Computational Genomics (0382.3102)
Lecture 3

Sequence Similarity and Pairwise Alignment II:
Affine Gap Penalties, Local Alignment,
BLAST and FASTA Heuristics

Prof. Benny Chor
School of Computer Science
Tel-Aviv University

Based in part on chapter *** in Gusfield’s book, chapter 3 in Kanehisa’s book,
and on a ppt presentation by Terry Speed (UC Berkeley)
Distances

Let $S$ be a (finite or infinite) set. A distance on $S$ is a function $D : S \times S \to \mathbb{R}^\geq 0$ satisfying the following three properties:

- Symmetry: $\forall v, w \in S, D(v, w) = D(w, v)$. 
Distances

Let $S$ be a (finite or infinite) set. A distance on $S$ is a function $D : S \times S \to \mathbb{R}^{\geq 0}$ satisfying the following three properties:

- **Symmetry:** $\forall v, w \in S$, $D(v, w) = D(w, v)$.
- **Non-negativity:** $\forall v, w \in S$, $D(v, w) \geq 0$, and $D(v, w) = 0$ if and only if $v = w$. 
Distances

Let $S$ be a (finite or infinite) set. A distance on $S$ is a function $D : S \times S \mapsto \mathbb{R}^{\geq 0}$ satisfying the following three properties:

- **Symmetry:** $\forall v, w \in S, D(v, w) = D(w, v)$.
- **Non-negativity:** $\forall v, w \in S, D(v, w) \geq 0$, and $D(v, w) = 0$ if and only if $v = w$.
- **Triangle inequality:** $\forall u, v, w \in S, D(u, w) \leq D(u, v) + D(v, w)$.
Famous Distances

A distance on $S$, $D : S \times S \mapsto \mathbb{R}^{\geq 0}$ is also called a norm in math jargon. Example of norms (for some of these it is not immediate to verify that triangle inequality holds).

- $D(v, w) = 1$ if $v \neq w$ (this norm is a bit boring).
Famous Distances

A distance on $S$, $D : S \times S \mapsto \mathbb{R}^{\geq 0}$ is also called a norm in math jargon. Example of norms (for some of these it is not immediate to verify that triangle inequality holds).

- $D(v, w) = 1$ if $v \neq w$ (this norm is a bit boring).
- Let $S = \mathbb{R}^d$ ($d$-dim. real vectors) and $p \geq 1$. $D(\langle v_1, \ldots, v_d \rangle, \langle u_1, \ldots, u_d \rangle)$

$$= \sqrt[p]{\sum_{i=1}^{d} |v_i - u_i|^p}.$$  

In math jargon, this is known as the $\ell_p$ norm.
More Famous Distances

- $\ell_p$ with $p = 2$ is the “regular” Euclidean distance.
More Famous Distances

- $\ell_p$ with $p = 2$ is the “regular” Euclidean distance.
- $\ell_p$ with $p = 1$ (sum of absolute values of differences).
More Famous Distances

- $\ell_p$ with $p = 2$ is the “regular” Euclidean distance.
- $\ell_p$ with $p = 1$ (sum of absolute values of differences).
- The $\ell_\infty$ norm: $\max_{i=1}^d |v_i - u_i|$ (the limit of $\ell_p$ as $p \to \infty$).
More Famous Distances

- $\ell_p$ with $p = 2$ is the “regular” Euclidean distance.
- $\ell_p$ with $p = 1$ (sum of absolute values of differences).
- The $\ell_\infty$ norm: $\max_{i=1}^{d} |v_i - u_i|$ (the limit of $\ell_p$ as $p \rightarrow \infty$).
- Let $G = (V, E)$ be a finite, undirected, connected graph, with positive edges’ lengths. Let $u, v \in V$ be a pair of vertices.
More Famous Distances

- $\ell_p$ with $p = 2$ is the “regular” Euclidean distance.
- $\ell_p$ with $p = 1$ (sum of absolute values of differences).
- The $\ell_\infty$ norm: $\max_{i=1}^d |v_i - w_i|$ (the limit of $\ell_p$ as $p \to \infty$).
- Let $G = (V, E)$ be a finite, undirected, connected graph, with positive edges’ lengths. Let $u, v \in V$ be a pair of vertices.
- Define $D(u, v) =$ the length of the shortest path from $v$ to $u$ in $G$. This $D$ is a norm.
Distance vs. Similarity

- Distance and similarity are dual notions. If $A$ and $B$ are highly similar objects, than intuitively they have small distance.
Distance vs. Similarity

- Distance and similarity are dual notions. If $A$ and $B$ are highly similar objects, then intuitively they have small distance.
- This intuition indeed holds for pairwise global sequence alignment (see prob. 6 in assignment 1).
Distance vs. Similarity

- Distance and similarity are dual notions. If $A$ and $B$ are highly similar objects, than intuitively they have small distance.
- This intuition indeed holds for pairwise global sequence alignment (see prob. 6 in assignment 1).
- We can replace sequence similarity by distance and obtain qualitatively similar results for pairwise global sequence alignment.
Distance vs. Similarity

- Distance and similarity are dual notions. If \( A \) and \( B \) are highly similar objects, than intuitively they have small distance.
- This intuition indeed holds for pairwise global sequence alignment (see prob. 6 in assignment 1).
- We can replace sequence similarity by distance and obtain qualitatively similar results for pairwise global sequence alignment.
- Dynamic programming can be used to find minimum distance alignment in time \( O(n \cdot m) \).
Local Sequence Alignment

- In local sequence alignment, we have two input sequences (strings): The query $S$, and the text $T$. 

