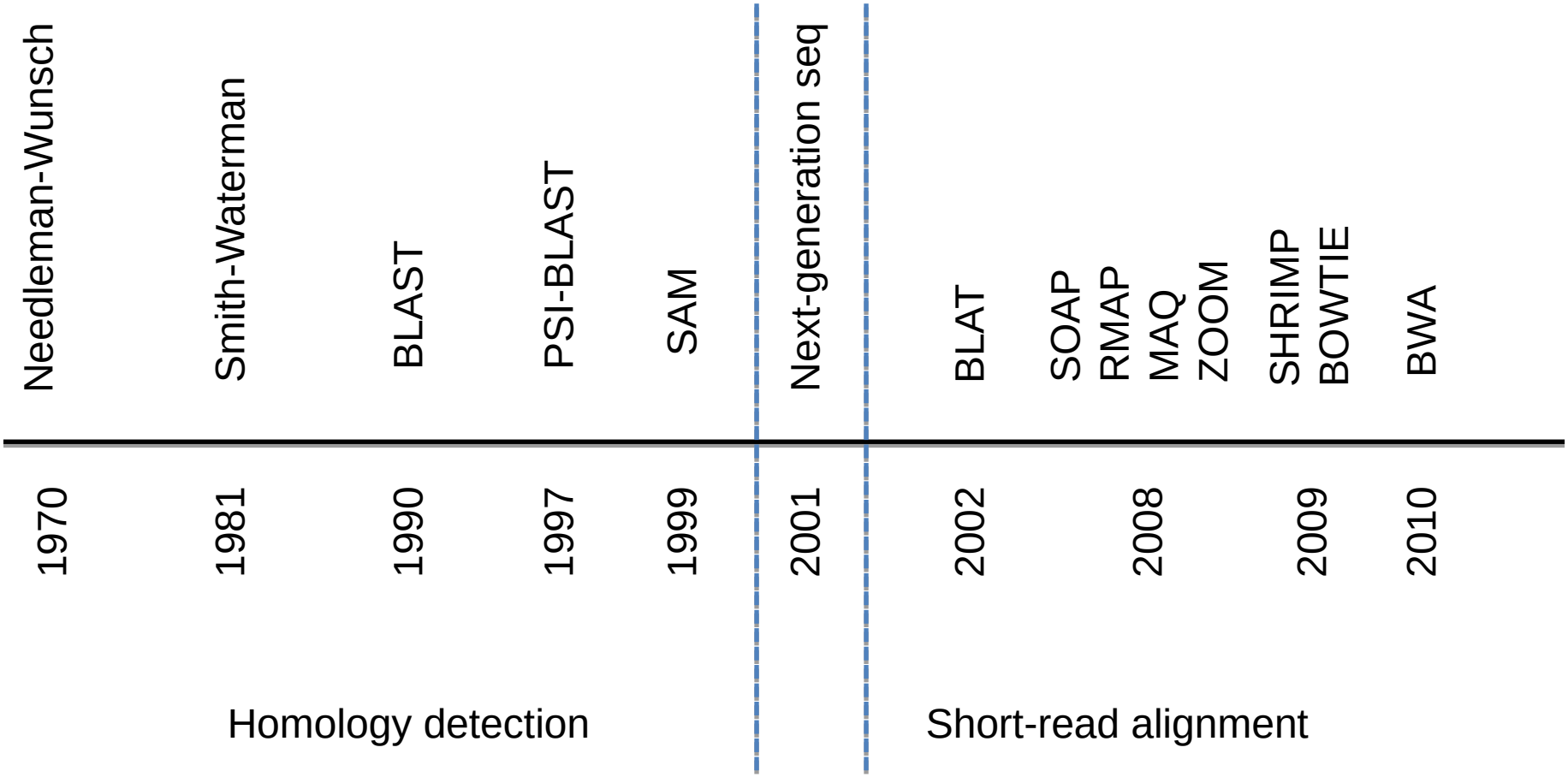
Lecture 5
Pairwise sequence alignment algorithms

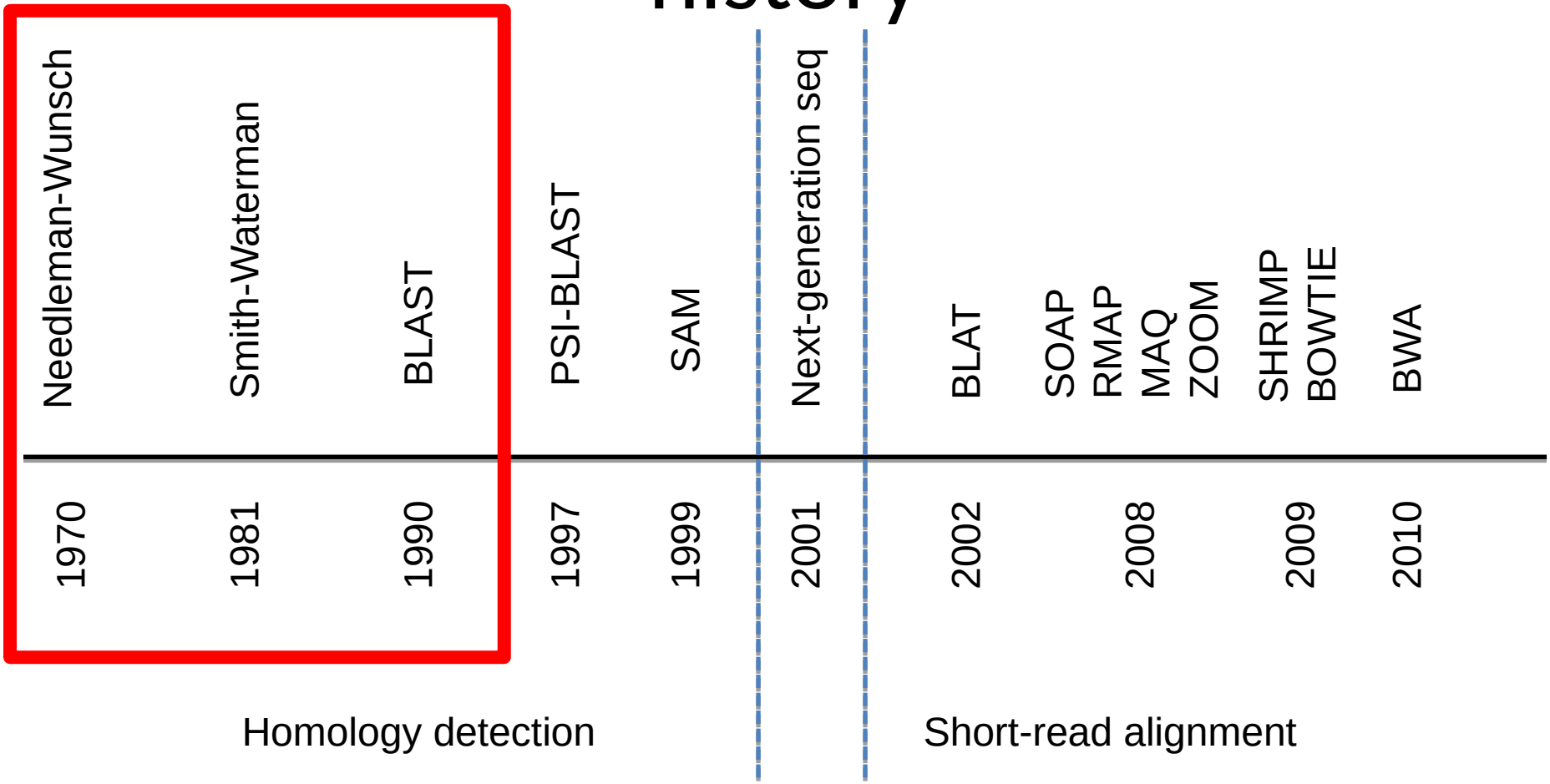
Rachel Karchin

BME 580.488, 580.688
Spring 2019

Some pairwise sequence alignment history



Some pairwise sequence alignment history



Alignment for homology detection

- When we compare sequences, we are looking for evidence that they have diverged from a common ancestor by a process of mutation and selection.

