Alignments 1
Sequence Analysis
Lecture 2

Why is similarity interesting?
• Common ancestry is a very important observation
  Makes it more likely that genes share the same function
• Homology: sharing a common ancestor
  – a binary property (yes/no)
  – It’s a nice tool:
    When (a known gene) G is homologous to (an unknown) X it means we gain a lot of information on X

Functional and evolutionary
• Evolutionary relation, reconstruction:
  – Based on sequence
    • Identity (simplest method)
    • Similarity
    • Homology (the ultimate goal)
  – Other (e.g., 3D structure)
• Functional relation
  Sequence → Structure → Function

Searching for similarities
• What is the function of a new gene?

Evolution and 3d protein structure information
Insect dihydrofolate
The distance from the active site (yellow) determines the rate of evolution.
(red = fast evolution
blue = slow evolution)

How to determine similarity?

- Frequent evolutionary events:
  1. Substitution
  2. Insertion, deletion
  3. Duplication
  4. Inversion

  We’ll use only these

Evolution at work

Alignment

- Mutations: substitution, insertion and deletion

  Which alignment is better?
  Use common sense and call it:
  - Simplest
  - Most probable
  - Maximum likely

Scoring

- Should give reasonable alignments
- And have to assign scores to:
  - Substitution (or match/mismatch)
    - DNA
    - proteins
  - Gap penalty
    - Linear: \( g(k) = αk \)
    - Affine: \( g(k) = β + αk \)
    - Concave, e.g.: \( g(k) = \log(k) \)
- The score for an alignment is the sum of scores of all alignment columns

Substitution matrices

- Define a score for match/mismatch of letters
- DNA
  - Simple:
    - Used in genome alignments:

Substitution matrices for aa

- Amino acids are not equal:
  1. Some are easily substituted, similar:
    - biochemical properties
    - structure
  2. Some mutations occur more often
    due to similar codons

- The two above give us substitution matrices

Blosum62 matrix
### Linear vs. affine scoring

<table>
<thead>
<tr>
<th>Scoring Rule</th>
<th>Linear</th>
<th>Affine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertion</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>Deletion</td>
<td>-3</td>
<td>-1</td>
</tr>
</tbody>
</table>

... and +1 for match

### The algorithm
- **Goal:** find the maximal scoring alignment
- **Scores:** m for match, s for mismatch, g for insertion/deletion
- **Dynamic programming**
  - Solve smaller subproblem(s)
  - Iteratively extend the solution
- **The best alignment** for \(X[1...i]\) and \(Y[1...j]\) is called \(M[i,j]\)

\[
\begin{align*}
X_1 & \ldots X_i & - \quad X_i \\
- & Y_1 & \ldots Y_j \\
\end{align*}
\]

### The algorithm for linear gap penalties

### Example: global alignment of two sequences
- **Align two DNA sequences:**
  - GAGGTA
  - GAGGCGA (note the length difference)
- **Parameters of the algorithm:**
  - Match: score \(N, N = 0\)
  - Mismatch: score \(-1\)
  - Gap: \(g = 2\)

### The algorithm. Step 1: init
- **Create the matrix**
  - **Initialization**
    - \(a = 0\) at \([0,0]\)
    - Apply the equation...
The algorithm. Step 1: init

- Initiation of the matrix:
  - 0 at pos (00)
  - Fill in the first row using the "-" rule
  - Fill in the first column using "*"

The algorithm. Step 2: fill in

- Continue filling in of the matrix, remembering from which cell the result comes (arrows)

The algorithm. Step 3: trace-back

- Start at the last cell of the matrix
- Go in the direction of arrows
- Sometimes the value may be obtained from more than one cell (which one?)

The algorithm. Step 3: trace-back

- Extract the alignments
  a) GACT-CA
  b) GA-GTCA

Take-home message

- Homology
- Why are we interested in similarity?
- Pairwise alignment: global alignment