CS 466
Introduction to Bioinformatics
Lecture 3

Mohammed El-Kebir

September 4, 2019

Outline

1. Edit distance recap
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.6, 6.8 and 6.9
- Lecture notes

Weighted Edit Distance – Practice Problem

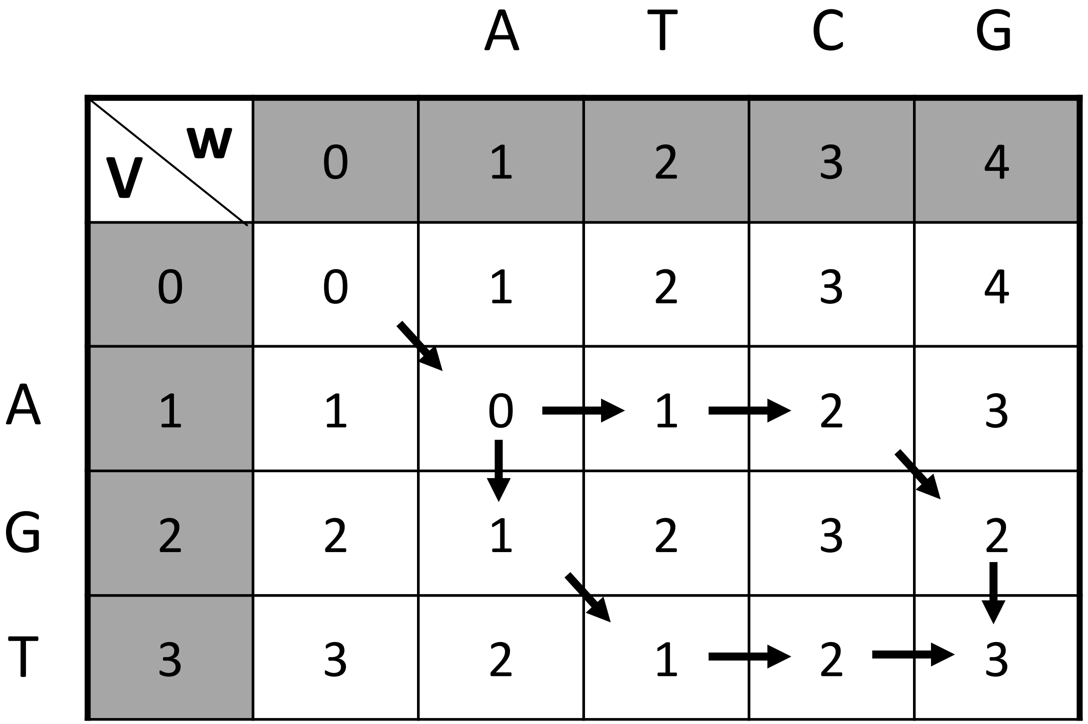
- Compute weighted edit distance between $\mathbf{v} = \mathrm{AGT}$ and $\mathbf{w} = \mathrm{ATCT}$.

		A	T	C	G
	V \ **w**	0	1	2	3	4
	0					
A	1					
G	2					
T	3					

$$d[i,j] = \min \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ d[i-1,j] + 1, & \text{if } i > 0, \\ d[i,j-1] + 1, & \text{if } j > 0, \\ d[i-1,j-1] + 2, & \text{if } i > 0,\ j > 0 \text{ and } v_i \neq w_j, \\ d[i-1,j-1], & \text{if } i > 0,\ j > 0 \text{ and } v_i = w_j. \end{cases}$$

Weighted Edit Distance – Practice Problem

- Compute weighted edit distance between $\mathbf{v} = \mathrm{AGT}$ and $\mathbf{w} = \mathrm{ATCT}$.



			A	T	C	G
	V \ w	0	1	2	3	4
	0	0	1	2	3	4
A	1	1	0	1	2	3
G	2	2	1	2	3	2
T	3	3	2	1	2	3

$$d[i,j] = \min \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ d[i-1,j]+1, & \text{if } i > 0, \\ d[i,j-1]+1, & \text{if } j > 0, \\ d[i-1,j-1]+2, & \text{if } i > 0,\ j > 0 \text{ and } v_i \neq w_j, \\ d[i-1,j-1], & \text{if } i > 0,\ j > 0 \text{ and } v_i = w_j. \end{cases}$$

Edit Distance – Additional Insights

- An alignment corresponds to a series of elementary operations

Example

```
T-ACAT-
TGAT-AT
```

$$\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT} \xrightarrow{\text{del}} \text{TGATT} \xrightarrow{\text{subst}} \text{TGATA} \xrightarrow{\text{ins}} \text{TGATAT}$$

Examples from http://profs.scienze.univr.it/~liptak/ACB/files/StringDistance_6up.pdf

Edit Distance – Additional Insights

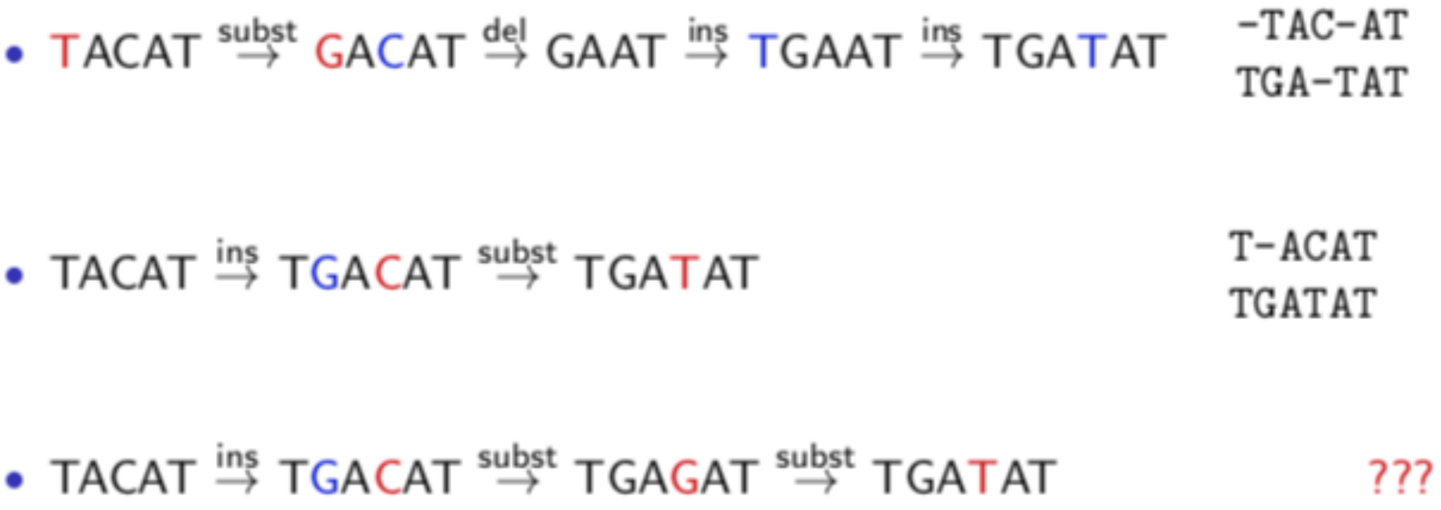
- An alignment corresponds to a series of elementary operations

Example

```
T-ACAT-
TGAT-AT
```

$\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT} \xrightarrow{\text{del}} \text{TGATT} \xrightarrow{\text{subst}} \text{TGATA} \xrightarrow{\text{ins}} \text{TGATAT}$

- But not every series of elementary operations corresponds to an alignment! Why?



- $\text{TACAT} \xrightarrow{\text{subst}} \text{GACAT} \xrightarrow{\text{del}} \text{GAAT} \xrightarrow{\text{ins}} \text{TGAAT} \xrightarrow{\text{ins}} \text{TGATAT}$

```
-TAC-AT
TGA-TAT
```

- $\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT}$

```
T-ACAT
TGATAT
```

- $\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGAGAT} \xrightarrow{\text{subst}} \text{TGATAT}$???

Examples from http://profs.scienze.univr.it/~liptak/ACB/files/StringDistance_6up.pdf

Distance Function / Metric

A **distance function** (metric) on a set X is a function $d : X \times X \to \mathbb{R}$ s.t. for all $x, y, z \in X$:

i. $d(x, y) \geq 0$ [non-negativity]

ii. $d(x, y) = 0$ if and only if $x = y$ [identity of indiscernibles]

iii. $d(x, y) = d(y, x)$ [symmetry]

iv. $d(x, y) \leq d(x, z) + d(z, y)$ [triangle inequality]

Question: Is edit distance a distance function?

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

i. $d(\mathbf{v}, \mathbf{w}) \geq 0$ [non-negativity]

Edit distance is defined by an alignment. This in turn uniquely determines a series of elementary operations, each with cost either 0 (match) or 1 (otherwise). Thus, $d(\mathbf{v}, \mathbf{w}) \geq 0$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

ii. $d(\mathbf{v}, \mathbf{w}) = 0$ if and only if $\mathbf{v} = \mathbf{w}$ [identity of indiscernibles]

(=>) By the premise, $d(\mathbf{v}, \mathbf{w}) = 0$. By definition, the optimal alignment can only consist of operations with cost 0. That is, the alignment consist of only matches. Thus, $\mathbf{v} = \mathbf{w}$.