Durbin et al. Biological Sequence Analysis

Fundamentals

- Assume a simplified model of evolution
  - Substitutions
  - Deletions
  - Insertions

Durbin et al. Biological Sequence Analysis

Fundamentals

- Null hypothesis
  - Similarity between sequences is due to chance
- Alternative hypothesis
  - Similarity between sequences is due to a common ancestor

Conceptualizing pairwise alignment

- Tabular representation
- Scoring system
- Search strategy

Pairwise alignment basics

Two sequences

```
GACCC
GACAA
```

What is the best way to match up their shared equivalent positions?

Pairwise alignment basics

Two sequences

```
GACCC
GACAA
```

What is the best way to match up their shared equivalent positions?

```
GACCC
GACAA
```

```
GACCC--
--GACAA
```

```
GACC-C
GACAA-
```

```
G-ACCC
GACAA-
```

```
GACCC-
GA-CAA
```

```
GACCC-
GAC-AA
```

Pairwise alignment basics

Two sequences

GACCC
GACAA

Tabular representation

	G	A	C	A	A
G					
A					
C					
C					
C					

Pairwise alignment basics

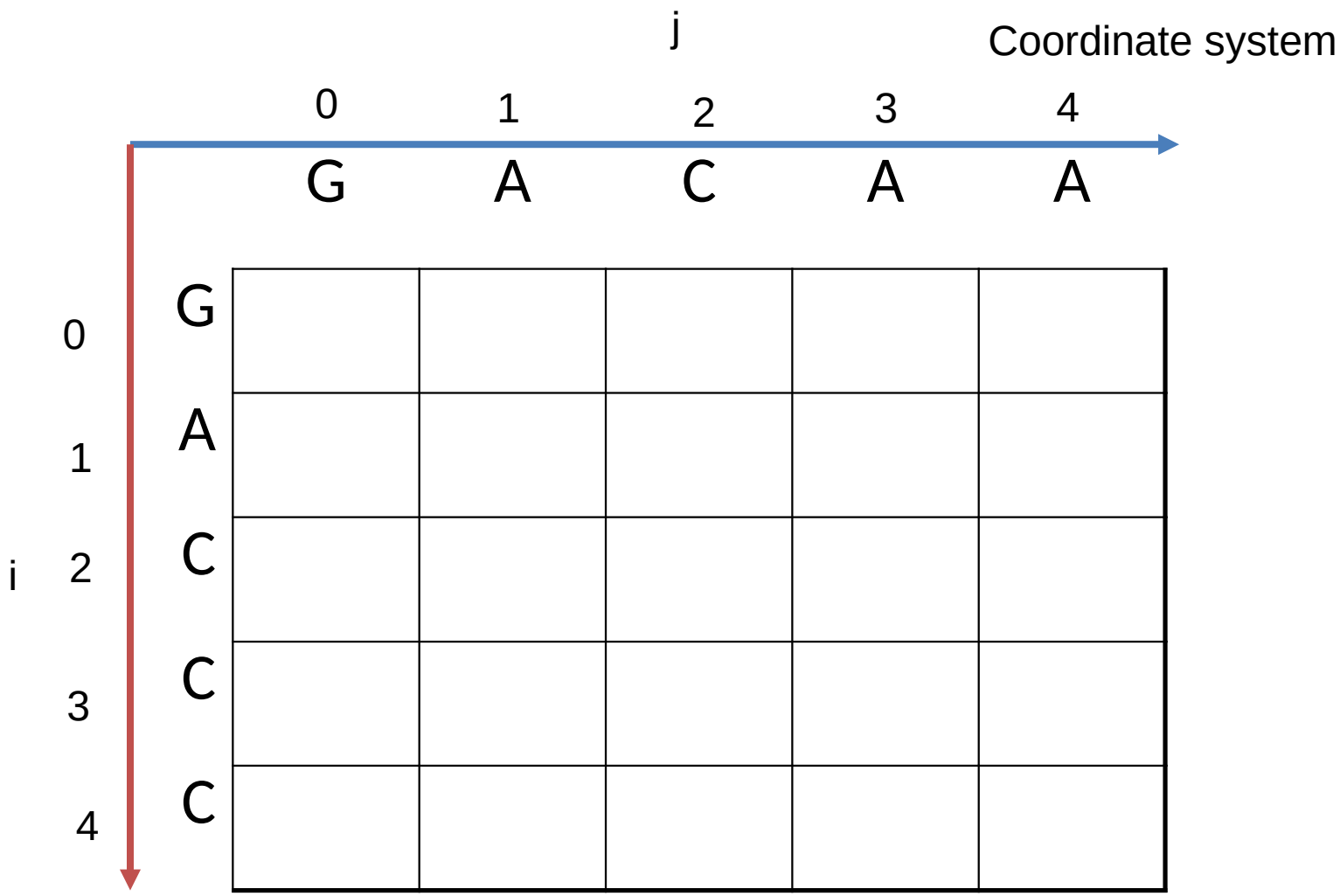
j

Coordinate system

i

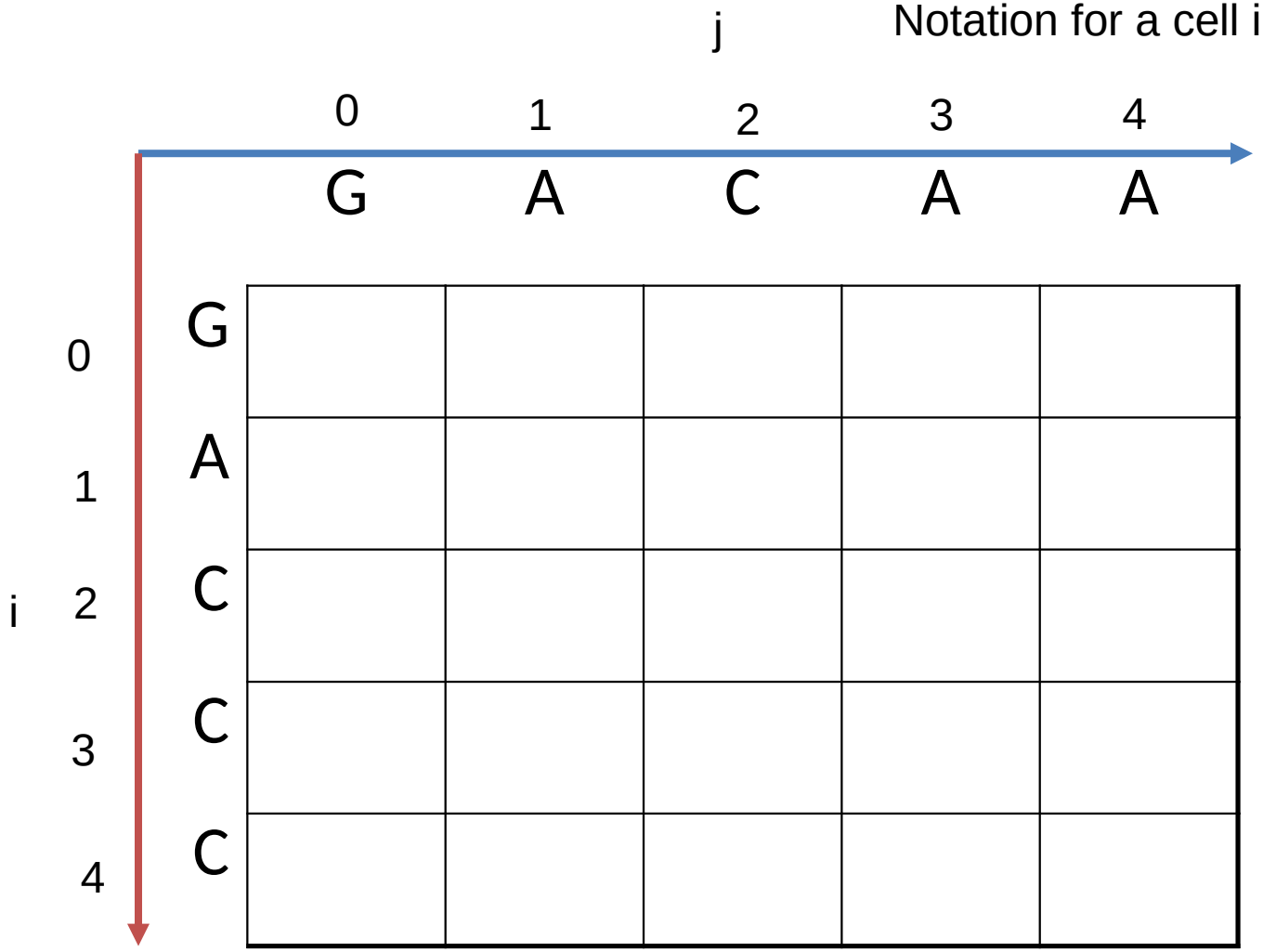
	G	A	C	A	A
G					
A					
C					
C					
C					

Pairwise alignment basics



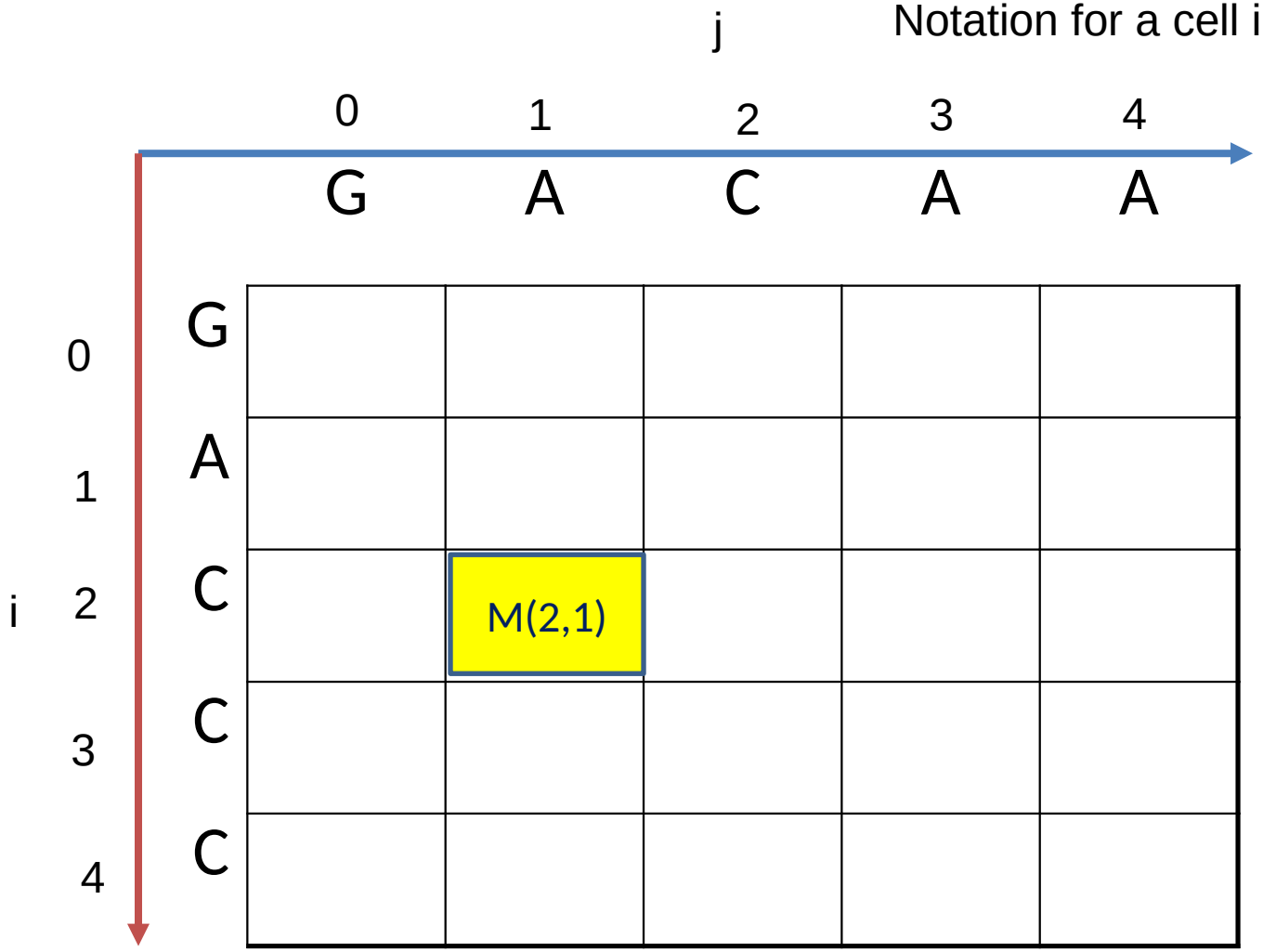
Pairwise alignment basics

Notation for a cell in the matrix is M(i,j)



