Part I. Sequence Analysis:
2. Linear Space algorithm, Local Alignment, Scoring Function

Lecture 5 – Oct 14, 2015
CSE 527 Computational Biology
Instructor: Su-In Lee
TA: Javad Hosseini
TTh 12:00-1:20 @ SAV 130

Review: Global Alignment Algorithm
- Input:
  X: AGGCTATCACCTGACCTCCAGGGATGCCC
  Y: TAGCTATCACCGCGTGTGCTATTTGCCC
- Output:
  -AGGCTATCACCTGACCTCCAGGGATGCCC--TGCCC---
  TAGCTATCAC--GACC--GTTGCA--TGCCC
- Needleman-Wunsch algorithm can solve with O(MN) time and space (M: length of X, N: length of Y)

Introduction: Compute optimal score
It is easy to compute F(M, N) in linear space

Allocate (column[1])
Allocate (column[2])

For j = 1...N
  If j > 1, then:
    Free(column[j - 2])
    Allocate(column[j])
  For i = 1...M
    F(i, j) = ...
Linear-space alignment

To compute both the optimal score and the optimal alignment:

Divide & Conquer approach:

Notation:

\( x' , y' \): reverse of \( x, y \)

E.g. \( x = \text{accgg} \);
\( x' = \text{ggcca} \)

\( F(i, j) \): optimal score of aligning \( x'_{1...i} \) & \( y'_{1...j} \)

same as aligning \( x_{i+1...M} \) & \( y_{j+1...N} \)

Lemma: (assume \( M \) is even)

\[
F(M, N) = \max_{k = 0...M} \left( F(k, N/2) + F(M-k, N/2) \right)
\]

Example:

\[
\begin{array}{c}
\text{ACCAGGTG} - \text{GGACTGGCAG} \\
\text{ACC-GTGGCGAGGACTG} - \text{CAT}
\end{array}
\]

\( k^* = 8 \)

Now, using 2 columns of space, we can compute for \( k = 1...M \),
\( F(k, N/2), F(M-k, N/2) \)

PLUS the backpointers

Now, we can find \( k^* \) maximizing \( F(k, N/2) + F(M-k, N/2) \)

Also, we can trace the path exiting column \( N/2 \) from \( k^* \)
Linear-space alignment

- Iterate this procedure to the left and right!

Hirschberg’s Linear-space algorithm:

MEMALIGN(l, l', r, r'): (aligns x...x_l with y...y_r)

1. Let \( h = \lfloor (r' - r)/2 \rfloor \)
2. Find (in Time \( O((l' - l) \times (r' - r)) \), Space \( O(l' - l) \)) the optimal path, \( L_h \), entering column \( h - 1 \), exiting column \( h \)
   - \( k_1 \) = position at column \( h - 2 \) where \( L_h \) enters
   - \( k_2 \) = position at column \( h + 1 \) where \( L_h \) exits
3. MEMALIGN(l, \( k_1 \), r, \( h - 2 \))
4. Output \( L_h \)
5. MEMALIGN(\( k_2 \), l', \( h + 1 \), r')

Top level call: MEMALIGN(1, M, 1, N)

Time, Space analysis of Hirschberg’s algorithm:
To compute optimal path at middle column,
- For box of size \( M \times N \),
  - Space: \( 2N \)
  - Time: \( cMN \), for some constant \( c \)

Then, left, right calls cost \( c(M/2 \times k^* + M/2 \times (N - k^*)) = cMN/2 \)

All recursive calls cost:
- **Total Time**: \( cMN + cMN/2 + cMN/4 + \ldots = 2cMN = O(MN) \)
- **Total Space**: \( O(N) \) for computation,
  \( O(N + M) \) to store the optimal alignment

Local Alignment Algorithm
Global alignment algorithm

- Input:
  - X: AGGCTATCACCTGACCTCCAGCCGATGCC
  - Y: TAGCTATCGACCCCGGTGCGATTTTGCCGCAC

- Output:
  - AGGGCTATCACCTGACCTCCAGCCGATGCC
  - TAGCTATCACCTGACCGCGGTCGATTTTGCCGCAC

- Needleman-Wunsch algorithm can solve with O(MN) time and space (M: length of X, N: length of Y)
- However, it cannot do certain things ...

Local alignment: motivation

- Global alignment algorithms align sequences over their entire lengths

- Example local alignment problems
  - Functionally related proteins can have only short stretches (e.g., domains) of conserved sequences
  - Genes are shuffled between genome sequences

Domains are distinct functional and/or structural units in a protein sequence

- Oct4 – a transcription factor that is critically involved in the self-renewal of undifferentiated embryonic stem cells
- Length – 360 aa

Domains are conserved across related proteins

- Pit1, Oct1, Oct2, Oct4 and Oct6 contain the same DNA-binding domains

Aligning the human genome to the mouse genome

- Genes are shuffled between genomes

**REARRANGEMENTS**

- Inversion
- Translocation
- Duplication


A variant of the basic algorithm: Local alignment

- Maybe it is OK to have an unlimited # of gaps in the beginning and end:

```
AATTCGCCTACTACCTGACCTCCAGGCGATGCCCCCTCCGGC
```

```
GCGAGTTCATCTATCACC--GACCGC--GTCGGCAATTACTGCCA
```

