Sequence alignment

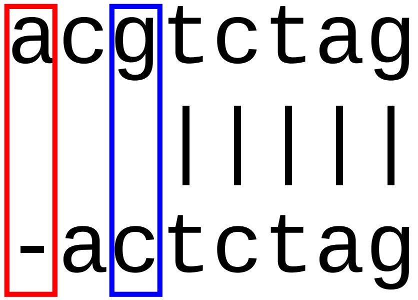
- Alignment specifies which positions in two sequences match

```
acgtctag   acgtctag    acgtctag
||            |||||    || |||||
actctag-   -actctag    ac-tctag
```

2 matches	5 matches	7 matches
5 mismatches	2 mismatches	0 mismatches
1 not aligned	1 not aligned	1 not aligned

Mutations: Insertions, deletions and substitutions

Indel: insertion or deletion of a base with respect to the ancestor sequence



```
acgtctag
   |||||
-actctag
```

Mismatch: substitution (point mutation) of a single base

- Insertions and/or deletions are called *indels*
  - *We can’t tell whether the ancestor sequence had a base or not at indel position*

Problems

- What sorts of alignments should be considered?
- How to score alignments?
- How to find optimal or good scoring alignments?
- How to evaluate the statistical significance of scores?

In this course, we discuss each of these problems briefly.

Course *Biological sequence analysis* tackles all four in-depth.

Sequence Alignment (chapter 6)

- The biological problem
- *Global alignment*
- Local alignment
- Multiple alignment

Global alignment

- Problem: find optimal scoring alignment between two sequences (Needleman & Wunsch 1970)
- Every position in both sequences is included in the alignment
- We give score for each position in alignment

Identity (match)	+1
Substitution (mismatch)	$-\mu$
Indel	$-\delta$

```
WHAT
||
WH-Y
```

- Total score: sum of position scores

S(WHAT/WH-Y) = $1 + 1 - \delta - \mu$

Dynamic programming

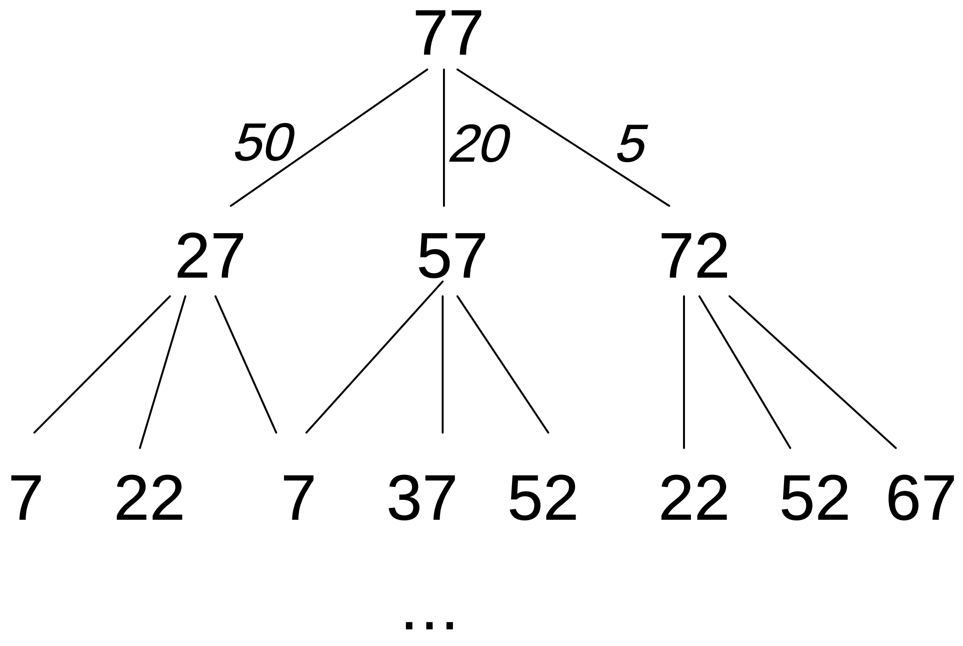
- How to find the optimal alignment?
- We use previous solutions for optimal alignments of smaller subsequences
- This general approach is known as dynamic programming

Introduction to dynamic programming: the money change problem

- Suppose you buy a pen for 4.23€ and pay for with a 5€ note
- You get 77 cents in change – what coins is the cashier going to give you if he or she tries to minimise the number of coins?
- The usual algorithm: start with largest coin (denominator), proceed to smaller coins until no change is left:
  - 50, 20, 5 and 2 cents
- This greedy algorithm is *incorrect*, in the sense that it does not always give you the correct answer

The money change problem

- How else to compute the change?
- We could consider all possible ways to reduce the amount of change
- Suppose we have 77 cents change, and the following coins: 50, 20, 5 cents
- We can compute the change with recursion
- Figure shows the recursion tree for the example