(<=) By the premise, $\mathbf{v} = \mathbf{w}$. This means that $|\mathbf{v}| = |\mathbf{w}|$ and each letter v_i equals w_i (where $i \in [|\mathbf{v}|]$). Moreover, only the match operations has minimum cost of 0, the other operations have cost 1. Hence, this is the optimal alignment with cost $d(\mathbf{v}, \mathbf{w}) = 0$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

iii. $d(\mathbf{v}, \mathbf{w}) = d(\mathbf{w}, \mathbf{v})$ [symmetry]

Let $\mathbf{A} = [a_{i,j}]$ be the optimal alignment corresponding to $d(\mathbf{v}, \mathbf{w})$, i.e. $\mathbf{A}$ is an $2 \times k$ matrix where $k \in \{\max(|\mathbf{v}|, |\mathbf{w}|), \dots, |\mathbf{v}| + |\mathbf{w}|\}$. Define the function $f(\mathbf{A}) = \mathbf{B}$ such that $\mathbf{B}$ is obtained by interchanging the two rows of $\mathbf{A}$. Since the cost of any insertion, deletion and mismatch is 1, we have that alignment $\mathbf{B}$ has cost $d(\mathbf{v}, \mathbf{w})$. The existence of an alignment from $\mathbf{w}$ to $\mathbf{v}$ with cost less than $d(\mathbf{v}, \mathbf{w})$, yields a contradiction as it implies that $\mathbf{A}$ is not an optimal alignment from $\mathbf{v}$ to $\mathbf{w}$. Hence, $d(\mathbf{w}, \mathbf{v}) = d(\mathbf{v}, \mathbf{w})$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

iv. $d(\mathbf{v}, \mathbf{w}) \leq d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$ [triangle inequality]

Assume for a contradiction that $d(\mathbf{v}, \mathbf{w}) > d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$. Let S be the sequence of elementary operations for transforming $\mathbf{v}$ into $\mathbf{u}$. Let S' be the sequence of elementary operations for transforming $\mathbf{u}$ into $\mathbf{w}$. Note that $d(\mathbf{v}, \mathbf{u}) = |S|$ and $d(\mathbf{u}, \mathbf{w}) = |S'|$. Concatenate S and S' and remove redundant operations, yielding sequence S''. By definition, $|S''| \leq |S| + |S'|$. We can obtain an alignment of $\mathbf{v}$ and $\mathbf{w}$ from S'' with cost $|S''| \leq d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$. This yields a contradiction with $d(\mathbf{v}, \mathbf{w}) > d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$ being the cost of the optimal alignment of $\mathbf{v}$ and $\mathbf{w}$.

Outline

1. Edit distance recap
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

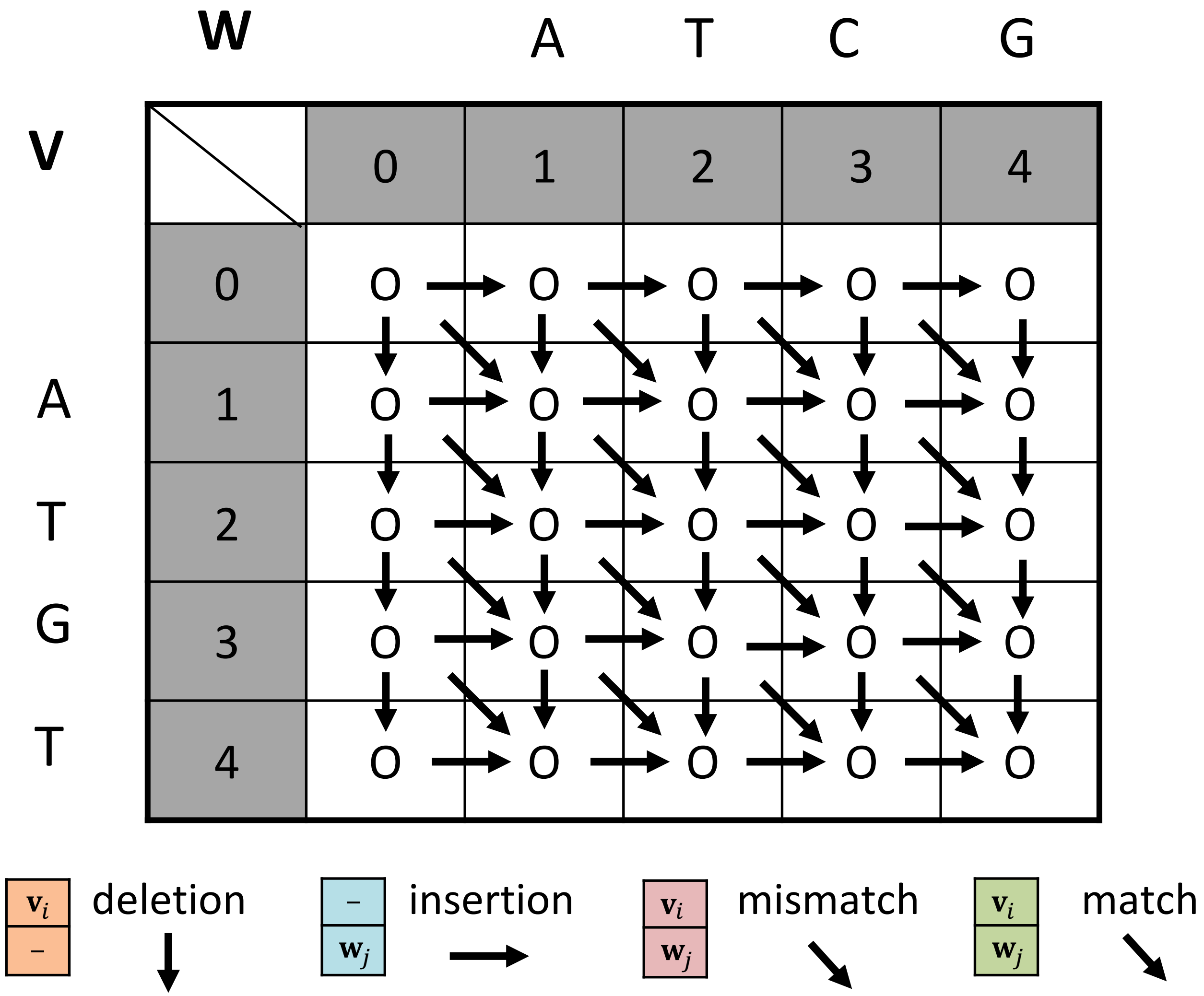
Reading:

- Jones and Pevzner. Chapters 6.6, 6.8 and 6.9

Biological Sequence Alignment

- Weighted edit distance: find alignment with minimum distance
  - Shortest path in weighted edit graph
- Sequence alignment: find alignment with maximum similarity
  - Longest path in weighted edit graph
  - Score function: $\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$

Question: What is an example of δ?



$\delta(v_i, -)$ $\delta(-, w_j)$ $\delta(v_i, w_j)$

Scoring Matrices

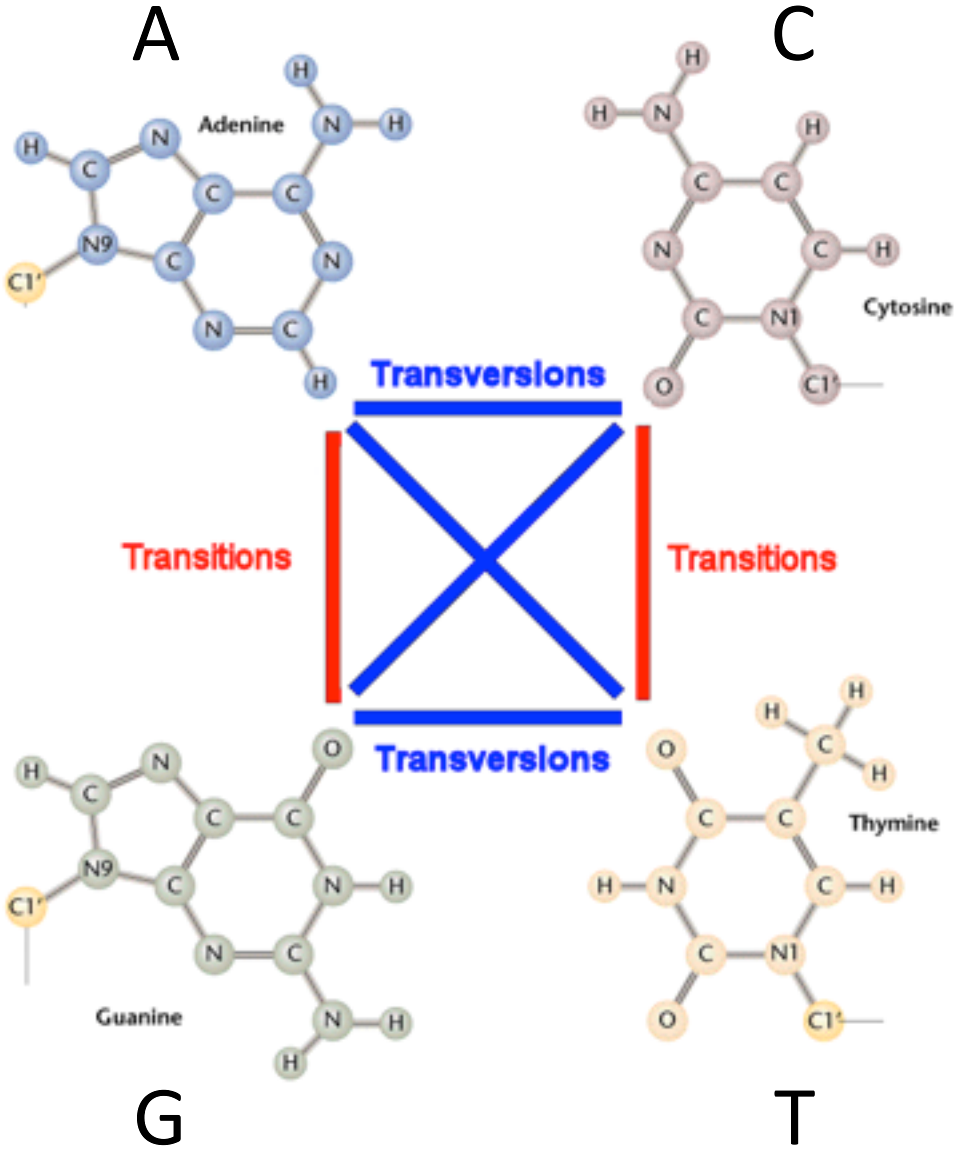
Transitions: interchanges among purines (two rings) or pyrimidines (one ring)

- A <--> G
- C <--> T

Transversions: interchanges between purines (two rings) and pyrimidines (one ring)

- A <--> C, A <--> T
- G <--> C, G <--> T

Transitions more likely than transversions!