Then, we don’t want to penalize gaps in the ends

The local alignment problem

Given two strings

\[ x = x_1 \ldots x_M \]
\[ y = y_1 \ldots y_N \]

Find substrings \( x' \), \( y' \) whose similarity (optimal global alignment value) is maximum

\[ x = \text{aaaacc cccggg gttta} \]
\[ y = \text{ttccgggaaccaacc} \]

The Smith-Waterman algorithm

**Idea:** Ignore badly aligning regions

Modifications to Needleman-Wunsch:

**Initialization:** \( F(0, j) = F(i, 0) = 0 \)

**Iteration:**

\[
F(i, j) = \max \begin{cases} 
0 & \\
F(i-1, j) - d & \\
F(i, j-1) - d & \\
F(i-1, j-1) + s(x_i, y_j) & 
\end{cases}
\]
The Smith-Waterman algorithm

**Termination:**

1. If we want the best local alignment...
   \[ F_{OPT} = \max_{i,j} F(i, j) \]
   Find \( F_{OPT} \) and trace back
2. If we want all local alignments scoring > t
   For all i, j find \( F(i, j) > t \), and trace back?

Waterman–Eggert '87: find all non-overlapping local alignments with minimal recalculation of the DP matrix

### Scoring Function

\[ \sigma(A,G), \sigma(A,-), \sigma(A,-), \text{etc?} \]

<table>
<thead>
<tr>
<th>F(i,j)</th>
<th>F(i-1,j-1) + \sigma(S[i], T[j])</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(i-1,j) + \sigma(S[i], - )</td>
<td></td>
</tr>
<tr>
<td>F(i,j-1) + \sigma(- , T[j])</td>
<td></td>
</tr>
</tbody>
</table>

### Scoring Rules / Matrices

- How should \( \sigma \) be defined?
  - \( \sigma(A,G), \sigma(A,-), \sigma(A,-), \text{etc?} \)
  \[ F(i,j) = \max \left\{ F(i-1,j-1) + \sigma(S[i], T[j]) \right\} \]

- Why are they important?
  - The choice of a scoring rule can strongly influence the outcome of sequence analysis

- What do they mean?
  - Scoring matrices implicitly represent a particular theory of evolution
  - Elements of the matrices specify the similarity of one letter to another (nucleotide or amino acid)

### Probabilistic Interpretation

**X:** TCCAGGTG–GAT

\[ \text{||| | |} \]

**Y:** TGCAAGTGCG–T

**Chance or true homology?**

Sharing a common ancestor
Likelihood Ratio

\[
\text{Pr( Data | Homology )} = \frac{\prod_i \Pr(x_i) \Pr(y_i)}{\prod_i \Pr(x_i) \Pr(y_i)}
\]

Given an alignment between TCCAGG and TGCAAG,

\[
\begin{align*}
X: & \quad & \text{TCCAGG} & \text{TGCAAG} \\
& & \text{TCCAGG} & \text{TGCAAG} \\
Y: & & \text{TGCAAG} & \text{TGCAAG}
\end{align*}
\]
Score: Log Likelihood Ratio
- The most commonly used alignment score
  - Log likelihood ratio of the alignment under two models
    - Common ancestry
    - By chance

\[
\text{Score} = \log \left( \prod_i \frac{\Pr(x_i, y_i)}{\Pr(x_i)\Pr(y_i)} \right) = \\
= \sum_i \log \left( \frac{\Pr(x_i, y_i)}{\Pr(x_i)\Pr(y_i)} \right) = \sum_i s(x_i, y_i)
\]

Scoring Matrix for DNA Alignment
- A simple positive score for matches (+1) and a negative for mismatches and gaps (-1) are most often used.
- Transversions penalized more than transitions
  - Transitions: replacement of a purine base with another purine (A <-> G) or replacement of a pyrimidine with another pyrimidine (C <-> T)
  - Transversions: replacement of a purine with a pyrimidine or vice versa.
  - Transition mutations are more common than transversions

Scoring Matrix for Protein Alignment
- Created based on biological evidence.
  - Alignments can be thought of as two sequences that differ due to mutations.
  - Some of these mutations happen more frequently
  - Some mutations have little effect on the protein’s function
  - Some penalties will be less harsh than others.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>R</th>
<th>N</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>-2</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>R</td>
<td>-1</td>
<td>7</td>
<td>-1</td>
<td>3</td>
</tr>
<tr>
<td>N</td>
<td>-</td>
<td>-</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>K</td>
<td>-</td>
<td>-</td>
<td>-6</td>
<td>-</td>
</tr>
</tbody>
</table>

- Although R (arginine) and K (Lysine) are different amino acids, they have a positive score.
- They are both positively charged amino acids
  - R → K will not greatly change function of protein.

\[
\text{AKRANR} - 1 + (-1) + (-2) + 5 + 7 = 11
\]

\[
\text{KAAANKK}
\]
Amino acid (aa) changes that tend to preserve the physical/chemical properties of the original aa