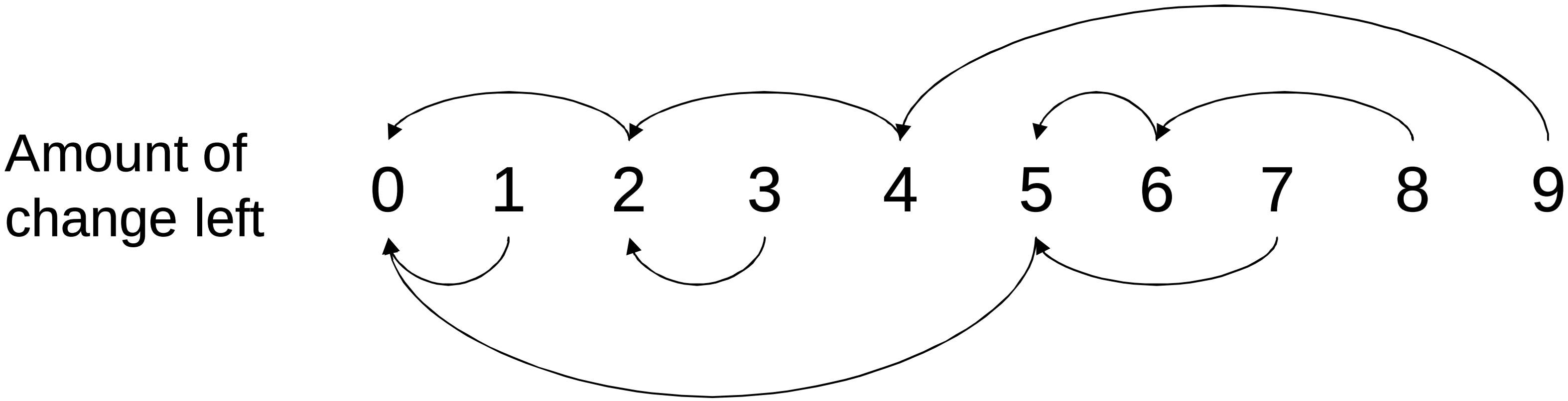


- Many values are computed more than once!
- This leads to a correct but very inefficient algorithm

The money change problem

- We can speed the computation up by solving the change problem for all $i \leq n$
  - Example: solve the problem for 9 cents with available coins being 1, 2 and 5 cents
- Solve the problem in steps, first for 1 cent, then 2 cents, and so on
- In each step, utilise the solutions from the previous steps

The money change problem



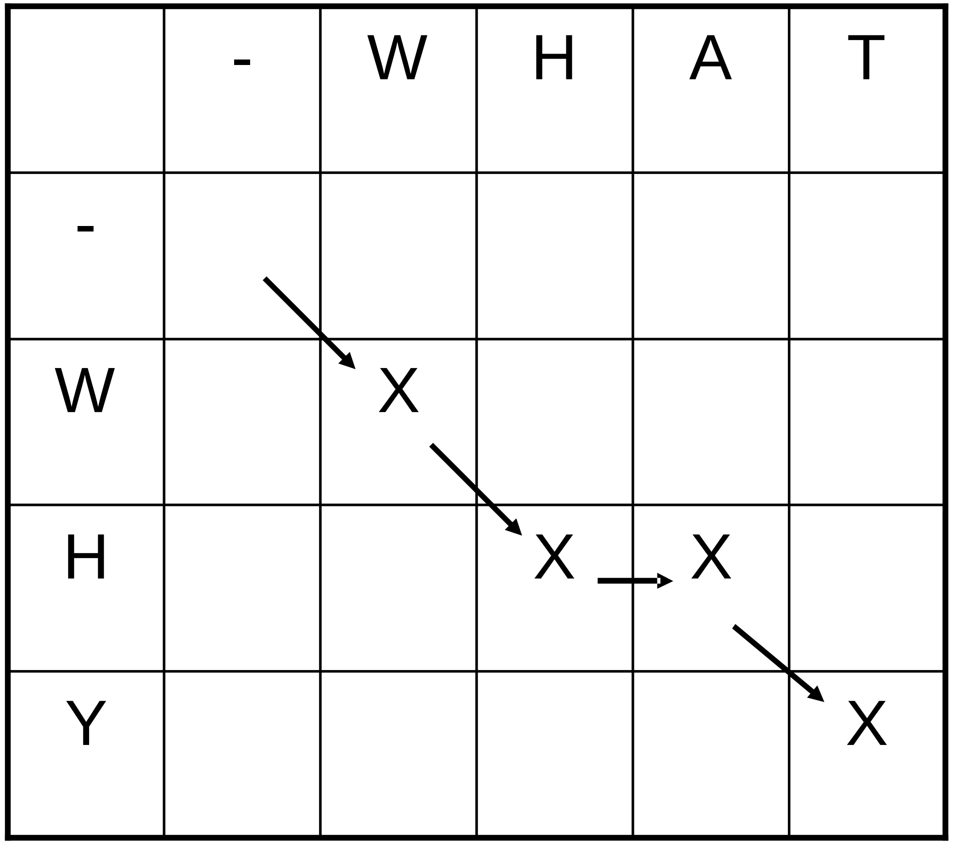
- Algorithm runs in time proportional to Md, where M is the amount of change and d is the number of coin types
- The same technique of storing solutions of subproblems can be utilised in aligning sequences

Representing alignments and scores

Alignments can be represented in the following tabular form.