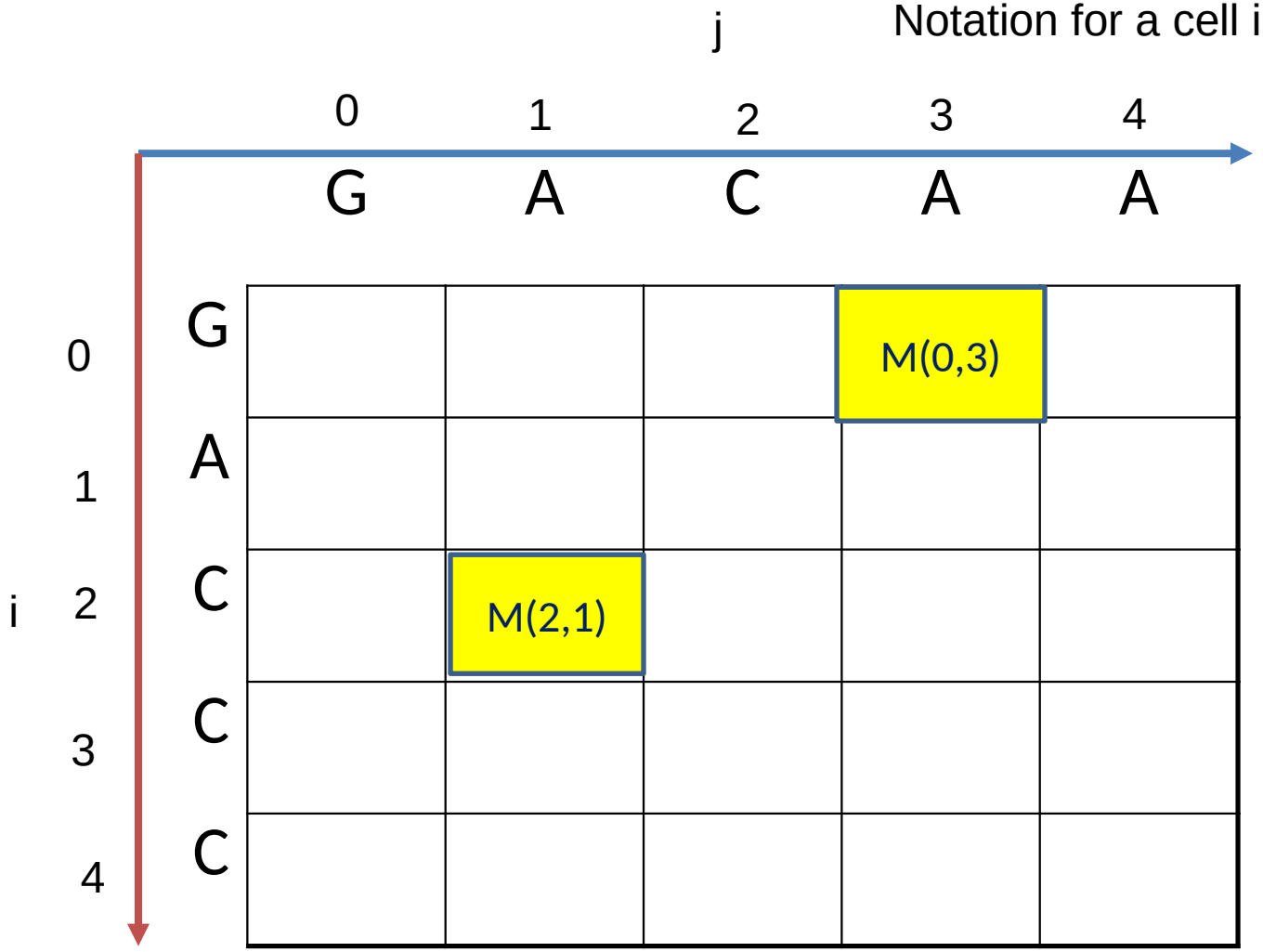
Pairwise alignment basics

Notation for a cell in the matrix is M(i,j)

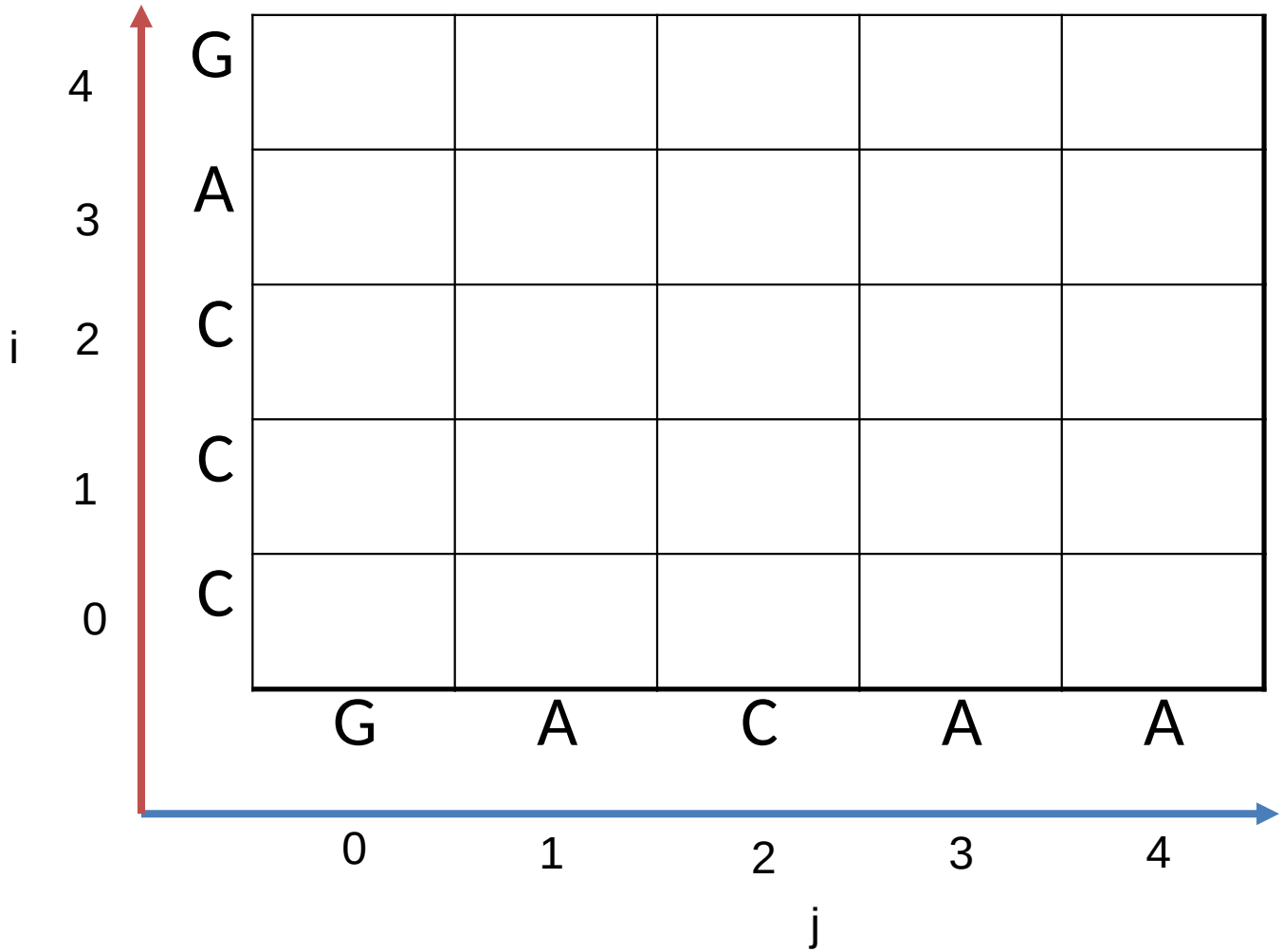


Pairwise alignment basics

Notation for a cell in the matrix is M(i,j)



