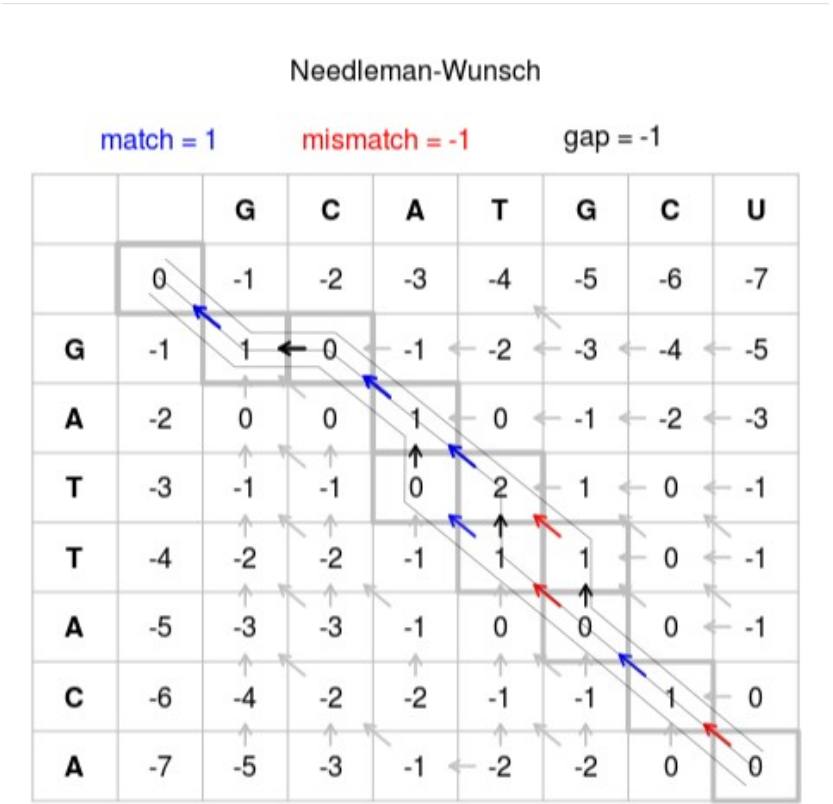
Propädeutikum: Programmierung in der Bioinformatik

Needleman-Wunsch

Thomas Mauermeier
21.01.2020
Ludwig-Maximilians-Universität München

Was war Needleman-Wunsch nochmal?



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

- Alignment-Algorithmus
  - globales Alignment (versucht jeden Nt mit jedem anderen Nt zu alignen)
- Siehe auch: Einführung in die Bioinfo, 9. Foliensatz, Folien 46-57

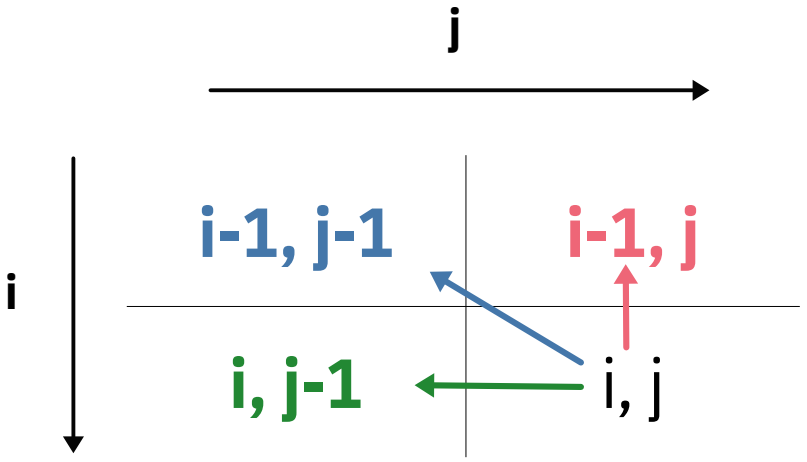
https://en.wikipedia.org/wiki/File:Needleman-Wunsch_pairwise_sequence_alignment.png

Schritt 0: Was brauchen wir?

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

- 2 Sequenzen seq1, seq2
- 2D-Matrizen der Größe seq1.length+1 × seq2.length+1
  - einmal für Score
  - (evtl. auch einmal für Traceback)
- Scores für:
  - Match (gleiche Zeichen)
  - Mismatch (verschiedene Zeichen)
  - Gap (für Indels, also Lücken)

Schritt 0: Was brauchen wir?



- Scoring-Funktion:
  Score an Position (i, j) wird das **Maximum** der Werte:
  - Wenn Zeichen **gleich**:
    matrix(i-1, j-1) + match
    Wenn Zeichen **ungleich**:
    matrix(i-1, j-1) + mismatch
  - **matrix(i-1, j) + gap**
  - **matrix(i, j-1) + gap**

Schritt 1: Initialisierung der Tabelle

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

- matrix(0, 0) auf 0 setzen
- Zeile 0, Spalte 0 = nur Gaps
  - daher für Zeile 0 immer der Fall: **matrix(i, j-1) + gap**
  - und für Spalte 0 immer der Fall: **matrix(i-1, j) + gap**

Scores: match = 1, mismatch = -1, gap = -1

Schritt 1: Initialisierung der Tabelle

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

- matrix(0, 0) auf 0 setzen
- Zeile 0, Spalte 0 = nur Gaps
  - daher für Zeile 0 immer der Fall: **matrix(i, j-1) + gap**
  - und für Spalte 0 immer der Fall: **matrix(i-1, j) + gap**

Scores: match = 1, mismatch = -1, **gap = -1**

Schritt 2: Füllen der Tabelle

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

- Zeile für Zeile durchlaufen und Score für (i, j) ermitteln

Scores: match = 1, mismatch = -1, gap = -1

Schritt 2: Füllen der Tabelle

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

- **matrix(i-1, j-1) + match**
  - **0 + 1 = 1**
- **matrix(i-1, j) + gap**
  - **-1 + -1 = -2**
- **matrix(i, j-1) + gap**
  - **-1 + -1 = -2**
- **max(1, -2, -2) = 1**
- **“Richtung” merken**

Scores: match = 1, mismatch = -1, gap = -1

Schritt 2: Füllen der Tabelle

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

- **matrix(i-1, j-1) + mismatch**
  - **-1 + -1 = -2**
- **matrix(i-1, j) + gap**
  - **-2 + -1 = -3**
- **matrix(i, j-1) + gap**
  - **1 + -1 = 0**
- **max(-2, -3, 0) = 0**
- **“Richtung” merken**

Scores: match = 1, mismatch = -1, gap = -1

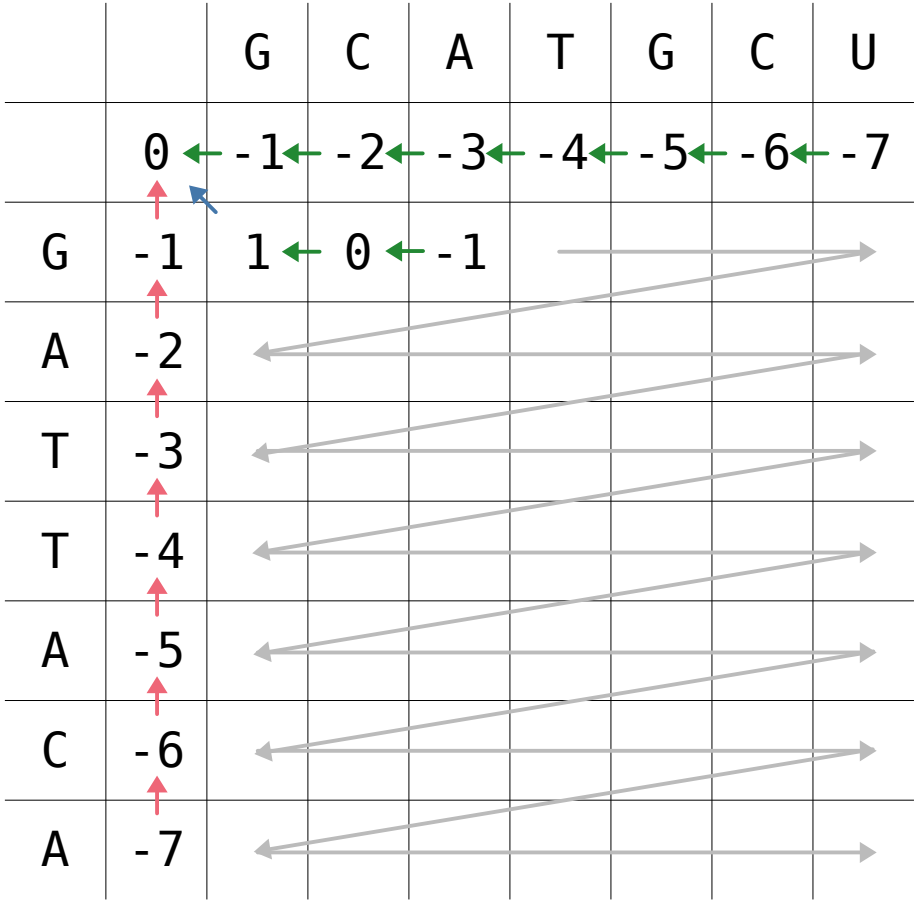
Schritt 2: Füllen der Tabelle

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

- **matrix(i-1, j-1) + mismatch**
  - **-2 + -1 = -3**
- **matrix(i-1, j) + gap**
  - **-3 + -1 = -4**
- **matrix(i, j-1) + gap**
  - **0 + -1 = -1**
- **max(-3, -3, -1) = -1**
- **“Richtung” merken**

Scores: match = 1, mismatch = -1, gap = -1

Schritt 2: Füllen der Tabelle



- Nach selbem Schema weiter bis Tabelle gefüllt ist

Schritt 3: Traceback (“Richtungen auswerten”)

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

- Schritt 2 komplett durchgelaufen
- Von ganz unten rechts beginnend Pfad nach oben finden und Alignment bauen
- **Achtung:**
  Zur Vereinfachung suchen wir nur *ein* optimales Alignment, d.h. es muss nur *ein* Pfad abgelaufen werden, wenn mehrere möglich sind!

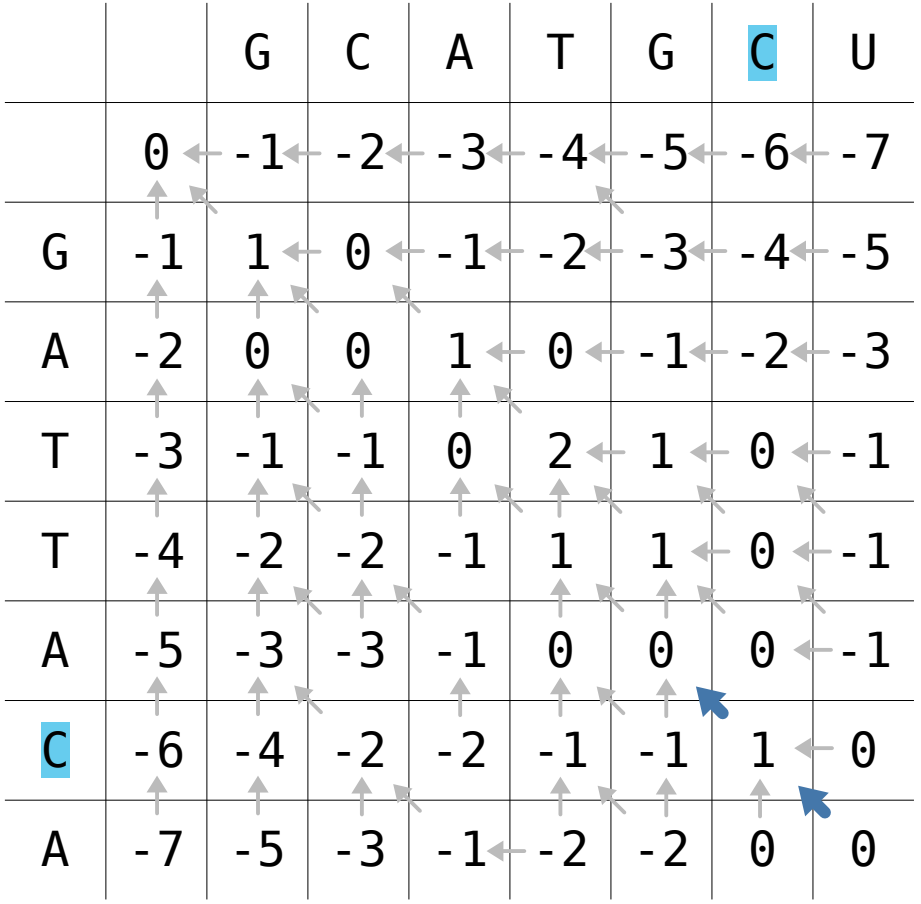
Schritt 3: Traceback (“Richtungen auswerten”)

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

- Diagonal = Mismatch:

  U
  A

Schritt 3: Traceback (“Richtungen auswerten”)



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

- Diagonal = Match:

  CU
  CA

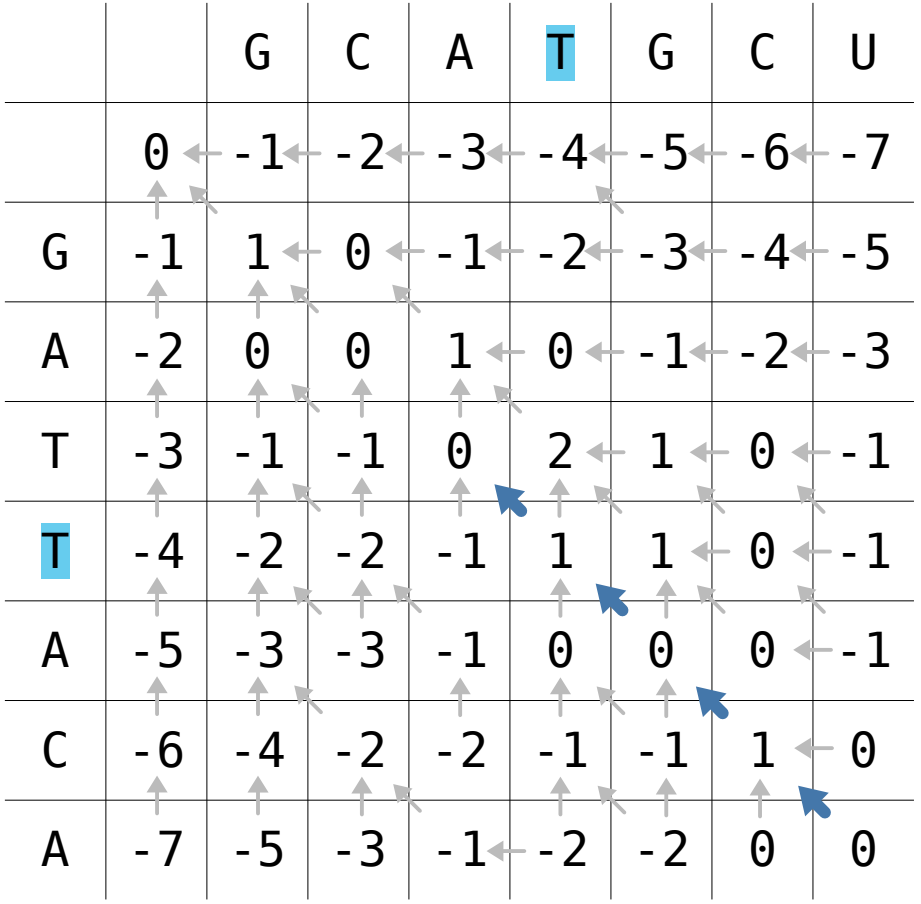
Schritt 3: Traceback (“Richtungen auswerten”)

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

- Diagonal = Mismatch:

  GCU
  ACA

Schritt 3: Traceback (“Richtungen auswerten”)



- Diagonal = Match:

  TGCU
  TACA

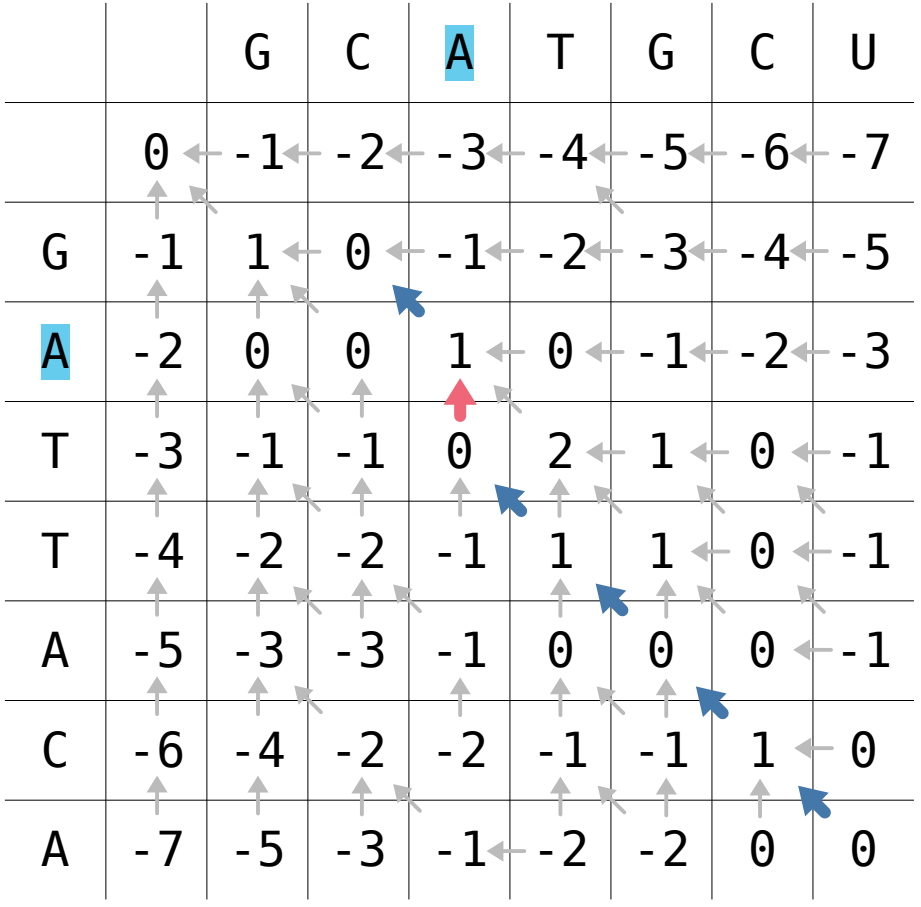
Schritt 3: Traceback (“Richtungen auswerten”)