Scoring Matrices

Transitions: interchanges among purines (two rings) or pyrimidines (one ring)

- A <--> G
- C <--> T

Transversions: interchanges between purines (two rings) and pyrimidines (one ring)

- A <--> C, A <--> T
- G <--> C, G <--> T

Transitions more likely than transversions!

δ	A	T	C	G	-
A	1	-2	-2	-1	-1
T	-2	1	-1	-2	-1
C	-2	-1	1	-2	-1
G	-1	-2	-2	1	-1
-	-1	-1	-1	-1	$-\infty$

Global Alignment – Needleman-Wunsch Algorithm

Global Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find alignment with maximum score.

- An alignment is a source-to-sink path in the edit graph
- An alignment $\mathbf{A} = [a_{i,j}]$ is a $2 \times k$ matrix s.t. (i) $k = \{\max(m,n), \dots, m+n\}$, (ii) $a_{i,j} \in \Sigma \cup \{-\}$ and (iii) there is no $j \in [k]$ where $a_{1,j} = a_{2,j} = -$

$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \qquad\qquad \text{deletion} \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \qquad\qquad \text{insertion} \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \qquad \text{match/mismatch} \end{cases}$$

Demonstration

- http://alfehrest.org/sub/nwa/index.html

- $\mathbf{v} = \mathrm{ATGTTAT}$ and $\mathbf{w} = \mathrm{ATCGTAC}$.

δ	A	T	C	G	-
A	1	-2	-2	-1	-1
T	-2	1	-1	-2	-1
C	-2	-1	1	-2	-1
G	-1	-2	-2	1	-1
-	-1	-1	-1	-1	$-\infty$

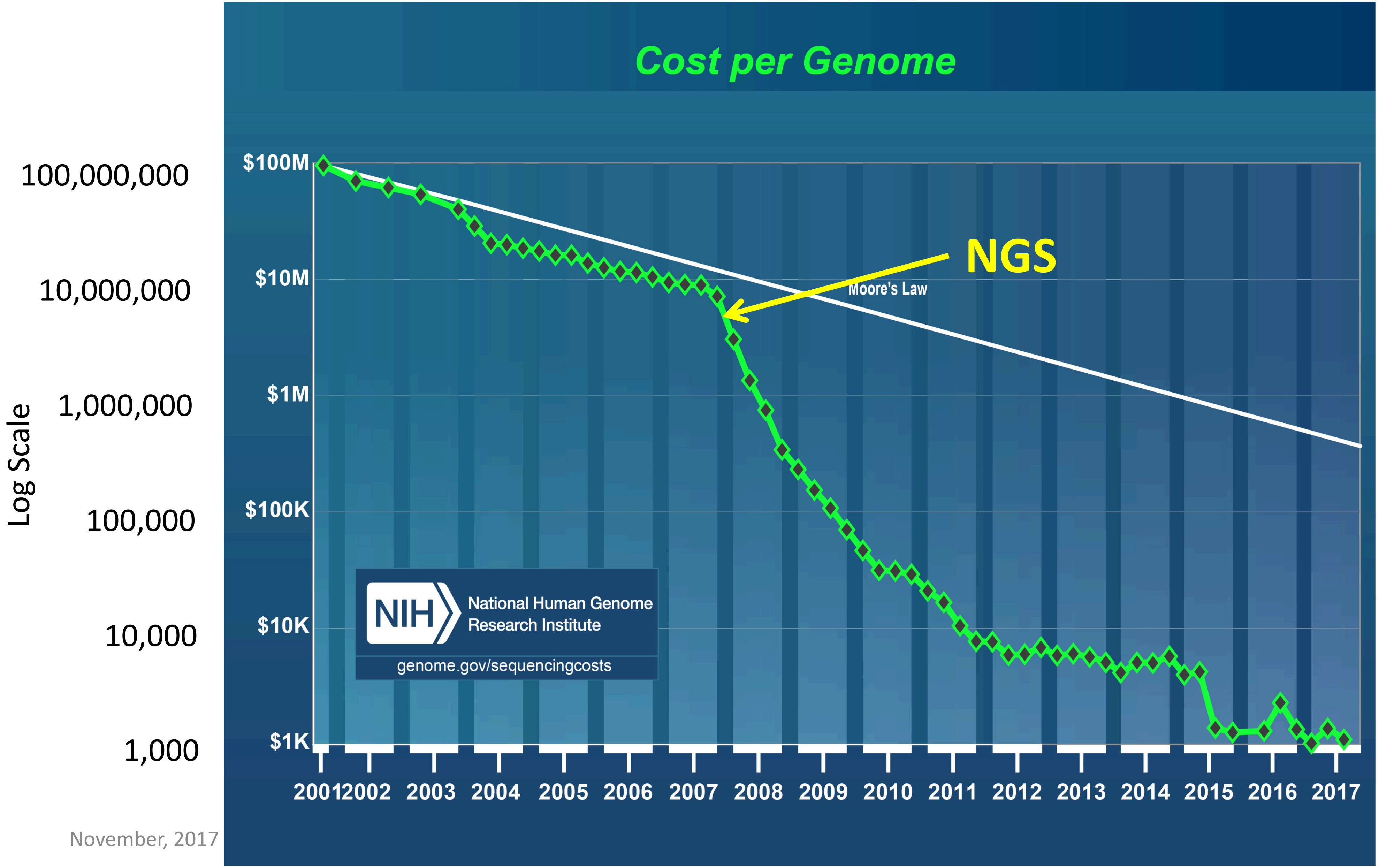
Outline

1. Edit distance recap
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

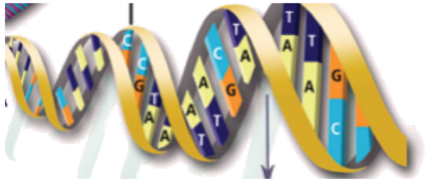
Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes

Next Generation Sequencing (NGS) Technology



NGS Characterized by Short Reads



Genome
Millions -billions nucleotides

Next-generation DNA sequencing

… CATTCAGTAG …

… AGCCATTAG …

… GGTAGTTAG …

… GGTAAACTAG …

… TATAATTAG …

… CGTACCTAG …

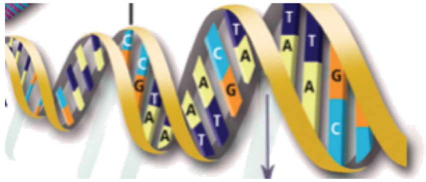
10-100’s million ***short reads***

Short read*:* 100 nucleotides

Allow for inexact matches due to:

- Sequencing errors
- Polymorphisms/mutations in reference genome

NGS Characterized by Short Reads



Genome
Millions -billions nucleotides

Next-generation DNA sequencing

… CATTCAGTAG …

… AGCCATTAG …

… GGTAGTTAG …

… GGTAAACTAG …

… TATAATTAG …

… CGTACCTAG …

10-100’s million ***short reads***

Short read*:* 100 nucleotides

Allow for inexact matches due to:

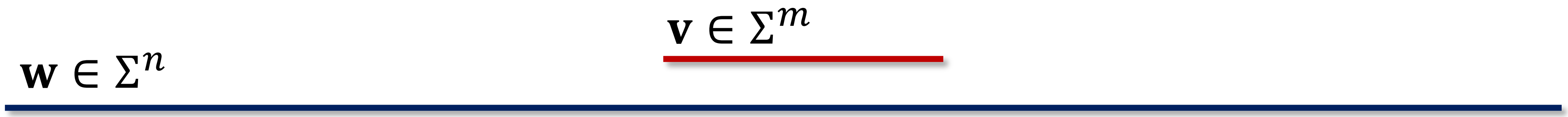
- Sequencing errors
- Polymorphisms/mutations in reference genome

Human reference genome is 3,300,000,000 nucleotides, while a short read is 100 nucleotides. Global sequence alignment will not work!

Question: How to account for discrepancy between lengths of reference and short read?

Fitting Alignment

For short read alignment, we want to align complete short read $\mathbf{v} \in \Sigma^m$ to substring of reference genome $\mathbf{w} \in \Sigma^n$. Note that $m \ll n$.