Each alignment corresponds to a path through the table.

```
WHAT
||
WH-Y
```



	-	W	H	A	T
-					
W		X			
H			X	X	
Y					X

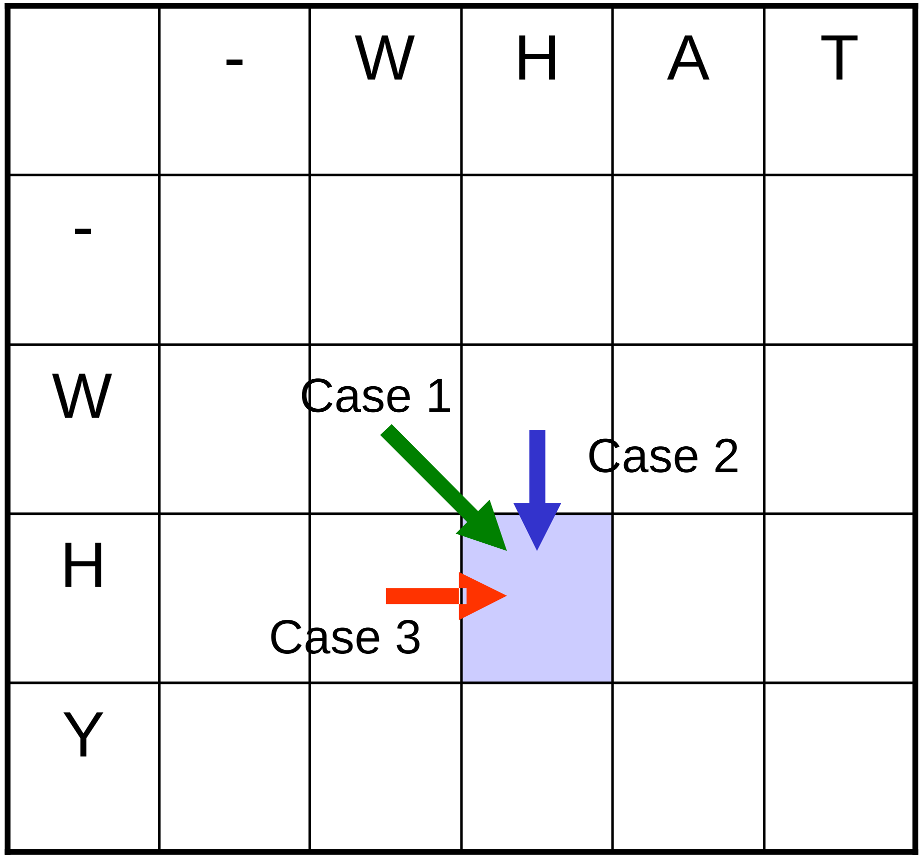
Representing alignments and scores

```
WHAT
||
WH-Y
```

Global alignment score $S_{3,4}$ = 2-δ-μ

	-	W	H	A	T
-	0				
W		1			
H			2	2-δ	
Y					2-δ-μ

Filling the alignment matrix



	-	W	H	A	T
-					
W					
H					
Y					

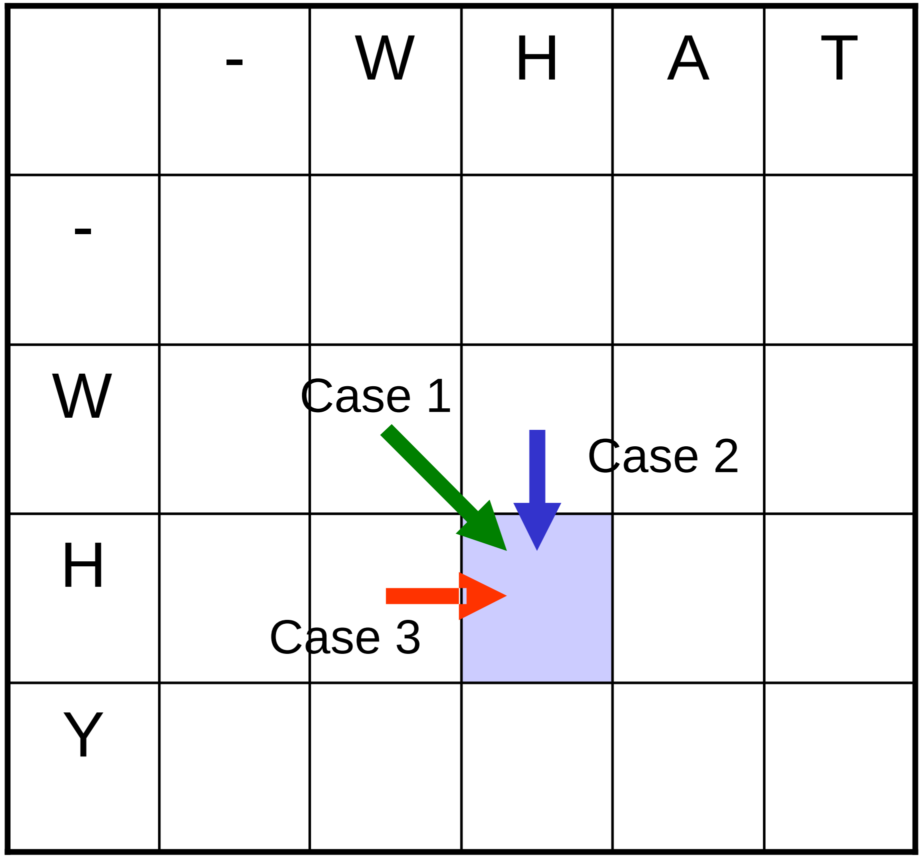
Consider the alignment process at shaded square.

Case 1. Align H against H (match or substitution).

Case 2. Align H in WHY against – (indel) in WHAT.

Case 3. Align H in WHAT against – (indel) in WHY.

Filling the alignment matrix (2)



Scoring the alternatives.

Case 1. $S_{2,2} = S_{1,1} + s(2, 2)$

Case 2. $S_{2,2} = S_{1,2} – \delta$

Case 3. $S_{2,2} = S_{2,1} – \delta$

$s(i, j) = 1$ for matching positions,

$s(i, j) = - \mu$ for substitutions.

Choose the case (path) that yields the maximum score.

Keep track of path choices.