Pairwise alignment basics



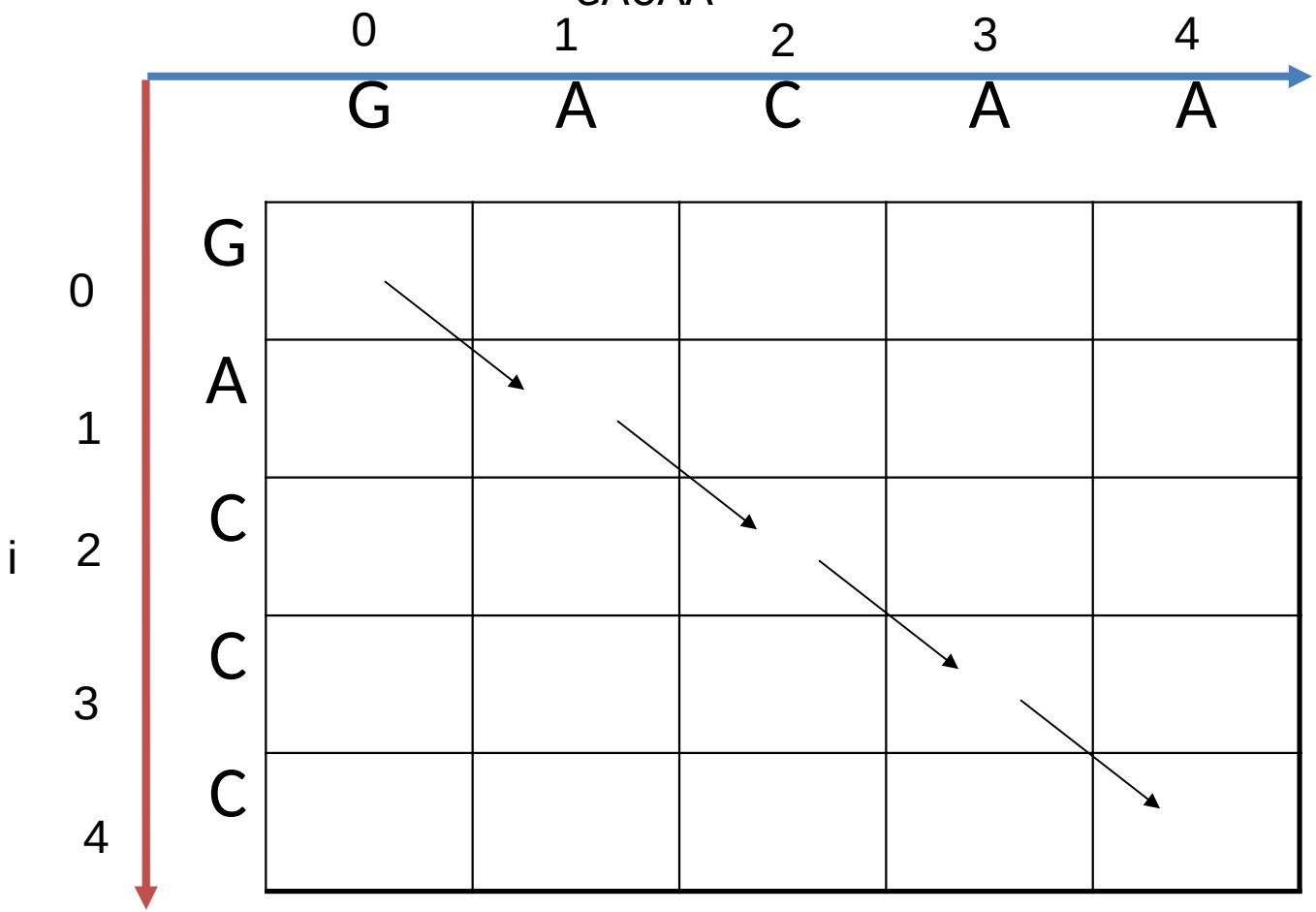
Coordinate system sometimes defined this way.

Pairwise alignment basics

Two sequences

GACCC
GACAA

Alignment is a path through the matrix



Pairwise alignment basics

What path is this alignment?

```
GACCC-
G-ACAA
```

	G	A	C	A	A
G					
A					
C					
C					
C					

Pairwise alignment basics

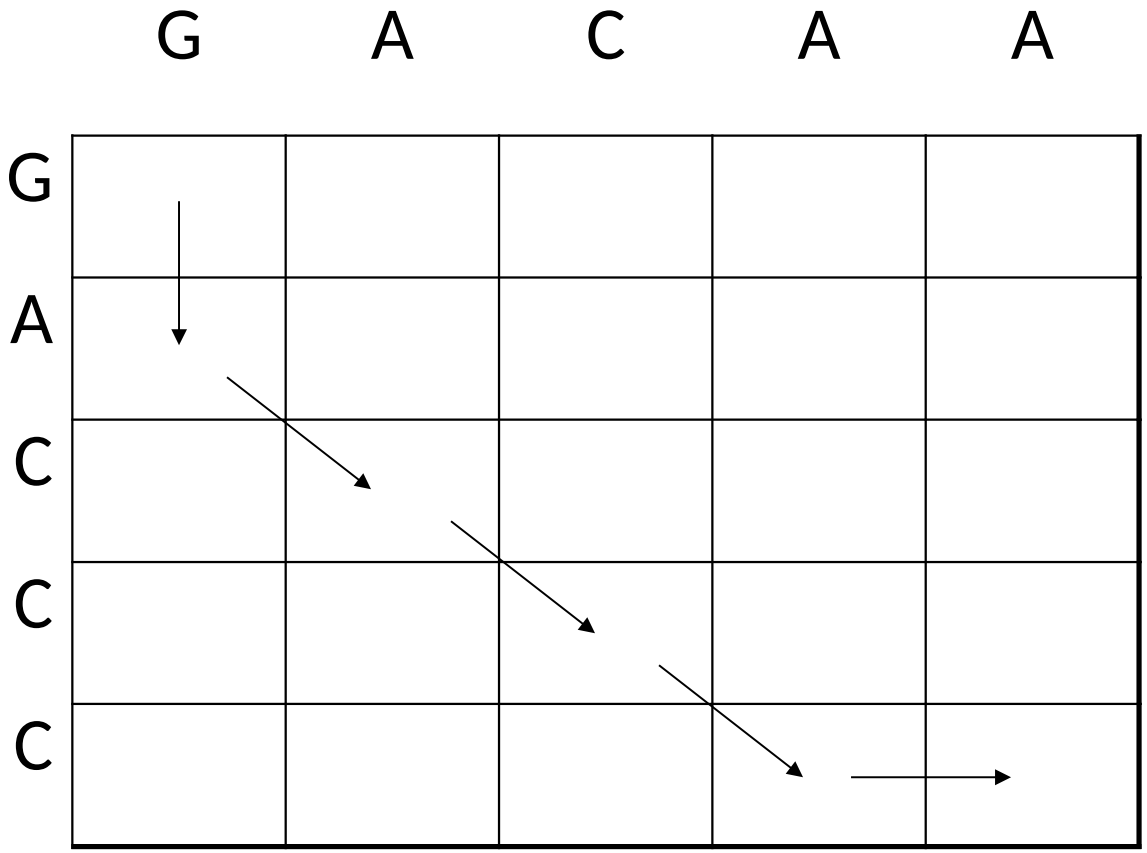
Let GACCC be the “reference”

GACCC-
G-ACAA

	G	A	C	A	A
G					
A					
C					
C					
C					

Pairwise alignment basics

GACCC-
G-ACAA



Pairwise alignment

- Giving the very large number of possible pairwise alignments for any two sequences, which one is best?

Pairwise alignment

- Requires a scoring system that gives points or penalties for matches and gaps

Pairwise alignment

- Also requires efficient search strategy to explore the very large alignment space.

Scoring system

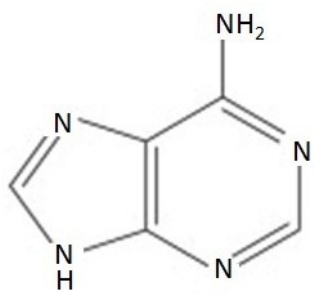
- “Match” points
- Gap costs (penalties) for insertions and deletions (indels)
- Any alignment can be scored by its matches (which are rewarded) and gap costs (which are penalized)

What do we want from a scoring system?