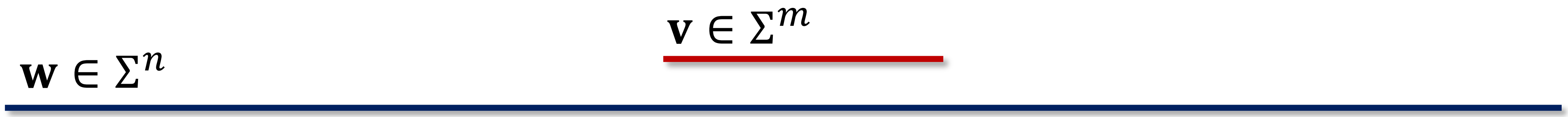
$\mathbf{v} \in \Sigma^m$

$\mathbf{w} \in \Sigma^n$

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

Fitting Alignment – Naive Approach

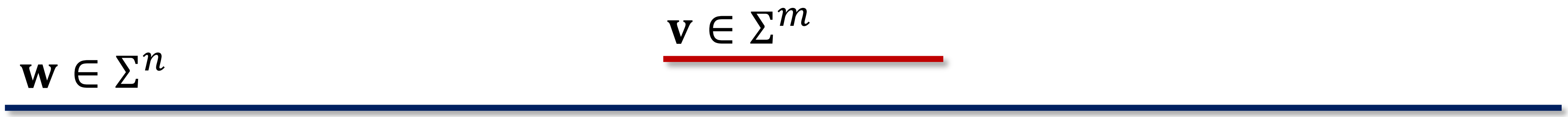
Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$



- Consider all contiguous non-empty substrings of $\mathbf{w}$, defined by $1 \leq i \leq j \leq n$
- How many?

Fitting Alignment – Naive Approach

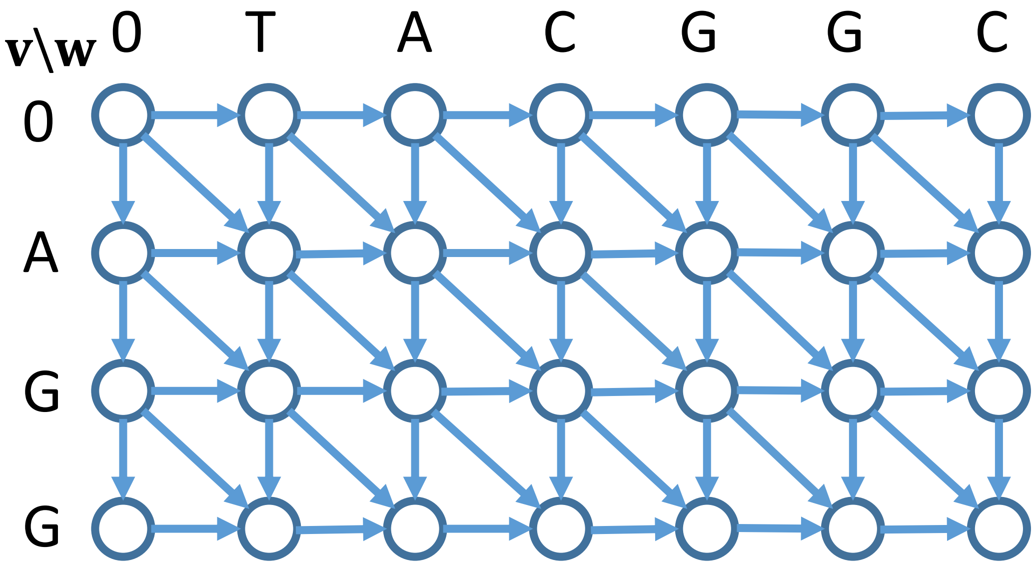
Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$



- Consider all contiguous non-empty substrings of $\mathbf{w}$, defined by $1 \leq i \leq j \leq n$
- How many? Answer: $n + \binom{n}{2}$
- What are their total lengths?
- What is the running time?

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

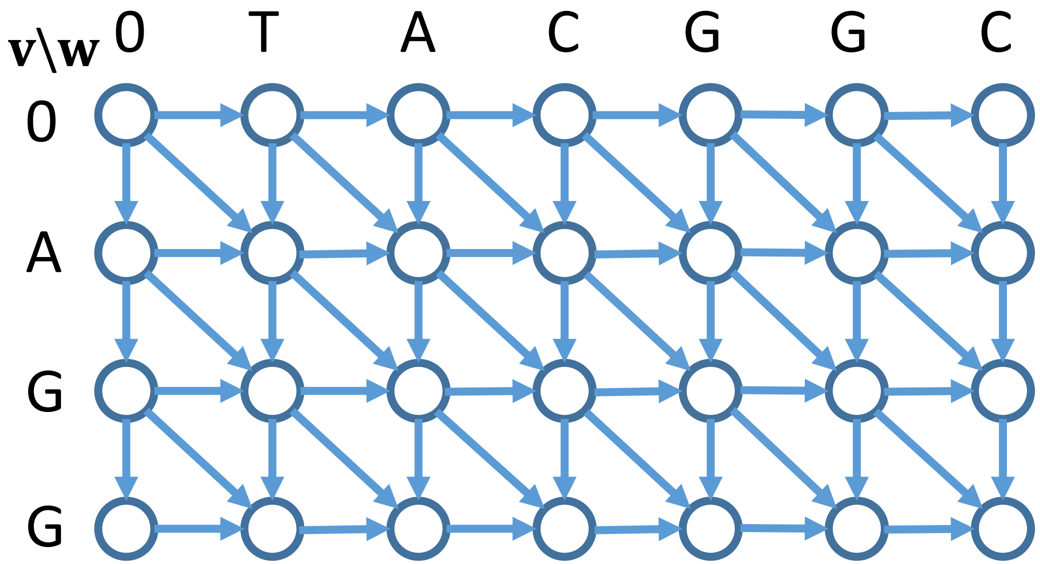


$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max\{s[m,0], \ldots, s[m,n]\}$$

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

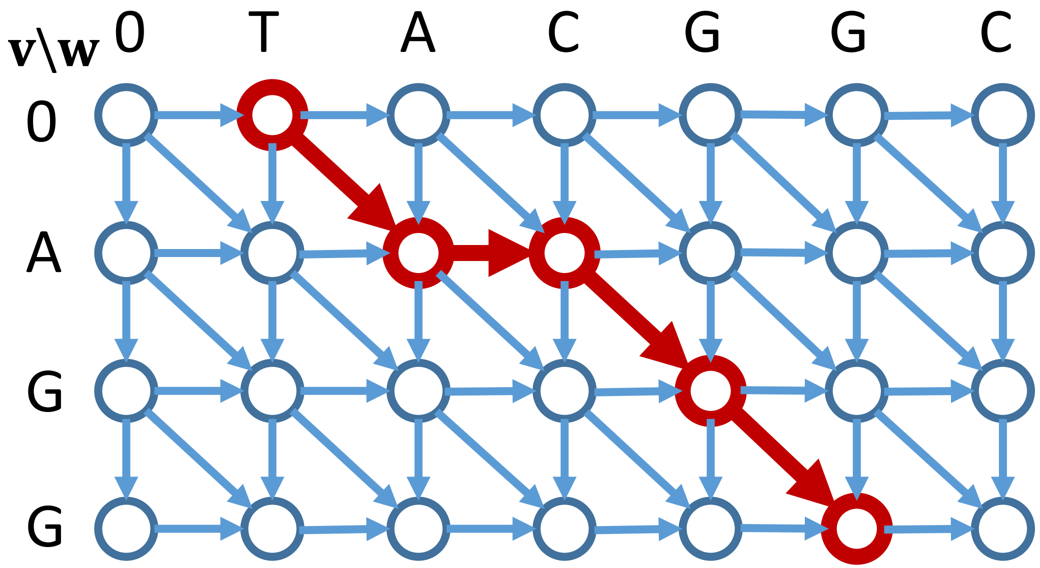


$$s[i,j] = \max \begin{cases} \boxed{0, \ \textbf{Start anywhere on first row} \ \text{if } i = 0,} \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \boxed{\max\{s[m,0], \ldots, s[m,n]\} \quad \textbf{End anywhere on last row}}$$

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$



v	-	A	-	G	G	-
w	T	A	C	G	G	C

$$s[i,j] = \max \begin{cases} 0, \ \text{Start anywhere on first row} & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

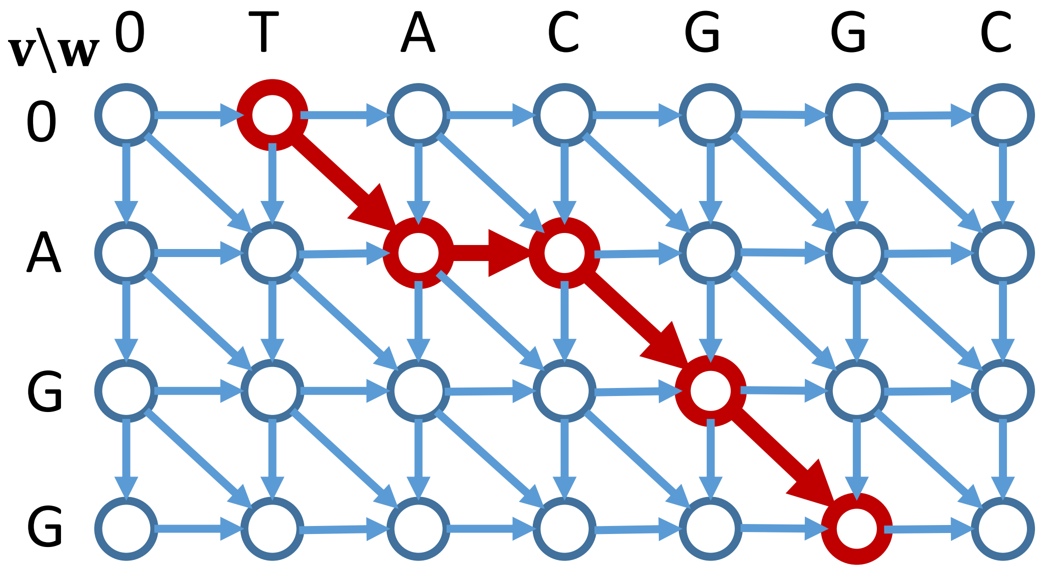
$$s^* = \max\{s[m,0], \ldots, s[m,n]\} \quad \text{End anywhere on last row}$$

Question: Let match score be 1, mismatch/indel score be -1. What is s^*?