- Polar to polar
  - aspartate (D) → glutamate (E)
- Nonpolar to nonpolar
  - alanine (A) → valine (V)
- Similarly behaving amino acids
  - leucine (L) to isoleucine (I)

More prone to mutate in the evolutionary process

Amino acids are not born equally

Amino acids that share similar properties are more prone to mutate to each other in the evolutionary process

**Conservation**

- Polar to polar
  - aspartate (D) → glutamate (E)
- Nonpolar to nonpolar
  - alanine (A) → valine (V)
- Similarly behaving amino acids
  - leucine (L) to isoleucine (I)

Amino acids that share similar properties are more prone to mutate to each other in the evolutionary process.

**The S in a Scoring Matrix**

(as log likelihood ratio)

\[ s(x_i, y_i) = \log \left( \frac{\Pr(x_i, y_i | \text{common ancestry})}{\Pr(x_i) \Pr(y_i)} \right) \]

Pr(x, aligned with y, at position i | common ancestry)

Pr(x, aligned with y, at position i | by change)

How do we acquire the probabilities Pr(a), Pr(a,b)?

**BLOSUM62**

Amino Acids

Hydrophobic

- P, T, F, Y, W

Basic

- K, R

Aromatic

- L, I, V

Chargeable

- E, Q

Small

- S, N, C

Hydrophilic

- G, A, D, H, N

Conservation

- Conservation of properties

Amino acids are not born equally

Amino acids that share similar properties are more prone to mutate to each other in the evolutionary process.
**BLOSUM**

- Most widely used scoring matrix for protein sequence alignment
- BLOck SUbstitution Matrices

For each pair of amino acid \((a, b)\)

\[
\log \frac{s(a, b)}{P_{ab}} = P_{a} + P_{b} - 1
\]

- Observed probability (or frequency) of a substitution in our alignments
- Probability of random substitution based on the frequency of each amino acid

**Constructing BLOSUM**

- To avoid bias in favor of a certain protein, first eliminate sequences that are more than \(x\) % identical
- The elimination is done by either
  - removing sequences from the block, or
  - finding a cluster of similar sequences and replacing it by a new sequence that represents the cluster (majority vote)

**BLOSUM**

- Conserved “blocks” of a large sample of proteins
  - A block contains aligned ungapped segments corresponding to the most highly conserved regions of proteins

| Bpi Bovine | npGivaRItqkgLdyacqqgVltlQkele |
| Bpi Human  | npGvvvRIsqkgLdyasqqgHla1Qkelk |
| Cept Human | eaGlolvRItkpaLvlvlvhetekviQtqafq |
| Lbp Human  | npGlvaRItdkgLqyaaqeglalQsell |
| Lbp Rabbit | npGlitRItdkgLeyasreglalQrkil |

- ~2000 blocks of aligned sequence segments from 500 groups of related proteins
- \(P_{a}, P_{b}\) and \(P_{ab}\) are derived from blocks

*Bunkenhof and Henikoff, Amino acid substitution matrices from protein blocks, PNAS 1992*
Constructing BLOSUMx

- To avoid bias in favor of a certain protein, first eliminate sequences that are more than $x\%$ identical.
- The elimination is done by either removing sequences from the block, or finding a cluster of similar sequences and replacing it by a new sequence that represents the cluster (majority vote).
- BLOSUMx is the matrix built from blocks with no more $x\%$ of similarity.
  - Note: BLOSUM62 is the default matrix for popular tools (e.g., protein BLAST).

Collecting substitution statistics

After clustering some sequences in each block, count amino acids pairs in each column:
1. 6 AA pairs, 4 AB pairs, 4 AC, 1 BC, 0 BB, 0 CC.
   - Total = 6+4+4+1=15
2. Normalize results to obtain probabilities ($p_a$'s and $p_{ab}$'s):
3. Compute log likelihood ratio score matrix from probabilities:
   \[
   s(a,b) = \frac{1}{\lambda} \log \left( \frac{p_{ab}}{p_a p_b} \right)
   \]

Other Scoring Matrices

- PAM (Point Accepted Mutation)
  - Based on global alignments of closely related proteins.
  - The PAM1 is calculated from comparisons of sequences with no more than 1% divergence.
  - Other PAMx matrices are extrapolated from PAM1 based on an evolutionary model.

- DNA scoring matrices
  - Less effective to compare the protein coding regions of genes at nucleotide level.
  - DNA is less conserved than protein sequences.

PAM vs. BLOSUM

- PAM (Point Accepted Mutation)
  - Based on global alignments of closely related proteins.
  - The PAM1 is calculated from comparisons of sequences with no more than 1% divergence.
  - Other PAMx matrices are extrapolated from PAM1 based on an evolutionary model.

- BLOSUM (BLOCK SUBstitution Matrices)
  - Based on global alignments.
  - BLOSUMx is empirically estimated from sequences with no more than $x\%$ identity in the blocks.
  - BLOSUMx matrix is not extrapolated from other BLOSUMy matrices.