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

- Hoch = Gap in Sequenz 1:

  -TGCU
  TTACA

Schritt 3: Traceback (“Richtungen auswerten”)

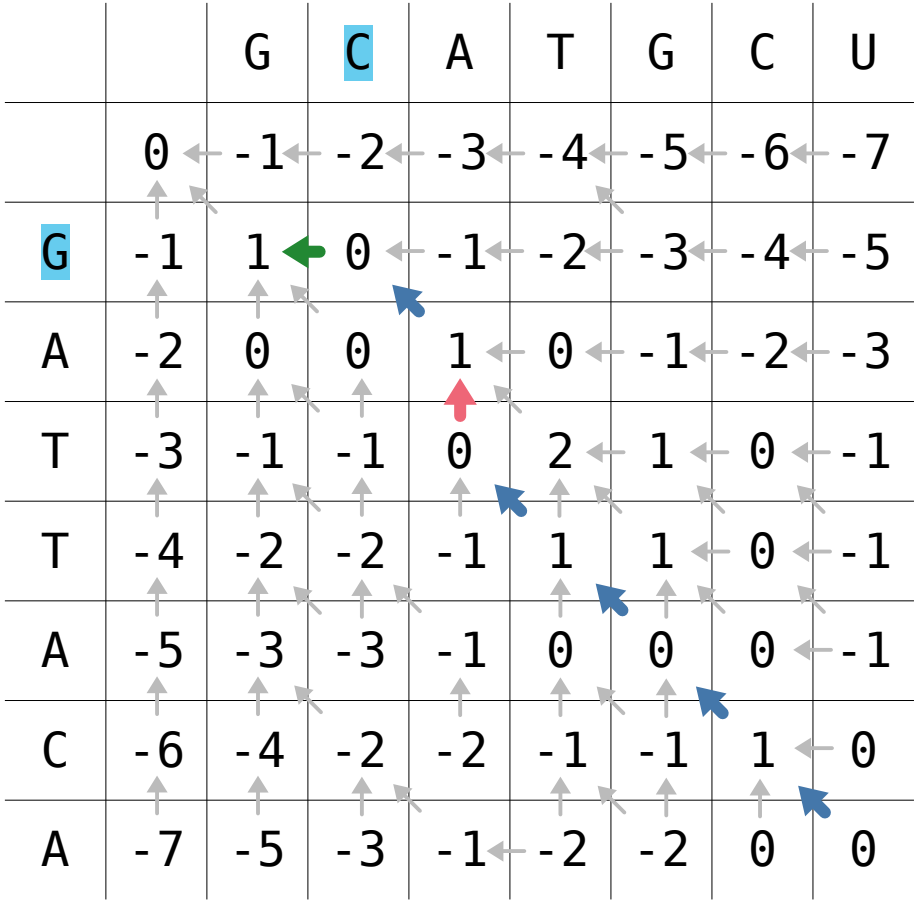


		G	C	A	T	G	C	U
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	0	-1	-2	-3	-4	-5
A	-2	0	0	1	0	-1	-2	-3
T	-3	-1	-1	0	2	1	0	-1
T	-4	-2	-2	-1	1	1	0	-1
A	-5	-3	-3	-1	0	0	0	-1
C	-6	-4	-2	-2	-1	-1	1	0
A	-7	-5	-3	-1	-2	-2	0	0