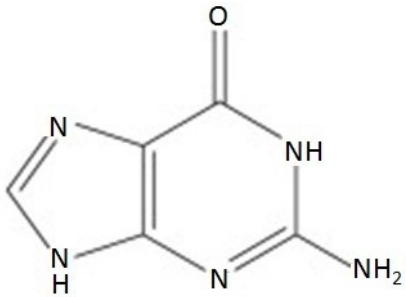
- For homology detection
  - Better scores for pairs of sequences that have diverged from a common ancestor
  - Worse scores for pairs that have not
- For short read alignment
  - Better scores for true matches
  - Worse scores for false matches

DNA/RNA sequences are composed of nucleotides

purines



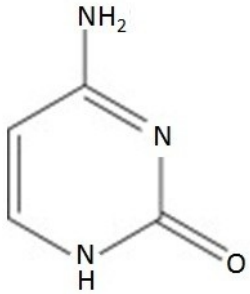
Adenine



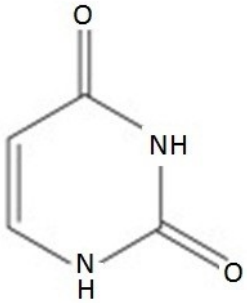
Guanine

DNA
A
G
C
T

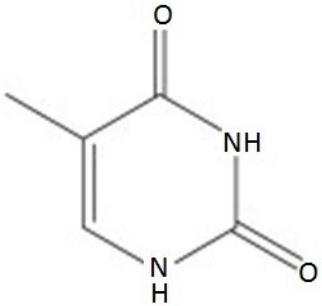
pyrimidines



Cytosine



Uracil



Thymine

RNA
A
G
C
U

4 nucleotide bases

Scoring matrices

- “Score” associated with each possible type of “match” or substitution

	A	C	G	T
A	6	1	2	1
C	1	6	1	2
G	2	1	6	1
T	1	2	1	6

Kimura scoring matrix

Gap penalties

- Constant gap penalty
- Linear gap penalty
- Affine gap penalty

Constant Gap penalties

- Simplest approach
  - Cost for opening a gap δ
  - Every gap gets same penalty, regardless of length

Linear Gap penalties

- Cost depends on the length of the gap
- Gap of length k gets cost of kc
- Where c is the "unit" gap cost

Affine Gap penalties

- A linear transformation followed by a translation
- Gap opening penalty δ and linear gap extension penalty ε
- Gap of length k gets penalty $\delta + (k-1)\,\varepsilon$

Gap penalties

- Constant gap penalty
- Linear gap penalty
- Affine gap penalty

Which works best?

How do we set parameters δ and ε?

Gap cost = *kc*

Linear Gap penalties

- Each gap of one base gets penalized the same, regardless of whether or not it is contiguous with other gaps
- Alignment with fewer gaps favored
- Penalty for large gap is same as many small gaps

What do you think about this from a biological point of view?

Gap cost = $\delta + (k-1)\varepsilon$

Affine Gap penalties

- Biologically, more likely to see a single gap of 10 than 10 single gaps
- We can model this using a gap opening penalty δ and a gap extension penalty ε
- Gap of length k gets penalty $\delta + (k-1)\varepsilon$

Should ε be smaller than δ ?
Why?

How can we efficiently search through alignment space?

Dynamic programming for sequence alignment

- Key insight is that we can align prefixes and then extend them one position at a time
- Efficient search through the huge alignment space is done by caching intermediate results

Global pairwise sequence alignment

- Global pairwise sequence alignment by dynamic programming introduced by Needleman and Wunsch in 1970.
- Known as the Needleman-Wunsch algorithm.

Dynamic programming for sequence alignment

- General idea

Matrix Initialization

Two sequences

GACCC
ACAA

		A	C	A	A
	0	-2	-2	-2	-2
G	-2				
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

- General idea

Aligning each nucleotide with the empty string is equivalent to putting a gap at the beginning of the alignment.

What kind of gap penalty is being used here?

Matrix Initialization

Why?

Two sequences

GACCC
ACAA

		A	C	A	A
	0	-2	-2	-2	-2
G	-2				
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

- General idea

What is this?

		A	C	A	A
	0	-2	-2	-2	-2
G	-2				
A	-2				
C	-2				
C	-2				
C	-2				

Matrix Initialization

Two sequences

GACCC
ACAA

Dynamic programming for sequence alignment

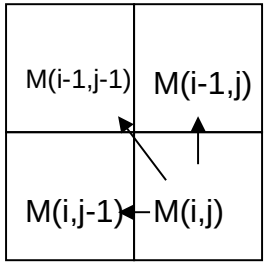
- General idea

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2				
A	-2				
C	-2				
C	-2				
C	-2				



To fill in each M(i,j), need to look where?

- General idea

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2				
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

- General idea

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

M(1,1) = max(
M(0,0) +S(G,A),
M(1,0) + ω(gap in seq 1),
M(0,1) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	M(1,1)			
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

- General idea

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

ω(.)= -2
constant

M(1,1) = max(
M(0,0) +S(G,A),
M(1,0) - 2,
M(0,1) -2

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	M(1,1)			
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

- General idea

Matrix
Fill

Recurrence relation

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

M(1,1) = max(
M(0,0) + 2,
M(1,0) - 2,
M(0,1) -2

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	M(1,1)			
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

M(1,1) = max(
0 + 2,
-2 - 2,
-2 - 2

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2			
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

M(1,1) = max(
0 + 2,
-2 - 2,
-2 - 2

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2			
A	-2				
C	-2				
C	-2				
C	-2				

Add a pointer to trace the step along the path we chose

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0		
A	-2				
C	-2				
C	-2				
C	-2				

M(1,2) = max(
-2 - 1,
2 - 2,
-2 - 2

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0		
A	-2				
C	-2				
C	-2				
C	-2				

M(1,2) = max(
-2 - 1,
2 - 2,
-2 - 2

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	
A	-2				
C	-2				
C	-2				
C	-2				

M(1,3) = max(
-2 + 2,
0 - 2,
-2 - 2

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	
A	-2				
C	-2				
C	-2				
C	-2				

M(1,3) = max(
-2 + 2,
0 - 2,
-2 - 2

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2				
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0			
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0			
C	-2				
C	-2				
C	-2				

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Only one “winner” is shown for each max operation, but there are often ties, with the winner selected at random. Thus, there are many optimal alignments
and even more that are slightly less than optimal (“sub-optimal alignments”)

$$M(i,j) = \max\big(M(i-1,j-1) + S(i,j),\ M(i,j-1) + \omega(\text{gap in seq 1}),\ M(i-1,j) + \omega(\text{gap in seq 2})\big)$$

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Complexity?

Matrix
Fill

Recurrence relation

M(i,j) = max(
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Where do we start?

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

For global alignment

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

For global alignment

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

In this coordinate system, bottom right cell gives us the optimal global alignment score

Dynamic programming for sequence alignment

Traceback

Want to identify the alignment over the length of both sequences

Backtrack over the winning path

How?

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Want to recover the (an) optimal alignment over the length of both sequences

Backtrack over the winning path

How?

Have to identify the direction into which each cell was reached.

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Want to recover the (an) optimal alignment over the length of both sequences

Backtrack over the winning path

How?

Have to identify the direction into which each cell was reached.

Forward pointers tell us

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

What does our alignment look like at this point?

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

What does our alignment look like at this point?