Question: Same scores. What is optimal global alignment and score?

Fitting Alignment – Dynamic Programming

- Online:
  https://valiec.github.io/AlignmentVisualizer/index.html



$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max\{s[m,0], \ldots, s[m,n]\}$$

v	–	A	–	G	G	–
w	T	A	C	G	G	C

Question: Let match score be 1, mismatch/indel score be -1. What is s^*?

Question: Same scores. What is optimal global alignment and score?

Outline

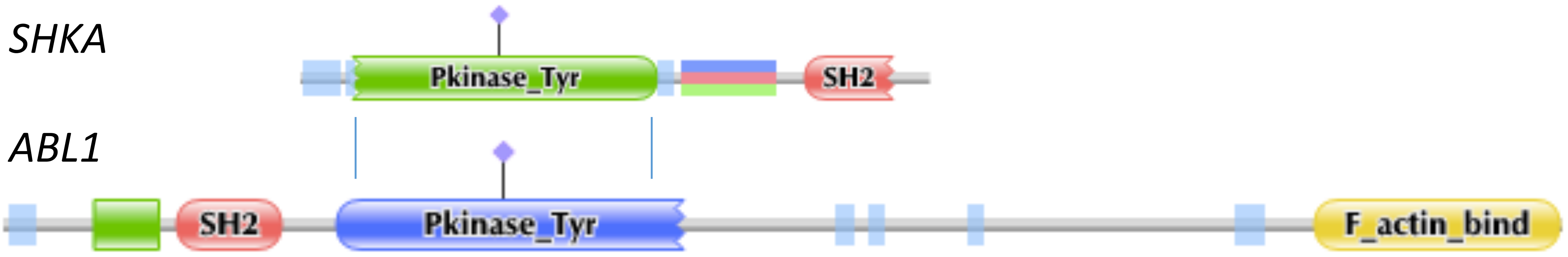
1. Edit distance
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.6, 6.8 and 6.9
- Lecture notes

Local Alignment – Biological Motivation

Proteins are composed of functional units called domains. Such domains may occur in different proteins even across species.



From Pfam database (http://pfam.sanger.ac.uk/)

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

Global, Fitting and Local Alignment

Global Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find alignment of $\mathbf{v}$ and $\mathbf{w}$ with maximum score.

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

Local Alignment – Naive Algorithm

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

Brute force:

1. Generate all pairs $(\mathbf{v}', \mathbf{w}')$ of substrings of $\mathbf{v}$ and $\mathbf{w}$
2. For each pair $(\mathbf{v}', \mathbf{w}')$, solve global alignment problem.

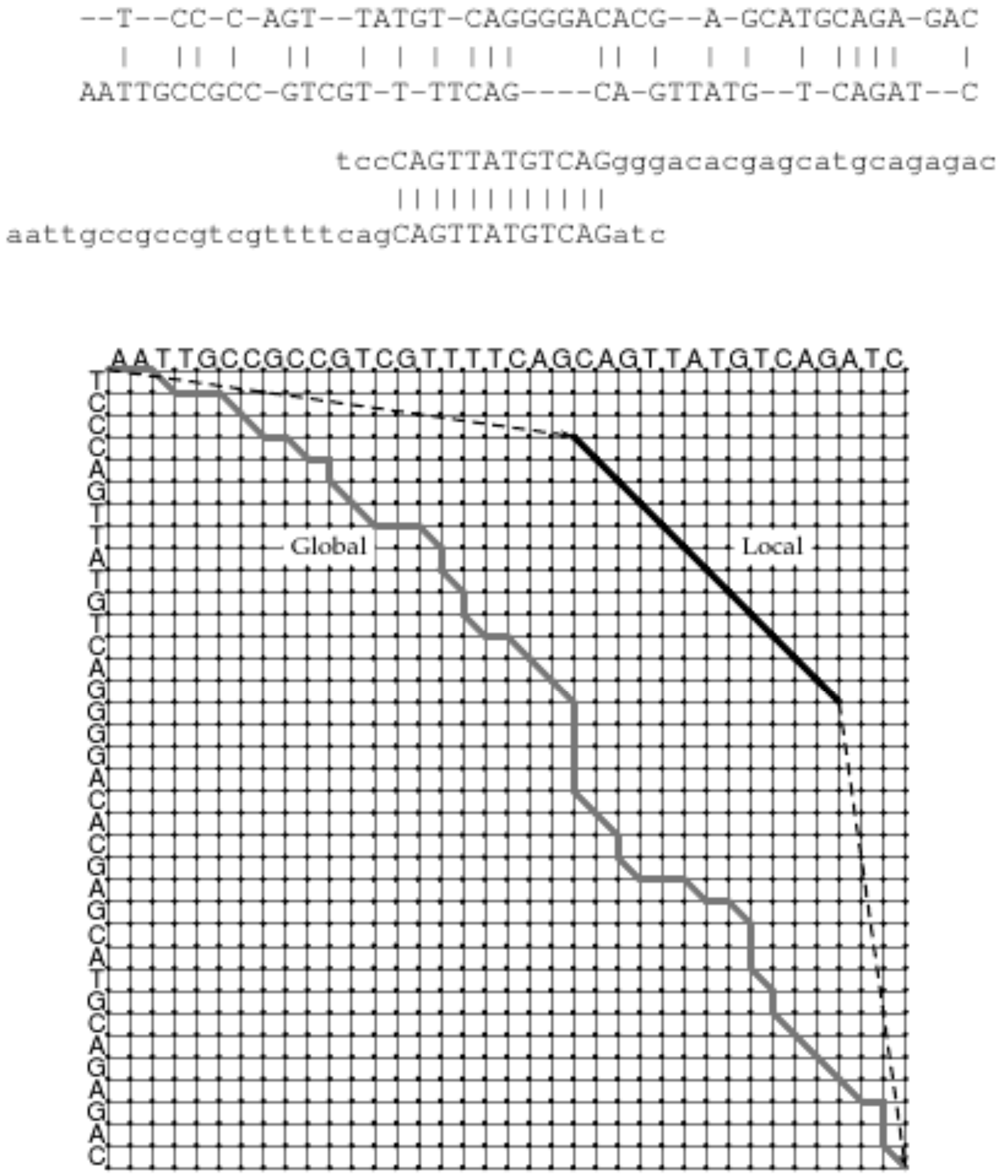
Question: There are $\binom{m}{2}\binom{n}{2}$ pairs of substrings.

But they have different lengths. What is the running time?

Key Idea

Global alignment:
- Start at $(0,0)$ and end at (m,n)