Global alignment: formal development

$A = a_1a_2a_3\ldots a_n$,

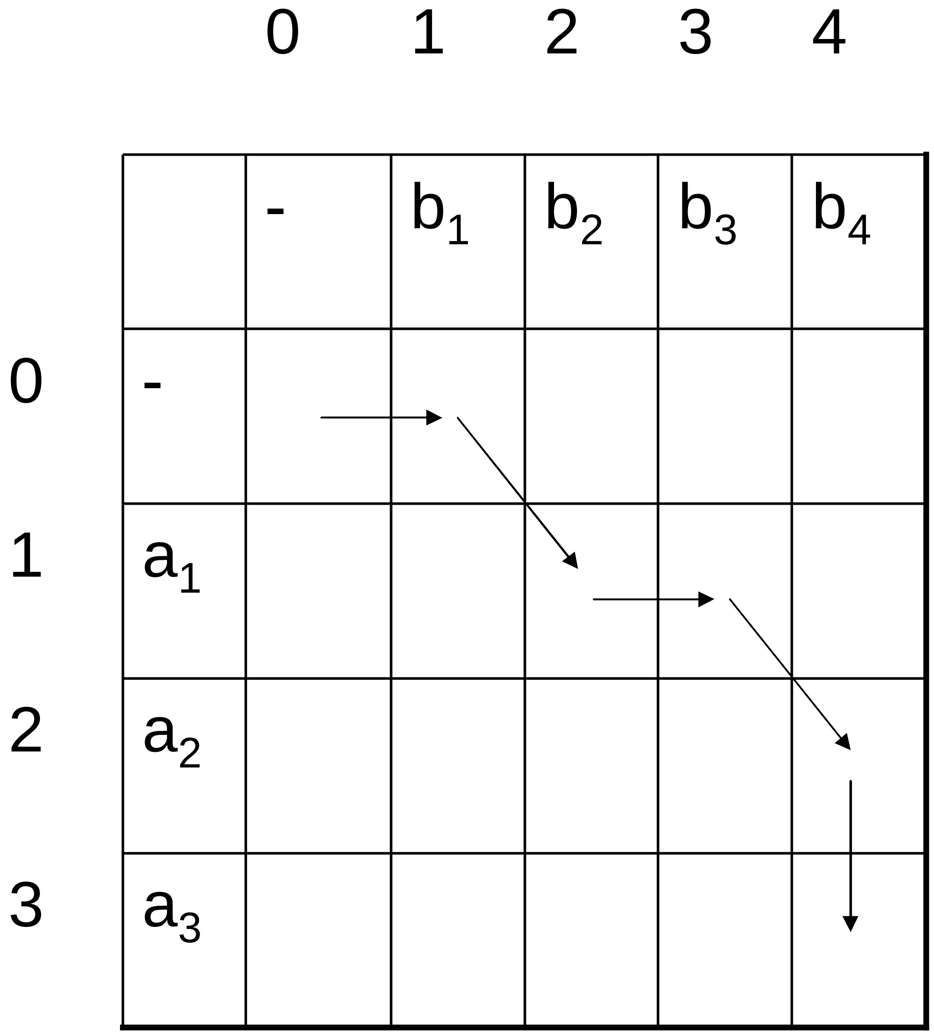
$B = b_1b_2b_3\ldots b_m$

b_1	b_2	b_3	b_4	-
-	a_1	-	a_2	a_3

- Any alignment can be written as a unique path through the matrix

- Score for aligning A and B up to positions i and j:

$S_{i,j} = S(a_1a_2a_3\ldots a_i, b_1b_2b_3\ldots b_j)$



		0	1	2	3	4
		-	b_1	b_2	b_3	b_4
0	-					
1	a_1					
2	a_2					
3	a_3					

Scoring partial alignments

- Alignment of A = $a_1a_2a_3\ldots a_n$ with B = $b_1b_2b_3\ldots b_m$ can end in three ways
  - Case 1: $(a_1a_2\ldots a_{i-1})\ a_i$

    $(b_1b_2\ldots b_{j-1})\ b_j$
  - Case 2: $(a_1a_2\ldots a_{i-1})\ a_i$

    $(b_1b_2\ldots b_j)$ -
  - Case 3: $(a_1a_2\ldots a_i)$ –

    $(b_1b_2\ldots b_{j-1})\ b_j$

Scoring alignments

- Scores for each case:
  - Case 1: $(a_1 a_2 \ldots a_{i-1})\ a_i$

    $(b_1 b_2 \ldots b_{j-1})\ b_j$
  - Case 2: $(a_1 a_2 \ldots a_{i-1})\ a_i$

    $(b_1 b_2 \ldots b_j)$ –
  - Case 3: $(a_1 a_2 \ldots a_i)$ –

    $(b_1 b_2 \ldots b_{j-1})\ b_j$

$$s(a_i, b_j) = \begin{cases} +1 \text{ if } a_i = b_j \\ -\mu \text{ otherwise} \end{cases}$$

$$s(a_i, -) = s(-, b_j) = -\delta$$

Scoring alignments (2)

- First row and first column correspond to initial alignment against indels:

  $S(i, 0) = -i\,\delta$

  $S(0, j) = -j\,\delta$

- Optimal global alignment score $S(A, B) = S_{n,m}$

		0	1	2	3	4
		-	b_1	b_2	b_3	b_4
0	-	0	-δ	-2δ	-3δ	-4δ
1	a_1	-δ				
2	a_2	-2δ				
3	a_3	-3δ				

Algorithm for global alignment

```
Input sequences A, B, n = |A|, m = |B|
Set $S_{i,0}$ := -δi for all i
Set $S_{0,j}$ := -δj for all j
for i := 1 to n
    for j := 1 to m
        $S_{i,j}$ := max{$S_{i-1,j}$ – δ, $S_{i-1,j-1}$ + s($a_i$,$b_j$), $S_{i,j}$-1 – δ}
    end
end
```

Algorithm takes O(nm) time and space.

Global alignment: example

μ = 1

δ = 2

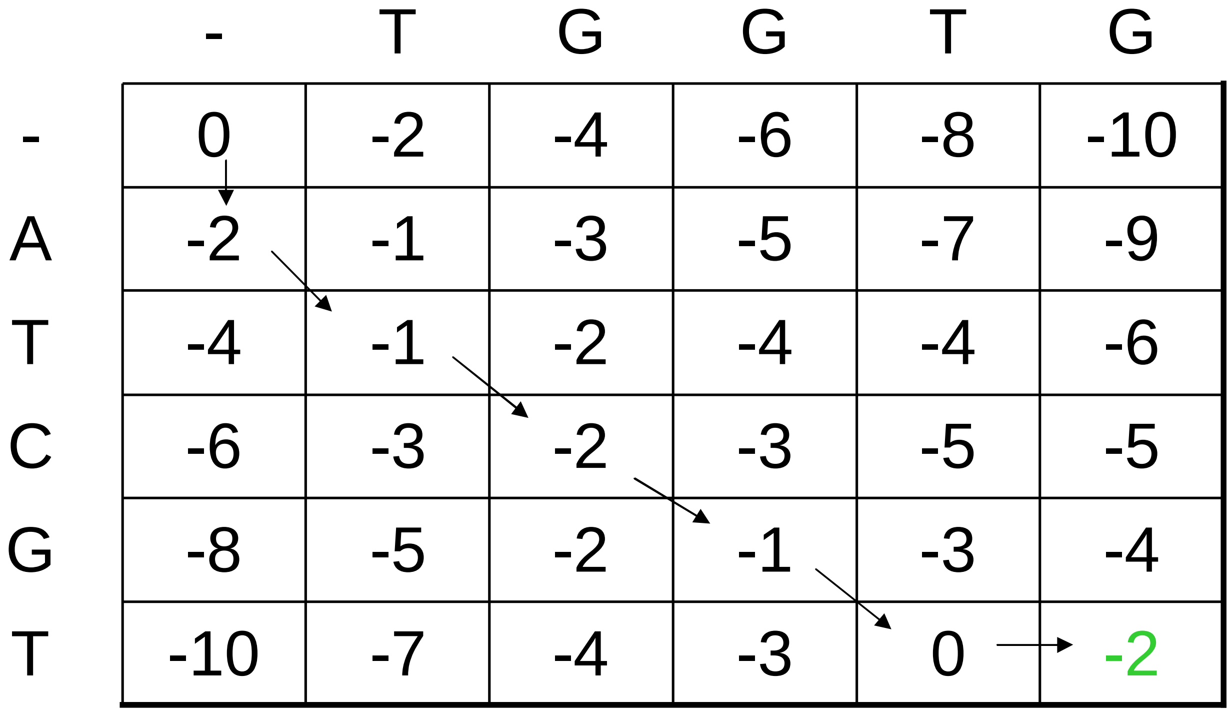
	-	T	G	G	T	G
-	0	-2	-4	-6	-8	-10
A	-2					
T	-4					
C	-6					
G	-8					
T	-10					?

Global alignment: example (2)

μ = 1

δ = 2

```
ATCGT-
 |  ||
-TGGTG
```



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

Sequence Alignment (chapter 6)

- The biological problem
- Global alignment
- *Local alignment*
- Multiple alignment