CCC
CAA

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

ACCC
-CAA

Dynamic programming for sequence alignment

Traceback

Follow the path backwards

Identify the (a) winning path

GACCC
A-CAA

		A	C	A	A
	0	-2	-2	-2	-2
G	-2	2	0	0	0
A	-2	0	1	2	2
C	-2	-2	2	0	1
C	-2	-3	0	1	-1
C	-2	-3	-1	-1	0

- Global sequence alignment is not always desirable.
  - Useful when two sequences are very similar

- When there are short regions of high similarity and longer regions of low similarity, we want to use local sequence alignment

Local pairwise sequence alignment

- Ten years after Needleman-Wunsch
- Smith-Waterman (1981)

Local pairwise sequence alignment

- General idea

Recurrence relation

$$M(i,j) = \max(0,\ M(i-1,j-1) + S(i,j),\ M(i,j-1) + \omega(\text{gap in seq 1}),\ M(i-1,j) + \omega(\text{gap in seq 2}))$$

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

$\omega(.) = -2$

constant

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

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Zero “win” => start of a new alignment (reset)

Alignment can begin and end anywhere in the matrix

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

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

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

Local pairwise sequence alignment

Do we set a pointer from the parent to this cell?

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

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

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

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

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	
A	0				
C	0				
C	0				
C	0				

Local pairwise sequence alignment

Do we set a pointer from the parent to this cell?

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	
A	0				
C	0				
C	0				
C	0				

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Recurrence relation

M(i,j) = max(
0,
M(i-1,j-1) +S(i,j),
M(i,j-1) + ω(gap in seq 1),
M(i-1,j) + ω(gap in seq 2)
)

Scoring system:

S(i,j)

	Pur	Pyr
Pur	2	-1
Pyr	-1	2

ω(.)= -2
constant

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

Want to recover the (an) optimal alignment ~~over the length of both sequences~~

Backtrack over the winning paths

How?

Where do we start?

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

Want to recover the (an) optimal alignment ~~over the length of both sequences~~

Backtrack over the winning paths

How?

Where do we start?

Largest value in the matrix is score of the best local alignment between subsequences

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

What does this alignment look like?

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Local pairwise sequence alignment

Traceback

What does this alignment look like?

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

```
cc
cc
```

Local pairwise sequence alignment

Traceback

For this to work, what is requirement for expected value of a random match?

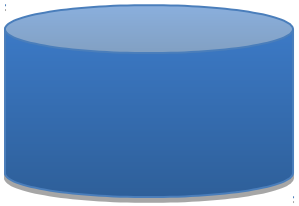
Why?

		C	C	A	A
	0	0	0	0	0
G	0	0	0	2	2
A	0	0	0	2	4
C	0	2	2	0	2
C	0	2	4	2	0
C	0	2	4	3	1

Trends in homology-based pairwise sequence alignment

Align two sequences

Align a target sequence with everything in a sequence database



Exponential growth in database size

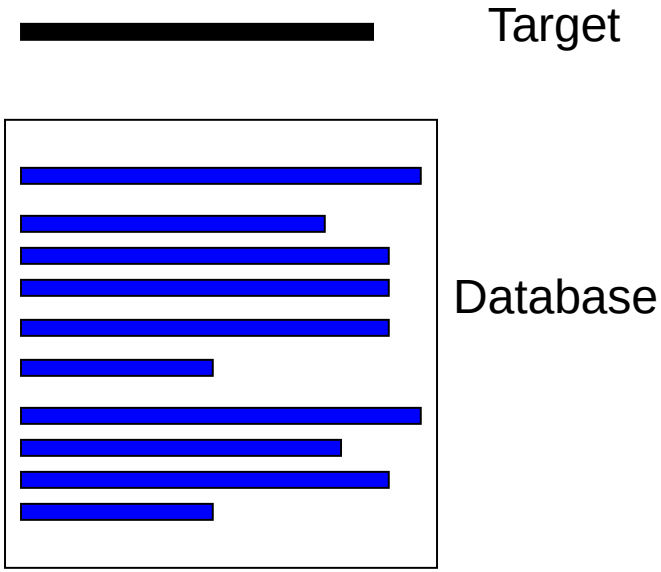
Motivation for heuristic alignments

- For two sequences of length *m* and *n*
  - Time to get optimal alignment score is $\Theta(mn)$
    - Θ notation like O notation but provides upper and lower bounds on asymptotic behavior (O is only for upper)
  - Space required to recover the alignment (naively $\Theta(mn)$ but can reduce to $\Theta(m+n)$
- Not realistic for large scale problems

BLAST

Key insight:

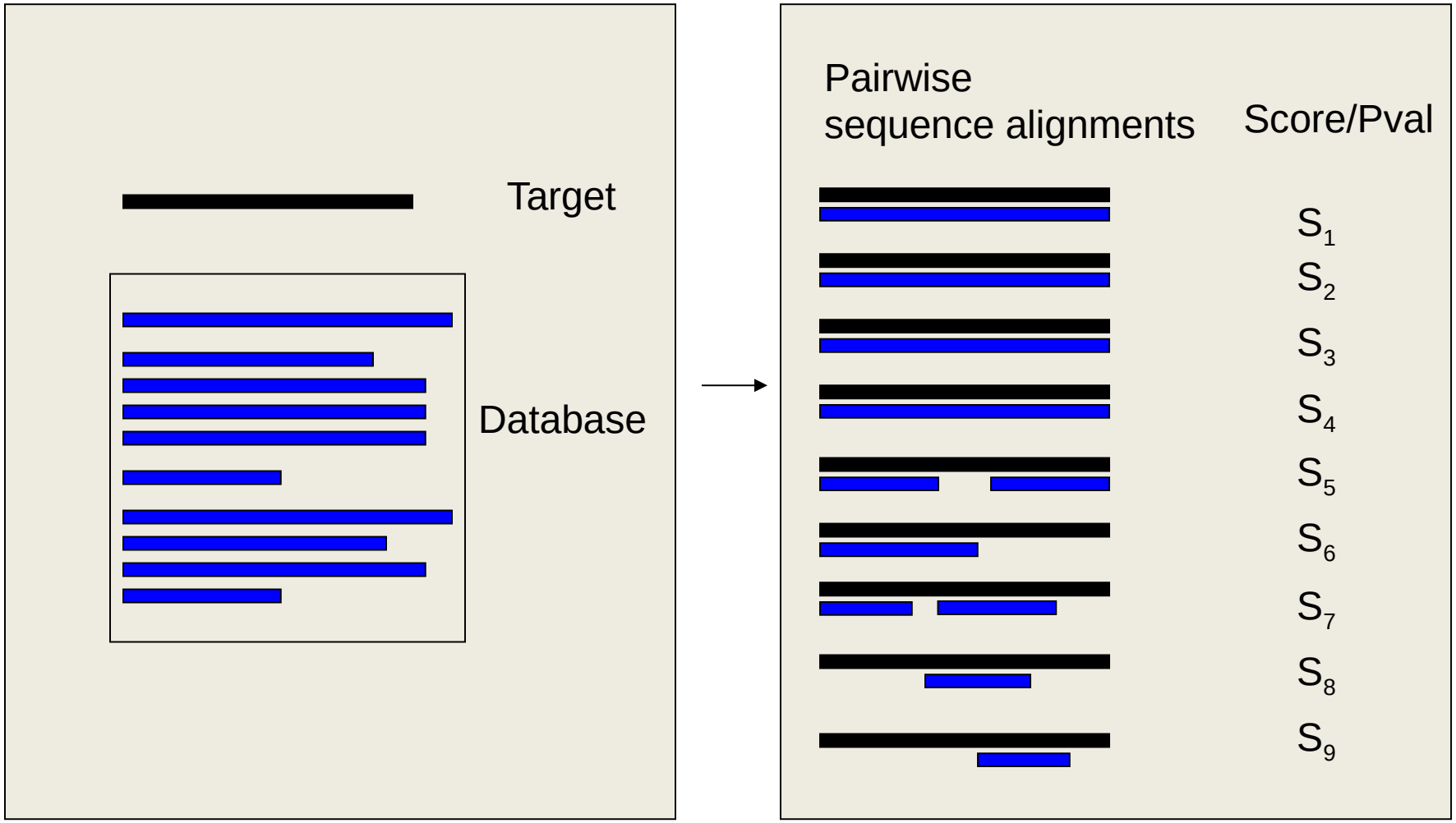
Biologically real alignments very likely to contain a short stretch of identities or very high scoring matches.