Local alignment:
- Start and end anywhere



```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C

                  tccCAGTTATGTCAGgggacacgagcatgcagagac
                     ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment Recurrence

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C
```

```
                  tccCAGTTATGTCAGggacacgagcatgcagagac
                     ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

$$s[i,j] = \max \begin{cases} 0, & \cancel{\text{if } i = 0 \text{ and } j = 0,} \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max_{i,j} s[i,j]$$

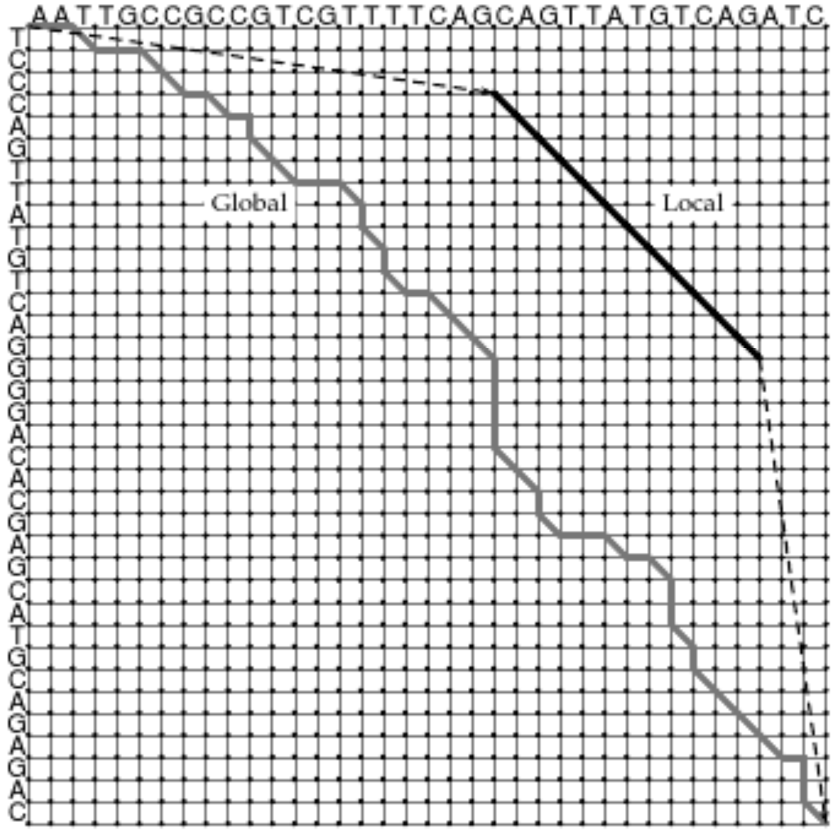


Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment Recurrence

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

$$s[i,j] = \max \begin{cases} 0, \quad \textbf{Start anywhere} & \cancel{\text{if } i = 0 \text{ and } j = 0,} \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max_{i,j} s[i,j] \quad \textbf{End anywhere}$$

Running time: $O(mn)$

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C

                  tccCAGTTATGTCAGgggacacgagcatgcagagac
                     ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

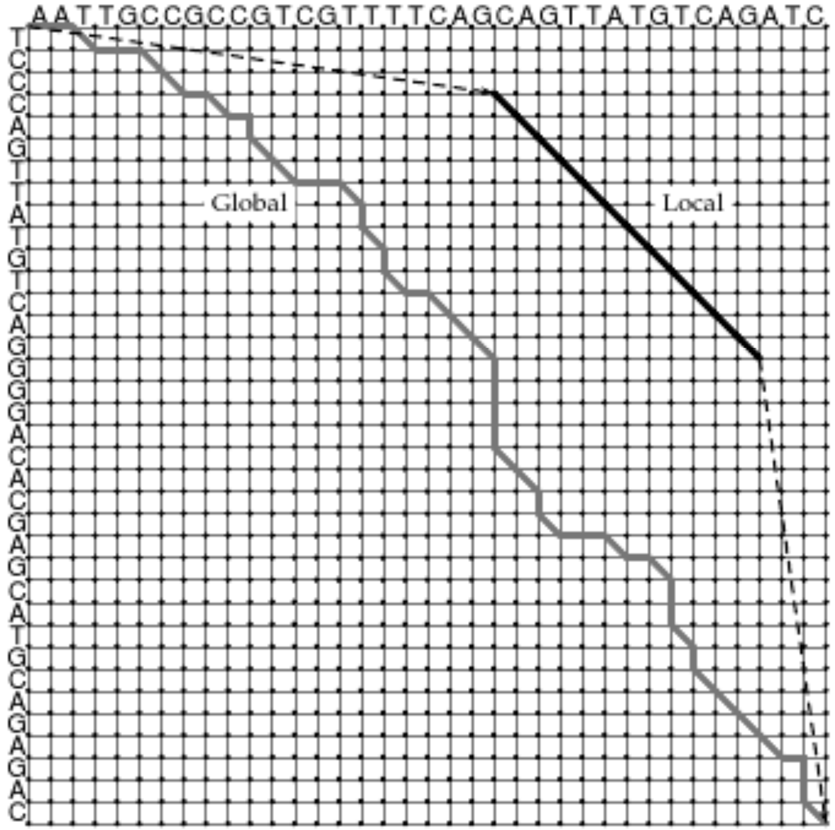
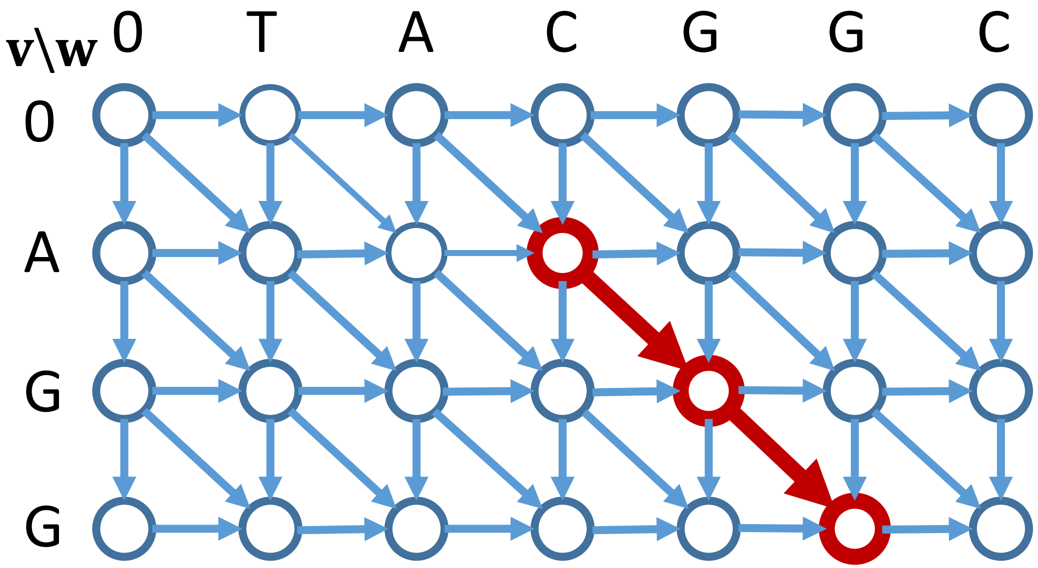


Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment – Dynamic Programming

- Online: https://valiec.github.io/AlignmentVisualizer/index.html



V	G	G
W	G	G

$$s[i,j] = \max \begin{cases} 0, & \cancel{\text{if } i = 0 \text{ and } j = 0,} \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max_{i,j} s[i,j]$$

Question: Let match score be 2, mismatch score be -2 and indel be -4. What is s^*?

Global, Fitting and Local Alignment

Global Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find alignment of $\mathbf{v}$ and $\mathbf{w}$ with maximum score.

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

Local Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a substring of $\mathbf{v}$ and a substring of $\mathbf{w}$ whose alignment has maximum global alignment score s^* among *all* global alignments of *all* substrings of $\mathbf{v}$ and $\mathbf{w}$

Outline

1. Edit distance
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

Reading:

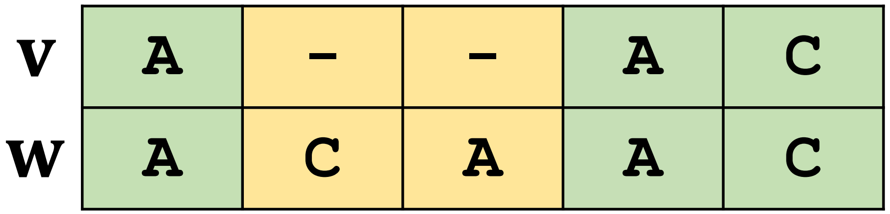
- Jones and Pevzner. Chapters 6.6, 6.8 and 6.9
- Lecture notes

Scoring Gaps

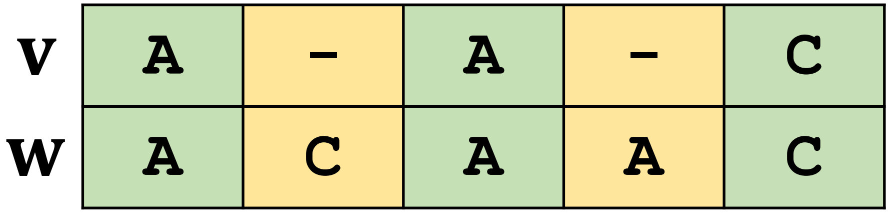
Let $\mathbf{v} = \mathrm{AAC}$ and $\mathbf{w} = \mathrm{ACAGGC}$

Match $\delta(c, c) = 1$;

Mismatch $\delta(c, d) = -1$ (where $c \neq d$); Indel $\delta(c, -) = \delta(-, c) = -2$



v	A	-	-	A	C
w	A	C	A	A	C



v	A	-	A	-	C
w	A	C	A	A	C

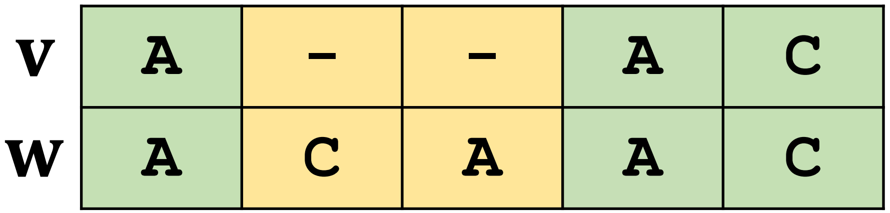
Both alignments have 3 matches and 2 indels.