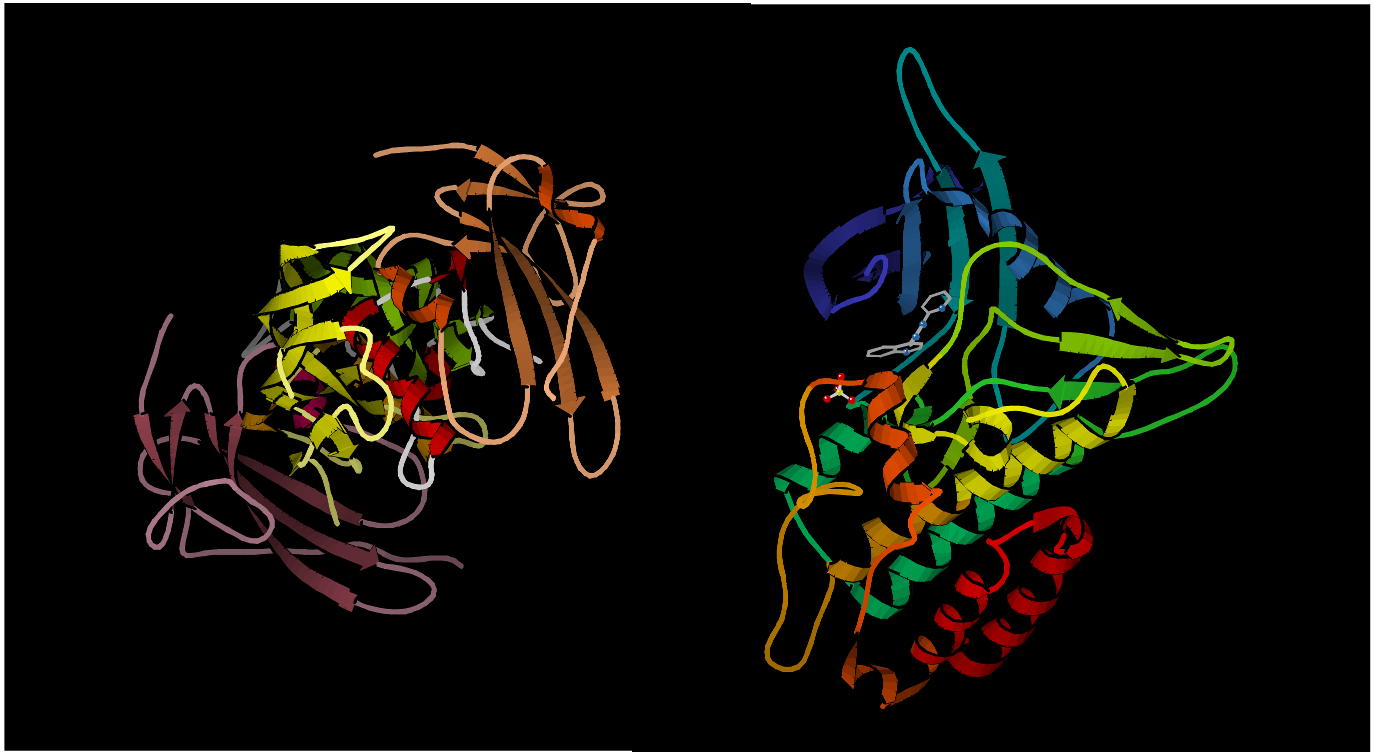
Local alignment: rationale

- Otherwise dissimilar proteins may have local regions of similarity

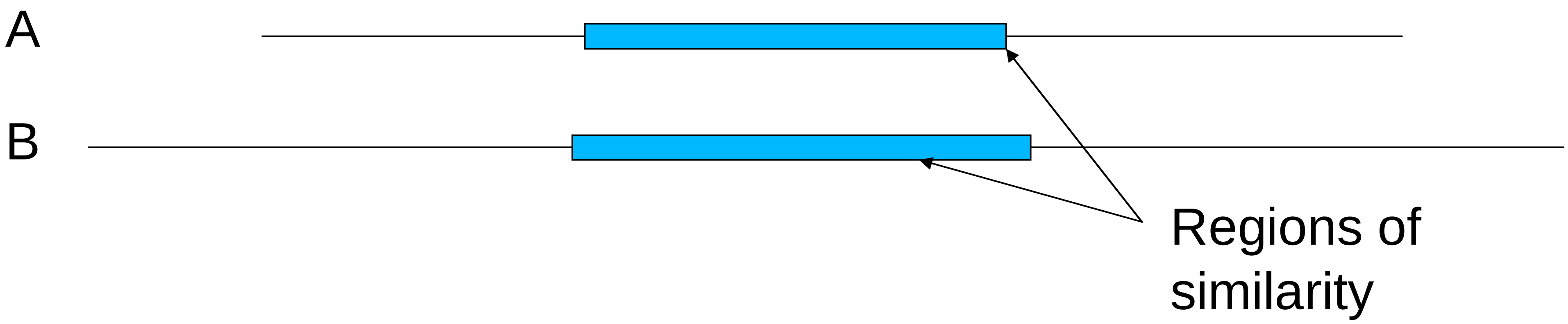
  -> Proteins may share a function

Human bone morphogenic protein receptor type II precursor (left) has a 300 aa region that resembles 291 aa region in TGF-β receptor (right).

The shared function here is protein kinase.



Local alignment: rationale



- Global alignment would be inadequate
- Problem: find the highest scoring *local* alignment between two sequences
- Previous algorithm with minor modifications solves this problem (Smith & Waterman 1981)

From global to local alignment

- Modifications to the global alignment algorithm
  - Look for the highest-scoring path **in** the alignment matrix (not necessarily through the matrix), or in other words:
  - Allow preceding and trailing indels without penalty

Scoring local alignments

A = $a_1a_2a_3\ldots a_n$, B = $b_1b_2b_3\ldots b_m$

Let I and J be intervals (substrings) of A and B, respectively: $I \subset A$, $J \subset B$

Best local alignment score:

$$M(A, B) = \max\{S(I, J) : I \subset A, J \subset B\}$$

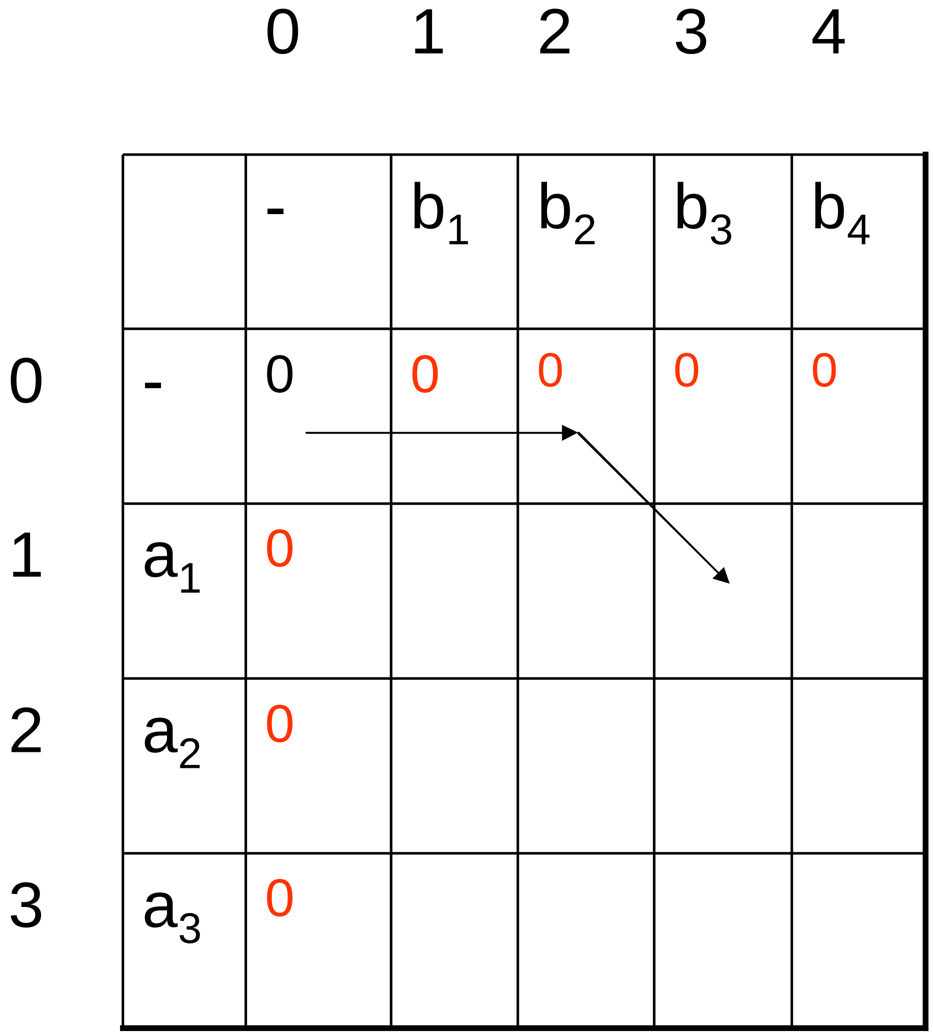
where S(I, J) is the score for substrings I and J.

Allowing preceding and trailing indels

- First row and column initialised to zero:

$M_{i,0} = M_{0,j} = 0$

```
b1 b2 b3
-  -  a1
```



		0	1	2	3	4
		-	b_1	b_2	b_3	b_4
0	-	0	0	0	0	0
1	a_1	0				
2	a_2	0				
3	a_3	0				

Recursion for local alignment

- $M_{i,j} = \max \{$

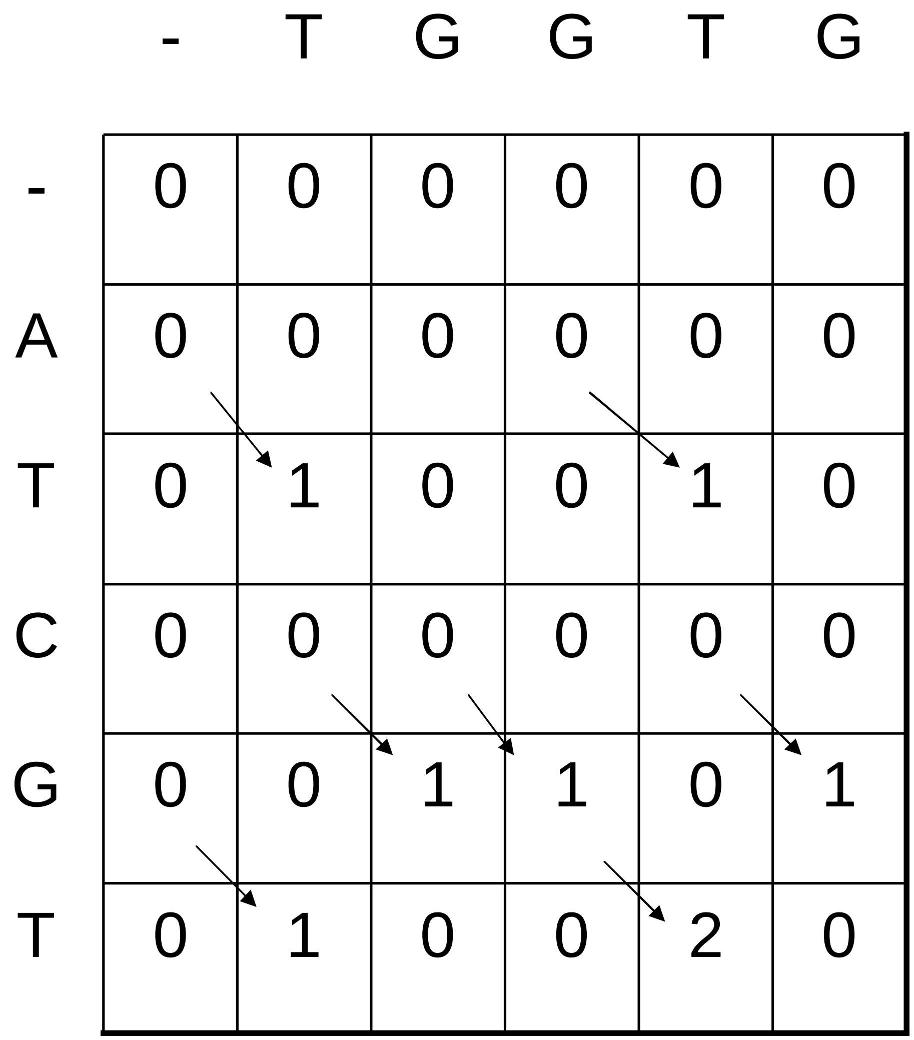
  $M_{i-1,j-1} + s(a_i, b_i),$

  $M_{i-1,j} - \delta,$

  $M_{i,j-1} - \delta,$

  0

  $\}$



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

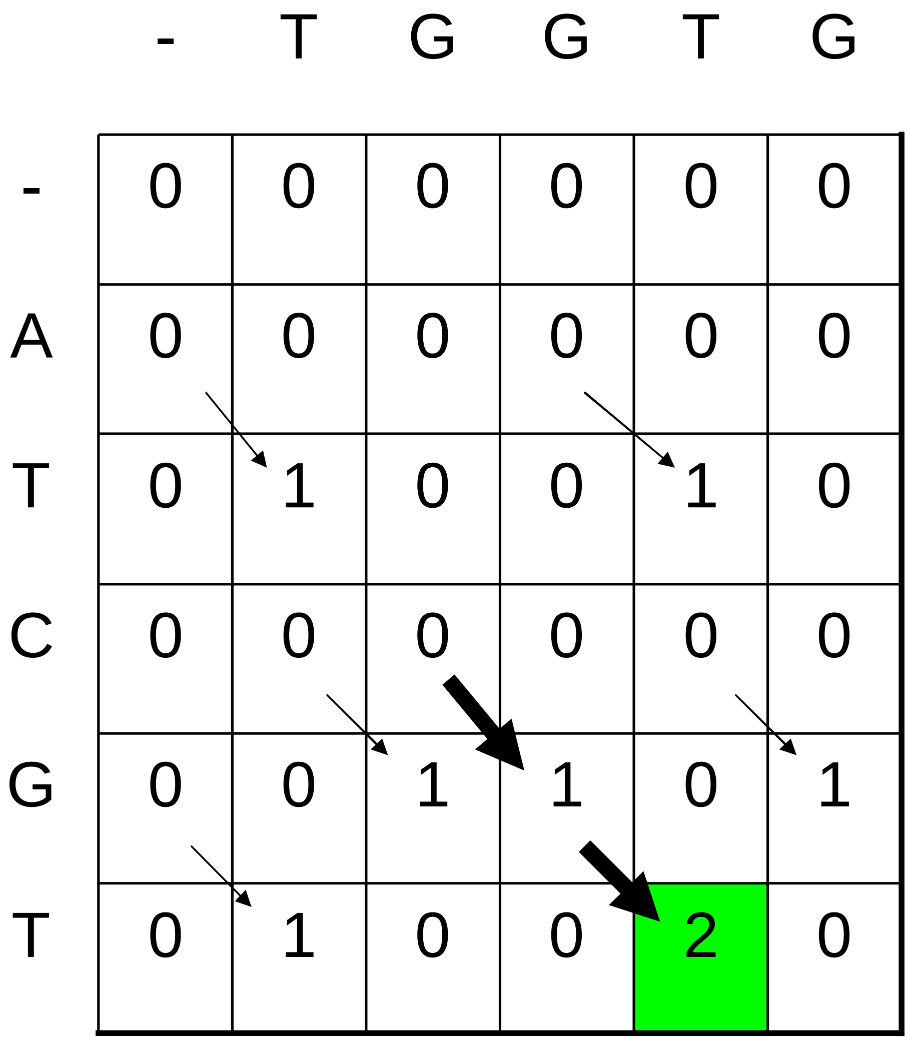
Finding best local alignment

- Optimal score is the highest value in the matrix

$$M(A, B) = \max\{S(I, J) : I \subset A, J \subset B\}$$

$$= \max_{i,j} M_{i,j}$$

- Best local alignment can be found by backtracking from the highest value in M



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

Local alignment: example

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

Local alignment: example

Scoring

Match: +2

Mismatch: -1

Indel: -2

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

C T – A A
C T C A A

Non-uniform mismatch penalties

- We used uniform penalty for mismatches:

s('A', 'C') = s('A', 'G') = … = s('G', 'T') = μ

- Transition mutations (A->G, G->A, C->T, T->C) are approximately twice as frequent than transversions (A->T, T->A, A->C, G->T)
  - use non-uniform mismatch penalties collected into a *substitution matrix*

	A	C	G	T
A	1	-1	-0.5	-1
C	-1	1	-1	-0.5
G	-0.5	-1	1	-1
T	-1	-0.5	-1	1

Gaps in alignment

- Gap is a succession of indels in alignment

```
C T – - - A A
C T C G C A A
```

- Previous model scored a length k gap as w(k) = -kδ
- Replication processes may produce longer stretches of insertions or deletions
  - In coding regions, insertions or deletions of codons may preserve functionality

Gap open and extension penalties (2)

- We can design a score that allows the penalty opening gap to be larger than extending the gap:

$$w(k) = -\alpha - \beta(k - 1)$$

- Gap open cost α, Gap extension cost β
- Our previous algorithm can be extended to use w(k) (not discussed on this course)

Amino acid sequences

- We have discussed mainly dna sequences
- Amino acid sequences can be aligned as well
- However, the design of the substitution matrix is more involved because of the larger alphabet