Algorithm

1. Build "words" — find short statistically significant sub-sequences in query
2. Find "seeds" — scan sequences in database for matching words
3. Extend — use (nearby) seeds to form local alignments called HSPs
4. Score — combine groups of consistent HSPs into local alignment with best score

BLAST

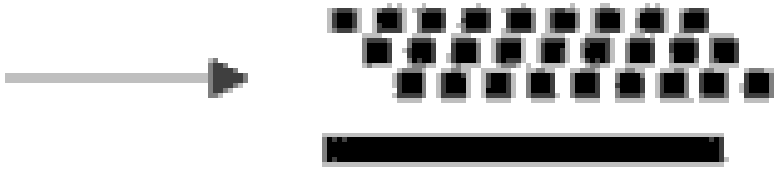

Target
Database
Pairwise
sequence alignments
Score/Pval
S_1
S_2
S_3
S_4
S_5
S_6
S_7
S_8
S_9

BLAST

- Key parameters:
  - *W* (word length)
  - *T* (threshold for scores in the initial word list)

BLAST word list

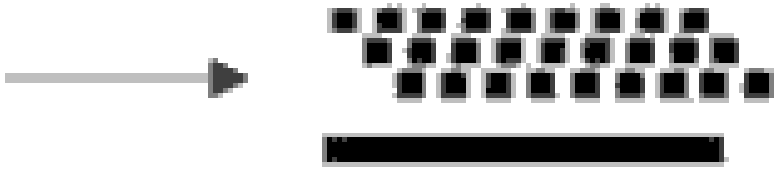
1. Build "words" — find short statistically significant sub-sequences in query



Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST word list

1. Build "words" — find short statistically significant sub-sequences in query



Q ACTGA *D* GACTGC

Word List

W=3

ACT
CTG
TGA

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST database seed finding

2. Find "seeds" — scan sequences in database for matching words

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST database seed finding

2. Find "seeds" — scan sequences in database for matching words

Q ACTGA *D* GACTGC

Word List

ACT
CTG
TGA

Seeds?

What are they?

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST database seed finding

2. Find "seeds" — scan sequences in database for matching words

Q ACTGA *D* GACTGC

Word List	Seeds?
ACT	ACT
CTG	CTG
TGA	

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST database seed finding

2. Find "seeds" — scan sequences in database for matching words

Q ACTGA *D* GACTGC

Word List	Seeds?
ACT	ACT
CTG	CTG
TGA	

What kind of data structures might you use to store the words?
The word/seed associations?

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST seed extension

3. Extend — use (nearby) seeds to form local alignments called HSPs

- Extend seeds in both directions.
- No gaps allowed
- Stop when alignment score drops below threshold T
- Extended seeds are called HSPs (high scoring pairs)

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST seed extension

3. Extend — use (nearby) seeds to form local alignments called HSPs

HSP
(high-scoring pair)

Q ACTGA

Word List

ACT
CTG
TGA

D GACTGC

GACTGC
ACT
CTG

→

GACTGC
ACTG

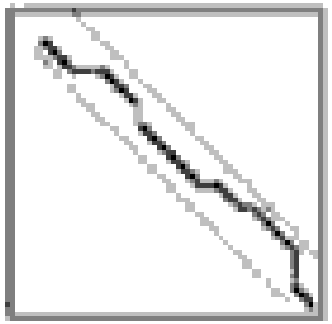
Seeds?

ACT
CTG

Parallel computing for bioinformatics and computational biology, Wu and Tseng, Editor: Zomaya

BLAST scoring

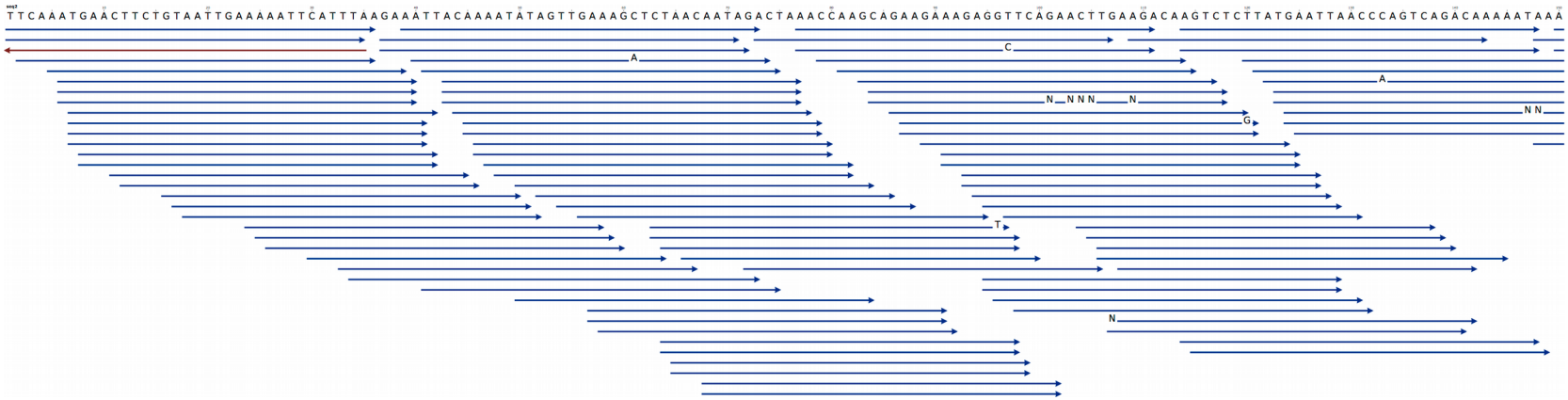
4. Score—combine groups of consistent HSPs into local alignment with best score



- Combine consistent HSPs
  - They can't overlap
  - They must be in same order in Q and D sequences
  - These "consistent local aligments" are assessed for statistical significance using an analytical null model

Next-generation sequencing creates a new alignment problem

- Align short reads to a large genome



Short-read aligners to a reference genome

- Need to be much faster than BLAST
- Reads or reference sequence or both can be indexed
- Types of indexes
  - Hash tables
  - Suffix arrays
  - Suffix trees
  - Burroughs-Wheeler Transform

Next-generation sequencing alignment algorithms

- If you are interested, check out
  - Ben Langmead Coursera videos: https://www.youtube.com/watch?v=IzXQVwWYFv4&index=9&list=PL2mpR0RYFQsBiCWVJSvVAO3OJ2t7DzoHA
  - Steven Salzberg lecture posted on Class Schedule website