Score: $(3 * 1) + (2 * -2) = -1$

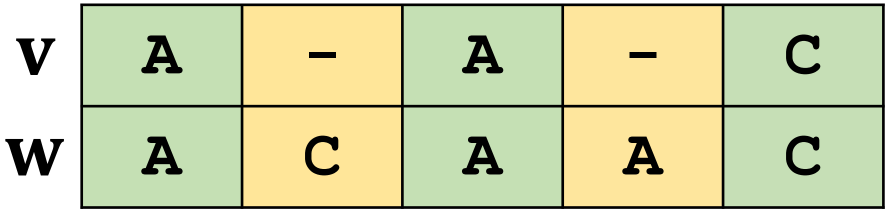
Scoring Gaps

Let $\mathbf{v} = \text{AAC}$ and $\mathbf{w} = \text{ACAGGC}$

Match $\delta(c, c) = 1$;
Mismatch $\delta(c, d) = -1$ (where $c \neq d$); Indel $\delta(c, -) = \delta(-, c) = -2$



v	A	-	-	A	C
w	A	C	A	A	C



v	A	-	A	-	C
w	A	C	A	A	C

Both alignments have 3 matches and 2 indels.
Score: $(3 * 1) + (2 * -2) = -1$

Question: Which alignment is better?

Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.

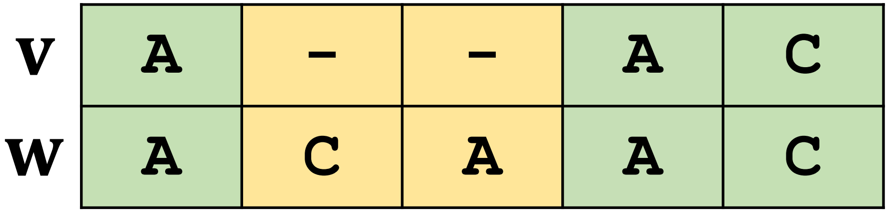
v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

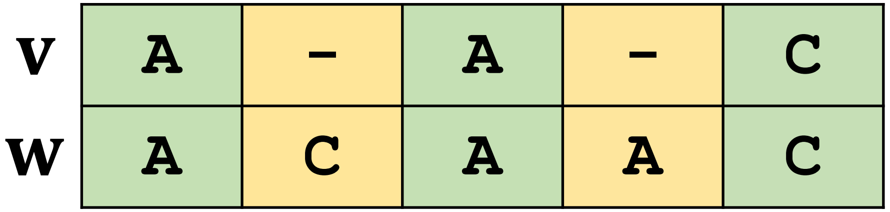
Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.



v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**



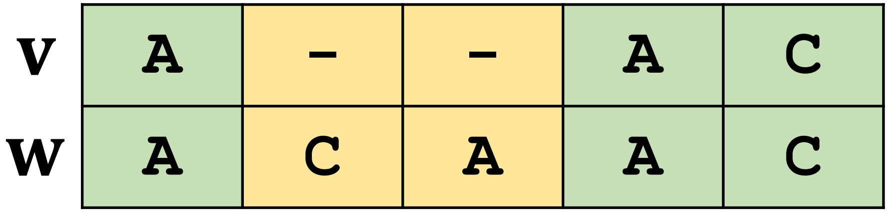
v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.

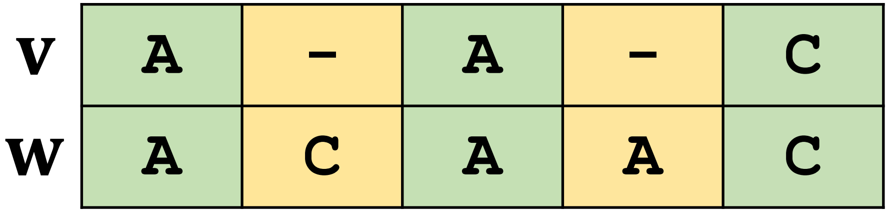
Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.



v	A	-	-	A	C
w	A	C	A	A	C



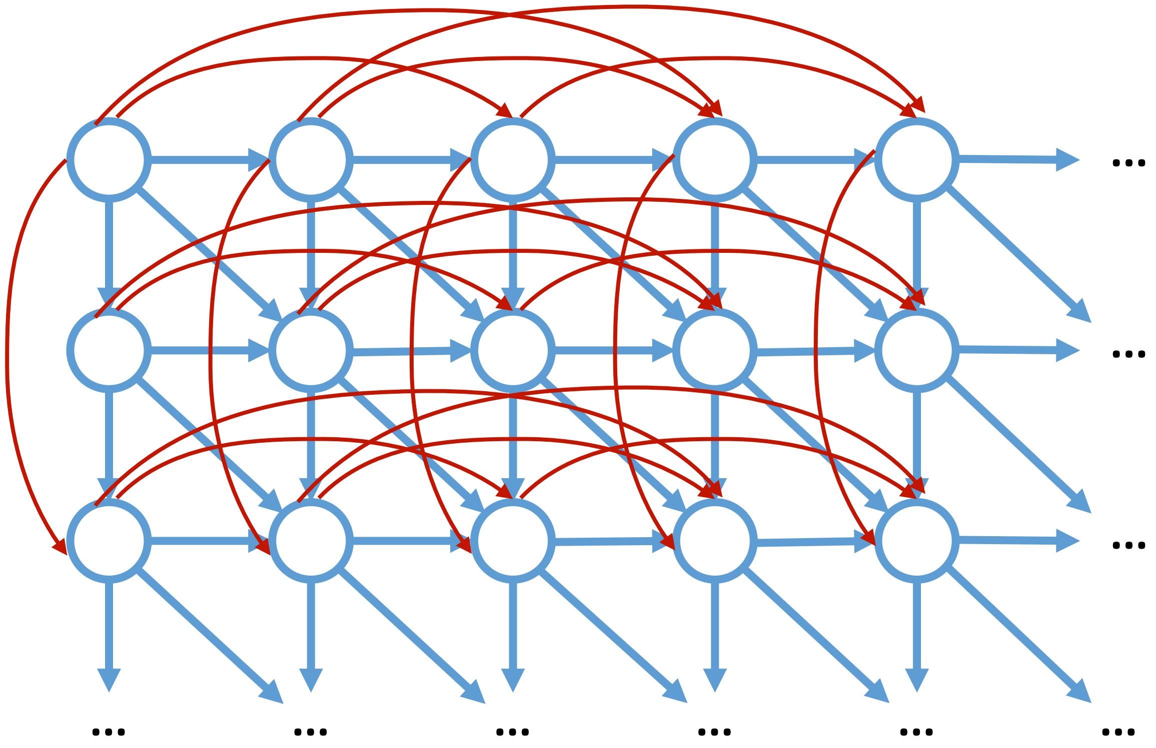
v	A	-	A	-	C
w	A	C	A	A	C

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.

Let $\rho = 10$ and $\sigma = 1$. Left: $(3 * 1) - (10 + 1 * 2) = -9$.
Right: $(3 * 1) - (10 + 1 * 1) - (10 + 1 * 1) = -19$.

Affine Gap Penalty Alignment – Naive Approach

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.



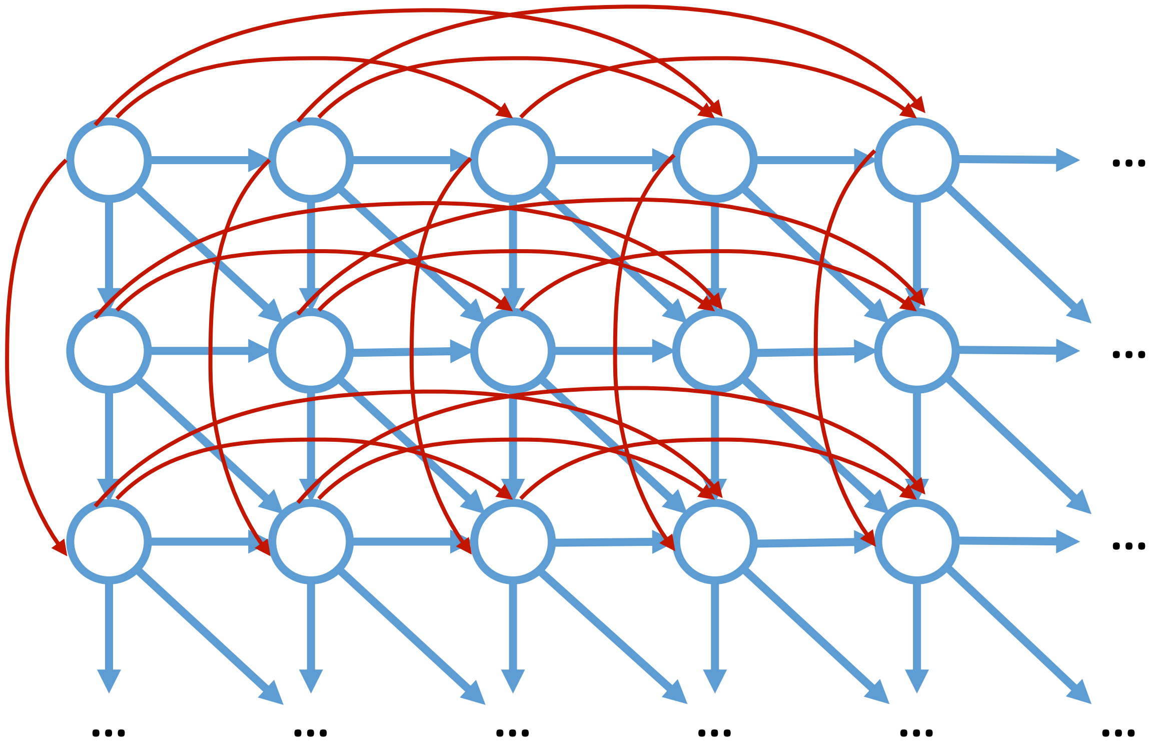
Idea: Insert horizontal (deletion) and vertical (insertion) edges spanning $k > 1$ gaps with score $-(\rho + \sigma k)$.

→ new edges

↷ old edges

Affine Gap Penalty Alignment – Naive Approach

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.



Idea: Insert horizontal (deletion) and vertical (insertion) edges spanning $k > 1$ gaps with score $-(\rho + \sigma k)$.

→ new edges

↷ old edges

Question: What's the recurrence?

Question: What's the running time?

Affine Gap Penalty Alignment

Idea: Three separate recurrences:

(i) Gap in first sequence $s^{\rightarrow}[i,j]$

(ii) Match/mismatch $s^{\searrow}[i,j]$

(iii) Gap in second sequence $s^{\downarrow}[i,j]$

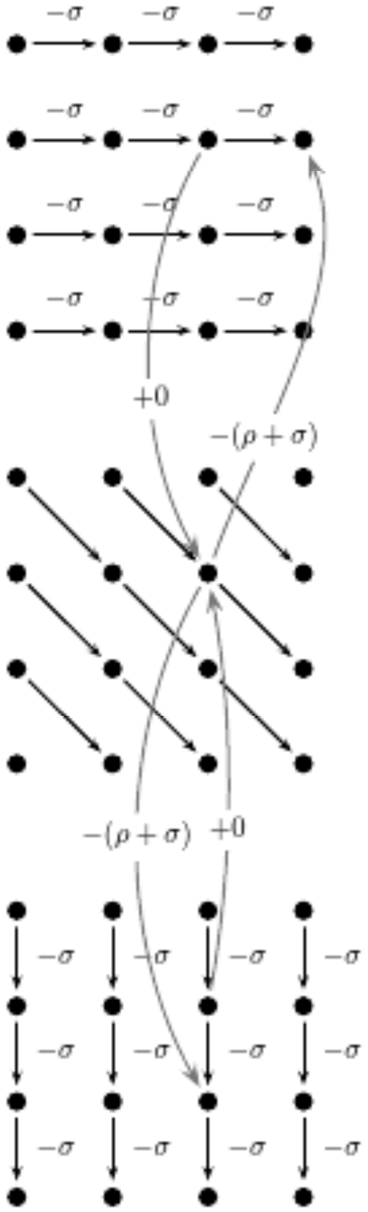


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i,j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment

Idea: Three separate recurrences:
(i) Gap in first sequence $s^{\rightarrow}[i,j]$
(ii) Match/mismatch $s^{\searrow}[i,j]$
(iii) Gap in second sequence $s^{\downarrow}[i,j]$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1]-\sigma, & \text{if } j>1, \\ s^{\searrow}[i,j-1]-(\sigma+\rho), & \text{if } j>0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i=0 \text{ and } j=0, \\ s^{\rightarrow}[i,j], & \text{if } j>0, \\ s^{\downarrow}[i,j], & \text{if } i>0, \\ s^{\searrow}[i-1,j-1]+\delta(v_i,w_j), & \text{if } i>0 \text{ and } j>0, \end{cases}$$

