Today’s Lecture

• Improved scoring of pairwise alignments
  – Affine gap penalties
  – Profiles
• Smith-Waterman special cases
Above path corresponds to following alignment (w/ lower case letters considered unaligned):

```
aCGTTGAATGAccca
```

```
gCAT−GAC−GA
```
Two strategies to allow partial non-independence while preserving dynamic programming framework:

- Enhance graph
- Allow scores to depend on position within the sequence (i.e. not just on a BLOSUM-type score matrix)
  - so some substitutions (of same residues) or gaps penalized more heavily than others
Gap Penalties

TNAVAHVD------DMPNAL
YEAAIQLQVTGVVVTDATL

• Usual scoring scheme assigns same penalty $g$ to each gap edge, so
  – weights on extended gaps of size $s$ are *linear* in $s$, i.e.
  – total gap penalty $\text{gap}(s) = s \times g$.
  – e.g. in above example, if each $g = -6$, total penalty on gap would be

$$\text{gap}(5) = 5 \times -6 = -30$$
Gap Penalties

• Would like more flexible gap penalties:
  • In proteins, insertions & deletions are rare;
    – but when occur, often consist of several residues, because
      • they are in regions (loops) tolerant of length changes
    – at DNA level, indels in protein coding sequence usually a multiple of 3 nucleotides
      • otherwise, would change reading frame
  • In noncoding sequence,
    – the most common indel size is 1
    – *but* larger indels occur much more frequently than multiple independent single-base indels
• Can allow arbitrary convex gap penalties
  – \( \text{gap}(s+t) \geq \text{gap}(s) + \text{gap}(t) \), where \( s \) and \( t \) are (integer) gap sizes

by extending edit graph:
  – add edges corresponding to arbitrary length gaps from each vertex to each horizontally or vertically downstream vertex
  – (convexity condition prevents favoring two adjacent short gaps over a single long gap).

Time complexity now \( O(MN(M+N)) \)
  – often unacceptable for moderate \( M, N \).
  – Also: how to choose appropriate weights? (need data to estimate!)
Affine Gap Penalties

- **Affine** gap penalties:
  - less general than arbitrary convex penalties, but
  - more general than linear penalties.
- Two parameters:
  - **gap opening** penalty $g_o$
  - **gap extension** penalty $g_e$
- $\text{gap}(n)$ (penalty for size $n$ gap) is then
  \[ g_o + n \cdot g_e = g_i + (n - 1) \cdot g_e \]
  where the gap **initiating** penalty $g_i = g_o + g_e$
• Example: for BLOSUM62, good penalties are
  – $g_i = -12$
  – $g_e = -2$

These perform much better than linear penalty
  – (e.g. $g = -6$)

• N.B. Durbin et al. reverse $g_i$ and $g_o$
  – $g_i$ is called the ‘gap opening’ penalty

• Can obtain affine penalties using extension of edit graph, retaining complexity $O(MN)$:
Edit Graph for Affine Gap Penalties

Double # vertices, creating left-right pair in place of each original vertex. Each cell looks like this:

• gap-opening edges from left vertex to right vertex of each pair: weight $g_o$

• gap extension edges going horizontally or vertically between right vertices: weight $g_e$

• diagonal edges originate from either left or right vertex, but always go to a left vertex.

each left vertex has out-degree and in-degree = 2

each right vertex has out-degree and in-degree = 3
• Paths in the augmented graph still correspond to alignments
  – can $\exists$ more than one path for same alignment
  – but highest scoring paths still give best alignments
• Score assigned to size $n$ gap is $g_o + n g_e$
  – i.e. affine penalty
• Smith-Waterman-Gotoh algorithm
Profiles (position-specific scoring)
The *Edit Graph* for a Pair of Sequences
Profiles: Position-specific scoring scheme specifying score of each possible substitution at each position of a sequence

From R. Luthy, I. Xenarios and P. Bucher, Improving the sensitivity of the sequence profile method Protein Sci. 3: 139-146 (1994)
• This is an important improvement!
  – reflects fact that different parts of sequence may evolve at different rates

• e.g. in proteins,
  – internal core region of tightly packed residues, or active sites of enzyme, are more highly conserved;
  – surface residues, particularly in loops, often less conserved.
  – so scores tend to be correlated (high scores in core, lower on surface)
Rates of amino acid exchange in mammalian proteins by burial status

- **H**: hydrophobic
- **P**: polar

**Exchange Type**
- H → H
- P → P
- P → H
- H → P

**Burial**
- Exposed
- Intermediate
- Buried

*Saunders & Green Mol Biol Evol 2007 24:2632-2647; doi:10.1093/molbev/msm190*
• PSIBLAST approach:
  – initially compare query sequence to database sequences (using BLOSUM-type scoring matrix),
  – build profile using initial matches
  – rescan database using profile

• Optimal choice of
  – substitution matrix,
  – gap penalties, or
  – profiles

  comes from LLR based on alignment data (target vs background)
Smith-Waterman special cases

- Various special cases are optimal path problems for *subgraphs* of edit graph:
- *Gap-free* alignments correspond to paths confined to a diagonal of edit graph
  - (i.e. subgraph without horizontal & vertical edges).
- Find *perfectly* matching segments using weights
  +1 for identical residue pair,
  -\(\infty\) (or large negative penalty) for mismatches or gaps.

Less efficient than “sorting pointers” method from lecture 1 / HW1.
The *Edit Graph* for a Pair of Sequences

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>T</th>
<th>G</th>
<th>A</th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>A</th>
<th>C</th>
<th>C</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

18
• Find *imperfect internal repeats* by searching edit graph of sequence against itself
  – i.e. the same sequence labels columns and rows

above ( & not including) the main diagonal:
  – if include main diagonal, best path will be identity match to self
  – complexity = $O(N^2)$ where $N =$ sequence length.

Graph for finding imperfect internal repeats: