Sequence Alignment
Brief review from last class

• DNA is has direction, we will use only one (5’ -> 3’) and generate the opposite strand as needed.

• DNA is a 3D object (see lecture 1) but we will model it as a 2D object/string.

• Two generative models of sequences:
  – Multinomial: probability is equal to product of individual probabilities (no prior dependence)
  – Markov: probability is equal to product of probabilities given a fixed number of preceding characters.
Today

• The next two weeks, we will discuss sequence alignment and all of its basic flavors.

• Arguably one of the most important algorithms in bioinformatics; over 40 years old.

• The ultimate goal of alignment is to describe sequence similarity, or how closely two sequences match each other.
  – Can be a score (number)
  – Can also be an “alignment” (visual)
Similarity vs. biology

- Similarity (also called identity) is the number of matches / alignment length

- Homology, on the other hand, implies sequences came from a common ancestor

- Two kinds of homology:
  - Orthologous - speciation-based split
  - Paralogous - gene duplication-based split
Various types
Examples

- Database search
  - Pairwise
  - Multiple

- Phylogeny

- Protein structure
  - Global
  - Suffix-Prefix
  - Local
  - Spliced

- Gene finding
Applications

• Prediction on function
  – Commonalities among sequences can imply similar functions

• Database searching (BLAST)
  – Find interesting genes in a new genome

• Sequence divergence
  – Look at evolutionary relationships

• Sequence assembly
  – Making a big sequence from a bunch of small ones
Global alignment

• Also called a *pairwise alignment*.

• Intuitive goal: related sequences will share many (most?) characters. To maximize this we introduce gaps represented by “-”
Two simple rules

- Rule #1:
  - A gap must be aligned to a nongap, i.e., “-” can not align to “-”

- Rule #2:
  - To distinguish good alignment from not so good ones, we introduce a scoring function $E$. Some functions have biological meaning, some are arbitrary.

- Consequence #1:
  - Alignment length can be no longer than sum of two sequences!
Example from text

• How do we align these proteins:
  – VIVALASVEGAS
  – VIVADAVIS
Alignments

• Show one sequence placed above another such that similarity is revealed.
• Alignments can be longer than either string!

Example:

A: CAT - TCA - C
B: C - T CGC A G C
Improving readability

Example:

A: C A T - T C A - C
B: C - T C G G A G C
Scoring functions

• Here is a basic scoring function that rewards 1 for a match and -1 for a mismatch gap

\[ E(-,a) = E(a,-) = E(a,b) = -1 \quad \forall a \neq b \]

\[ E(a,b) = 1 \quad \forall a = b \]

• Can also be represented as a substitution matrix.
In class example

S: CATCAC
T: CTCCAGC

\[ E(-,a) = E(a,-) = E(a,b) = -1 \quad \forall a \neq b \]
\[ E(a,b) = 1 \quad \forall a = b \]
Measuring similarity

Score: A measure of alignment quality

\[
\begin{align*}
\text{C A T} & \quad \text{T C A} \quad \text{C} \\
\text{C - T C G C A G C} & \quad \text{------------} \\
10 & \quad -5 \quad 10 \quad -5 \quad -2 \quad 10 \quad 10 \quad -5 \quad 10
\end{align*}
\]

Total = 33

Scored as $E(C,C)$, $E(A,-)$, $E(T,T)$, $E(-,C)$, etc.
Alignment overview

• Computationally, naïve alignments grow exponentially with $n$ : not good
  – There are $10^{17}$ alignments for two length 30 sequences.

• Luckily, a tried and true method for solving similar problems (we’ll provide an overview today) comes to the rescue.

Basic intuition

• Suppose we have an optimum alignment of size L. Is the following true?

• $A^* = A^*(s_1 \ldots s_i, t_1 \ldots t_j) + A^*(s_{i+1} \ldots s_n, t_{j+1} \ldots t_m)$
  - Where $|s| = n$ and $|t| = m$

• If so, what would happen if $i = n - 1$ and $j = m - 1$?
Visualization

Case 1: Match $s[n]$ w/ $t[m]$

\[
\begin{array}{cccccccc}
\text{n - 1} & \text{n} \\
\text{s: } & C & A & T & T & C & A & C \\
\text{t: } & C & - & T & T & C & A & G \\
\text{m -1} & \text{m}
\end{array}
\]

Case 2: Match $t[m]$ w/ gap

\[
\begin{array}{cccccccc}
\text{n - 1} & \\
\text{s: } & C & A & T & T & C & A & - \\
\text{t: } & C & - & T & T & C & A & G \\
\text{m -1} & \text{m}
\end{array}
\]

Case 3: Match $s[n]$ w/ gap

\[
\begin{array}{cccccccc}
\text{n - 1} & \text{n} \\
\text{s: } & C & A & T & T & C & A & C \\
\text{t: } & C & - & T & T & C & A & - \\
\text{m-1}
\end{array}
\]
Global alignment

- Dynamic programming (DP) will save the day!

- DP is a general technique used when a large problem can be broken into smaller, easier problems like this.

- To solve sequence alignment, we will fix two substrings and find the best way to add the next character from at least one string.
Notation from Jackson and Aluru

- \( S(i,j) = E(\text{Opt}(A[0,i], B[0,j])) \)
  - "\( S(i,j) \) is the evaluated score of the optimal alignment between the prefix of \( A \) ending at position \( i \) and the prefix of \( B \) ending at position \( j \)."

\[
S(n-1, m-1) = \max \begin{cases} 
S(n-2, m-2) + E(A[n-1], B[m-1]) \\
S(n-1, m-2) + E(\text{-'}, B[m-1]) \\
S(n-2, m-1) + E(A[n-1], \text{-'}) 
\end{cases}
\]

\[
S(i, j) = \max \begin{cases} 
S(i-1, j-1) + E(A[i], B[j]) \\
S(i, j-1) + E(\text{-'}, B[j]) \\
S(i-1, j) + E(A[i], \text{-'}) 
\end{cases}
\]
Requirements

• We will need four things to compute a global alignment:
  1. Substitution matrix (parameters)
  2. Recurrence relation
  3. Filling up a table
  4. Traceback
Pairwise Global Alignment

\[ T[i,j] = \text{Score of optimally aligning first } i \]
\[ \text{bases of } s \text{ with first } j \text{ bases of } t. \]

\[ T[i,j] = \max \begin{cases} 
T[i-1,j-1] + \text{score}(s[i], t[j]) \\
T[i-1,j] + g \\
T[i,j-1] + g 
\end{cases} \]
+10 for match, -2 for mismatch, -5 for space (rowwise)
Traceback yields both optimal alignments in this example
Some Results

• Most pairwise sequence alignment problems can be solved in $O(mn)$ time. Some speedups exist, most notably the Four Russians technique.

• Space requirement can be reduced to $O(m+n)$, while keeping run-time fixed [Myers88].

• Two highly similar sequences can be aligned in $O(dn)$ time, where $d$ is a measure of the distance between the sequences [Landau86].
Pairwise Sequence Alignment

Variations for future classes:

• Given two sequences, find if parts of them are similar (local alignment).

• Given a large sequence and a short sequence, find if the short sequence is similar to a stretch of the long sequence.

• Cool fact is these are easy to do once we learn the basics of global alignment!