$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j]-\sigma, & \text{if } i>1, \\ s^{\searrow}[i-1,j]-(\sigma+\rho), & \text{if } i>0. \end{cases}$$

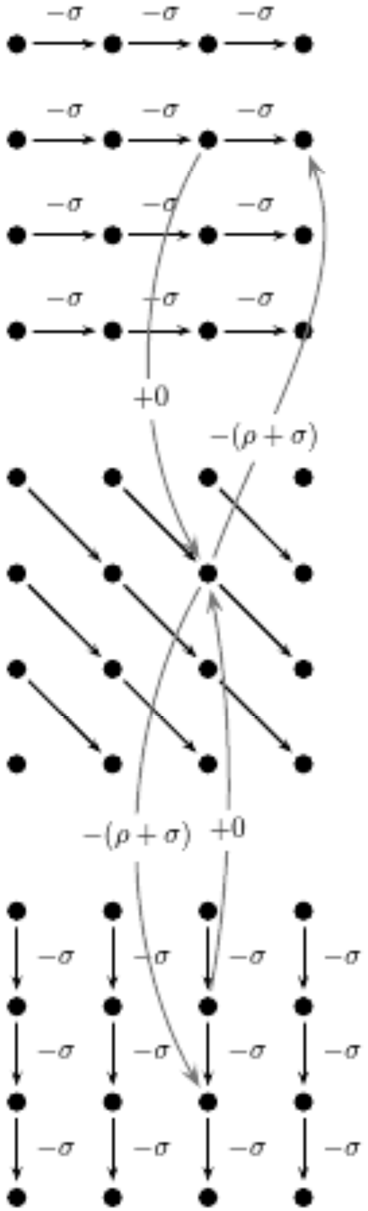


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i, j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment

Idea: Three separate recurrences:
(i) Gap in first sequence $s^{\rightarrow}[i, j]$
(ii) Match/mismatch $s^{\searrow}[i, j]$
(iii) Gap in second sequence $s^{\downarrow}[i, j]$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1] - \sigma, & \text{if } j > 1, \\ s^{\searrow}[i,j-1] - (\sigma + \rho), & \text{if } j > 0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s^{\rightarrow}[i,j], & \text{if } j > 0, \\ s^{\downarrow}[i,j], & \text{if } i > 0, \\ s^{\searrow}[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0, \end{cases}$$

$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j] - \sigma, & \text{if } i > 1, \\ s^{\searrow}[i-1,j] - (\sigma + \rho), & \text{if } i > 0. \end{cases}$$

Running time: $O(mn)$

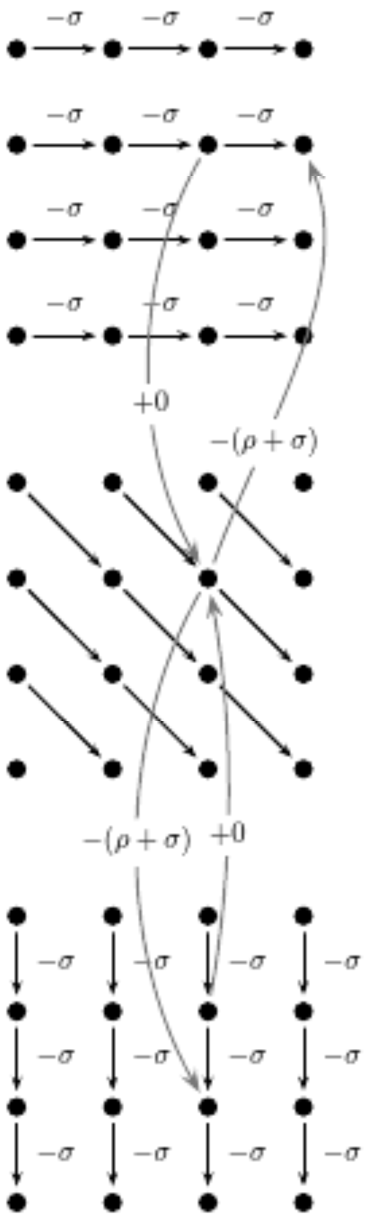


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i, j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment – Example

Let $\rho = 10$ and $\sigma = 1$. Match = 1. Mismatch = -1

$\mathbf{v} = \mathrm{AAC}$ $\mathbf{w} = \mathrm{ACAAC}$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1] - \sigma, & \text{if } j > 1, \\ s^{\searrow}[i,j-1] - (\sigma + \rho), & \text{if } j > 0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s^{\rightarrow}[i,j], & \text{if } j > 0, \\ s^{\downarrow}[i,j], & \text{if } i > 0, \\ s^{\searrow}[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0, \end{cases}$$

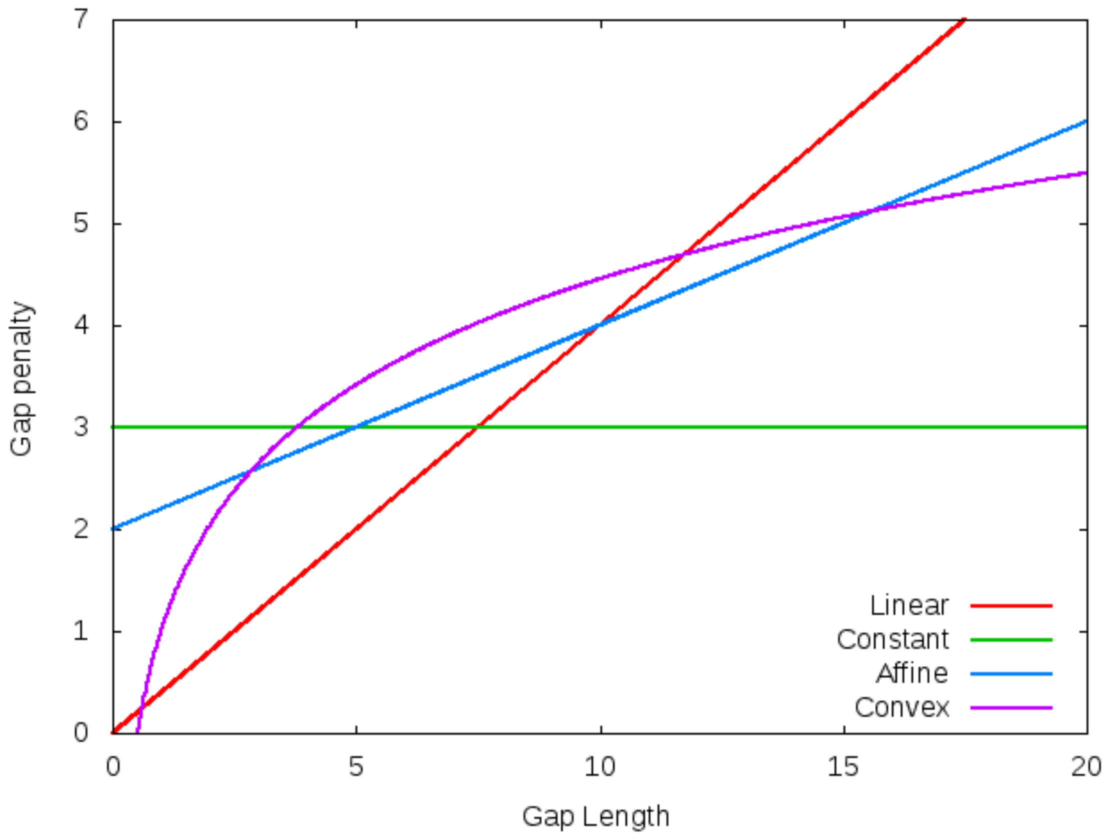
$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j] - \sigma, & \text{if } i > 1, \\ s^{\searrow}[i-1,j] - (\sigma + \rho), & \text{if } i > 0. \end{cases}$$

Gapped Alignment – Additional Insights

- Naive approach supports arbitrary gap penalties given two sequences $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$. This results in an $O(mn(m+n))$ algorithm.

- Alignment with **convex gap penalties** given two sequences $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ can be computed in $O(mn \log m)$ time.

  See: Dan Gusfield. 1997. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, New York, NY, USA.



Take Home Messages

1. Edit distance
2. Global alignment
3. Fitting alignment
4. Local alignment
5. Gapped alignment

Edit distance is shortest path in DAG

Global alignment is longest path in DAG

Small tweaks enable different extensions

Reading:

- Jones and Pevzner. Chapters 6.6, 6.8 and 6.9
- Lecture notes