- If $S'$ and $T'$ is an optimal alignment of $S$ and $T$, then $S''$ and $T''$ can contain indels.
Local Sequence Alignment

- In local sequence alignment, we have two input sequences (strings): The query $S$, and the text $T$.
- The goal is to find two substring: $A$ – substring of $S$ and $B$ – substring of $T$ such that the (global) alignment score between $A$ and $B$ is maximized (over all choices of pairs of substrings).
Local Sequence Alignment

- In **local** sequence alignment, we have two input sequences (strings): The query $S$, and the text $T$.
- The goal is to find two substring: $A$ – substring of $S$ and $B$ – substring of $T$ such that the (global) alignment score between $A$ and $B$ is maximized (over all choices of pairs of substrings).
- Notice that if $A'$, $B'$ is an optimal alignment of $A$ and $B$, then $A'$, $B'$ can contain **indels**.
Local Alignment DP Algorithm

- The global and local alignment problems seem very different. But a rather small change in the "global" DP algorithm yields an efficient $O(n \cdot m)$ "local" DP algorithm.
Local Alignment DP Algorithm

- The global and local alignment problems seem very different. But a rather small change in the ”global” DP algorithm yields an efficient $O(n \cdot m)$ ”local” DP algorithm.
- Goal: Fill the $m \times n$ matrix by values $U(i, j)$, the value of the best (global) alignment between all suffixes of $i$-prefix of $S$ and suffixes of $j$-prefix of $T$. 
Local Alignment DP Algorithm

- Initialize all boundary values (upper row, left column) to 0.
Local Alignment DP Algorithm

- Initialize all boundary values (upper row, left column) to 0.
- Update rule: 
  \[ U(i + 1, j + 1) = \max \left( \begin{array}{l} 
  U(i, j) + \delta(S[i + 1], T[j + 1]), \\
  U(i + 1, j) + \delta(-, T[j + 1]), \\
  U(i, j + 1) + \delta(S[i + 1], -), \\
  0 \end{array} \right). \]
Local Alignment DP Algorithm

- Initialize all boundary values (upper row, left column) to 0.
- Update rule: \( U(i + 1, j + 1) = \max \left( U(i, j) + \delta(S[i + 1], T[j + 1]), U(i + 1, j) + \delta(−, T[j + 1]), U(i, j + 1) + \delta(S[i + 1], −), 0 \right) \).
- Keep pointers like before (no pointer if \( \max = 0 \)).
Local Alignment DP Algorithm

- Initialize all boundary values (upper row, left column) to $0$.
- Update rule: $U(i + 1, j + 1) = \max$
  $U(i, j) + \delta(S[i + 1], T[j + 1])$,
  $U(i + 1, j) + \delta(-, T[j + 1])$,
  $U(i, j + 1) + \delta(S[i + 1], -)$,
  $0$.
- Keep pointers like before (no pointer if $\max = 0$).
- Pick the highest entry in the matrix. Trace back to recover optimal local alignment(s).
Local Sequence Alignment II

For example, suppose our sequences are $S = \text{GODSAVEOURQUEEN}$ and $T = \text{BARBERSHAVEOURKING}$. 
Local Sequence Alignment II

For example, suppose our sequences are

\[ S = \text{GODSAVEOURQUEEN} \quad \text{and} \quad T = \text{BARBERSHAVEOURKING}. \]

Our scoring gives +2 for a match, -1 for a mismatch, and -2 for indel.
Local Sequence Alignment II

For example, suppose our sequences are

\[ S = \text{GODSAVEOURQUEEN} \quad \text{and} \quad T = \text{BARBERSHAVEOURKING}. \]

Our scoring gives +2 for a match, -1 for a mismatch, and -2 for indel.

A reasonable (best?) choice would be the substrings \text{SAVEOUR} and \text{SHAVEOUR}, with the (global) alignment between them being
Local Sequence Alignment II

For example, suppose our sequences are

\[ S = \text{GODSAVEOURQUEEN} \quad \text{and} \quad T = \text{BARBERSHAVEOURKING}. \]

Our scoring gives +2 for a match, -1 for a mismatch, and -2 for indel.

A reasonable (best?) choice would be the substrings \text{SAVEOUR} and \text{SHAVEOUR}, with the (global) alignment between them being

\[
\begin{align*}
S & \_A V E O U R \\
SHAVEOUR
\end{align*}
\]
Local Sequence Alignment II

For example, suppose our sequences are
\[ S = \text{GODSAVEOURQUEEN} \quad \text{and} \]
\[ T = \text{BARBERSHAVEOURKING}. \]

Our scoring gives +2 for a match, -1 for a mismatch, and -2 for indel.

A reasonable (best?) choice would be the substrings
\[ \text{SAVEOUR} \quad \text{and} \quad \text{SHAVEOUR}, \]
with the (global) alignment between them being

\[
\begin{align*}
S & \_ \text{AVEOUR} \\
\text{SHAVEOUR} \\
\end{align*}
\]

In the context of the original sequences, we have
\[ \text{GODSAVEOURQUEEN} \quad \text{and} \]
\[ \text{BARBERSHAVEOURKING} \]
Other Versions of Alignment

- Global in $S$, local in $T$. 
Other Versions of Alignment

- Global in $S$, local in $T$.
- Affine gap penalties.
Other Versions of Alignment

- Global in $S$, local in $T$.
- Affine gap penalties.
- Linear $O(n + m)$ space algorithm.