- Diagonal = Match:

  A-TGCU
  ATTACA

Schritt 3: Traceback (“Richtungen auswerten”)

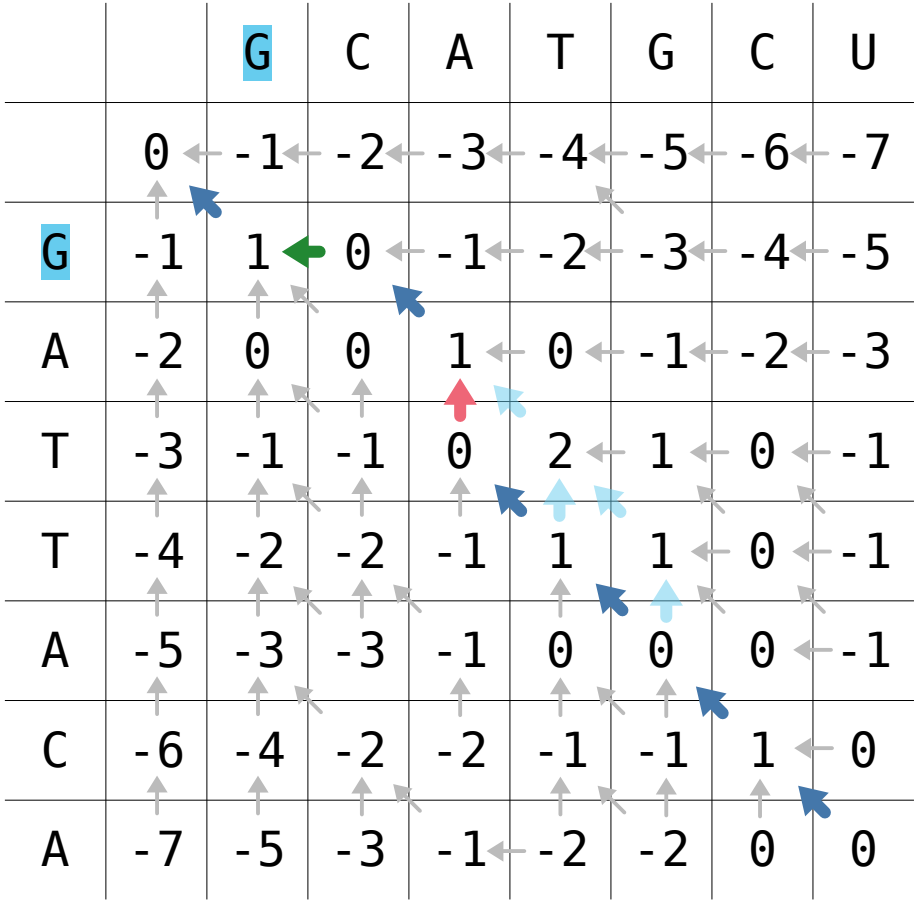


		G	C	A	T	G	C	U
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	0	-1	-2	-3	-4	-5
A	-2	0	0	1	0	-1	-2	-3
T	-3	-1	-1	0	2	1	0	-1
T	-4	-2	-2	-1	1	1	0	-1
A	-5	-3	-3	-1	0	0	0	-1
C	-6	-4	-2	-2	-1	-1	1	0
A	-7	-5	-3	-1	-2	-2	0	0

- Links = Gap in Sequenz 2:

```
CA-TGCU
-ATTACA
```

Schritt 3: Traceback (“Richtungen auswerten”)



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

- Diagonal = Match:

  GCA-TGCU
  G-ATTACA

- Ende erreicht: Alignment fertig!
  - Hellblaue, transparente Pfeile: Alternative Pfade
  - Ende wäre auch erreicht beim “Anstoßen” an Zeile/Spalte 0 (Erzeugt nur noch Gaps)

Eure Aufgabe

- Needleman-Wunsch implementieren
  - Input: 2 Sequenzen, die verschiedenen Scores
  - Output: Alignment (wie auf vorherigen Folien mit “ - ” für Gaps)
- Tipp: Überlegt euch zuerst welche Strukturen ihr braucht!
  - Welche Datenstrukturen brauche ich?
  - Wie kann ich Teile sinnvoll in Methoden auslagern?
  - insbesondere für Traceback: Wie merke ich mir die Richtungen?
- Viel Erfolg!