- More on the topic in the course Biological sequence analysis

Sequence Alignment (chapter 6)

- The biological problem
- Global alignment
- Local alignment
- *Multiple alignment*

Multiple alignment

- Consider a set of n sequences on the right
  - Orthologous sequences from different organisms
  - Paralogs from multiple duplications
- How can we study relationships between these sequences?

```
aggcgagctgcgagtgcta
cgttagattgacgctgac
ttccggctgcgac
gacacggcgaacgga
agtgtgcccgacgagcgaggac
gcgggctgtgagcgcta
aagcggcctgtgtgcccta
atgctgctgccagtgta
agtcgagccccgagtgc
agtccgagtcc
actcggtgc
```

Optimal alignment of three sequences

- Alignment of A = $a_1a_2…a_i$ and B = $b_1b_2…b_j$ can end either in (-, b_j), (a_i, b_j) or (a_i, -)
- $2^2 – 1 = 3$ alternatives
- Alignment of A, B and C = $c_1c_2…c_k$ can end in $2^3 – 1$ ways: (a_i, -, -), (-, b_j, -), (-, -, c_k), (-, b_j, c_k), (a_i, -, c_k), (a_i, b_j, -) or (a_i, b_j, c_k)
- Solve the recursion using three-dimensional dynamic programming matrix: O(n^3) time and space
- Generalizes to n sequences but impractical with moderate number of sequences

Multiple alignment in practice

- In practice, real-world multiple alignment problems are usually solved with heuristics
- Progressive multiple alignment
  - Choose two sequences and align them
  - Choose third sequence w.r.t. two previous sequences and align the third against them
  - Repeat until all sequences have been aligned
  - Different options how to choose sequences and score alignments

Multiple alignment in practice

- Profile-based progressive multiple alignment: CLUSTALW
  - Construct a distance matrix of all pairs of sequences using dynamic programming
  - Progressively align pairs in order of decreasing similarity
  - CLUSTALW uses various heuristics to contribute to accuracy

Additional material

- R. Durbin, S. Eddy, A. Krogh, G. Mitchison: Biological sequence analysis
- N. C. Jones, P. A. Pevzner: An introduction to bioinformatics algorithms
- Course Biological sequence analysis in Spring 2008

Demonstration of the EBI web site

- European Bioinformatics Institute (EBI) offers many biological databases and bioinformatics tools at http://www.ebi.ac.uk/