Alignments 1

Sequence Analysis

Part of the presentation courtesy of Radek Szklarczyk (radek@cs.vu.nl)

Searching for similarities

• What is the function of the new gene?
• The “lazy” investigation:
  – Find the set of similar proteins
  – Identify similarities and differences
  – For long proteins: identify domains
Is similarity really interesting?

- Common ancestry is more interesting
  - 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 that we gain a lot of information on X

---

Is similarity really interesting?

Fish → Shark
Land reptile → Ichthyosaur
Land mammal → Porpoise
Functional and evolutionary relation

- Evolutionary relation, reconstruction:
  - Based on sequence
    - Identity (simpliest method)
    - Similarity
  - Homology (the ultimate goal)
  - Other (e.g., 3D structure)
- Functional relation
  \[ \text{determines} \quad \text{determines} \]
  \[ \text{Sequence} \rightarrow \text{Structure} \rightarrow \text{Function} \]

Evolution and 3d structure

Isocitrate dehydrogenase:
The distance from the active site \textit{(yellow)} determines the rate of evolution

Dean, A. M. and G. B. Golding, 2000, 
Enzyme evolution explained (sort of), Pacific Symposium on Bioinformatics 2000
How to determine similarity

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

- Evolution at work

![Diagram showing a common ancestor (Z) and two branches (X, Y)]

Common ancestor, usually extinct
available

 Alignment

- Mutations: substitution, insertion and deletion

  GTACT--C
  GGA-TGAC

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

  1) GTA
     GCA
   2) GT-A
      G-CA
   3) GTA---
      ---GCA
Scoring

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

Substitution matrices

- Define a score for match/mismatch of letters
- DNA
  - - - Simple:
    | A | C | G | T |
    |---|---|---|---|
    | 1 | -1| -1| -1|
    | -1| 1 | -1| -1|
    | -1| -1| 1 | -1|
    | -1| -1| -1| 1 |
  - - - Used in genome alignments:
    | A | C | G | T |
    |---|---|---|---|
    | 91| -114| -31| -123|
    | -114| 100| -125| -31 |
    | -31| -125| 100| -114 |
    | -123| -31| -114| 91 |
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

|   | A  | R  | N  | D  | C  | Q  | E  | G  | H  | I  | L  | K  | M  | F  | P  | S  | T  | W  | Y  | V  |
|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| A | 4  | -1 | -2 | -2 | 0  | -2 | -1 | -1 | -1 | -2 | -1 | 1  | 0  | -3 | -2 | 0  |    |    |    |    |    |
| R | -1 | 5  | 0  | -2 | -3 | 1  | 0  | -2 | 0  | -3 | -2 | 2  | -1 | -3 | -2 | -1 | -1 | -3 | -2 | -3 |    |
| N | -2 | 0  | 6  | -1 | 3  | 0  | 0  | 0  | 1  | -3 | -3 | 0  | -2 | -3 | -2 | 1  | 0  | -4 | -2 | -3 |    |
| D | -2 | -2 | 1  | 6  | -3 | 0  | 2  | -1 | -1 | -3 | -4 | -1 | -3 | -3 | -1 | 0  | -1 | -4 | -3 | -3 |    |
| C | 0  | -3 | -3 | 9  | -3 | -4 | 3  | -1 | -1 | -3 | -1 | -2 | -3 | -1 | -2 | -1 | 0  | -3 | -2 | -3 | -1 |
| Q | -1 | 1  | 0  | -3 | 5  | 2  | -2 | 0  | -3 | -2 | 1  | 0  | -3 | -1 | 0  | -1 | -2 | -1 | -2 | -1 |
| E | -1 | 0  | 0  | 2  | -4 | 2  | 5  | -2 | 0  | -3 | -1 | 1  | -2 | -3 | -1 | 0  | -1 | -3 | -2 | -1 |
| G | 0  | -2 | 0  | -1 | 3  | -2 | -2 | 6  | -2 | -4 | -2 | -2 | -3 | -2 | 0  | -2 | -2 | -3 | -3 |    |
| H | -2 | 0  | 1  | -1 | -3 | 0  | 0  | 2  | 8  | -3 | -1 | -2 | -1 | -2 | -1 | -2 | -2 | 2  | -3 |    |
| I | -1 | 3  | -3 | 3  | -3 | -3 | -4 | -3 | 4  | -2 | -3 | 1  | 0  | -3 | -2 | -1 | -3 | -1 | 3  |    |
| L | -1 | 2  | -3 | -4 | 1  | -2 | -3 | -4 | -3 | 2  | 4  | -2 | 0  | -3 | -2 | -1 | -2 | -1 | 1  |    |
| K | -1 | 2  | 0  | -1 | -3 | 1  | 1  | -2 | -1 | -3 | -2 | 5  | -1 | -3 | -1 | 0  | -1 | -3 | -2 | -2 |    |
| M | -1 | 1  | -2 | -3 | -1 | 0  | -2 | -3 | -2 | 1  | 2  | -1 | 5  | 0  | -2 | -1 | -1 | -1 | -1 | 1  |    |
| F | -2 | -3 | -3 | 3  | -3 | -3 | -3 | -1 | 0  | 0  | 3  | 0  | 6  | -4 | -2 | -2 | 1  | 3  | -1 |    |
| P | -1 | -2 | -2 | 1  | 3  | -1 | -2 | -2 | -3 | -3 | -1 | -2 | -4 | 7  | -1 | -1 | -4 | -3 | -2 |    |
| S | 1  | -1 | 1  | 0  | -1 | 0  | 0  | 0  | 1  | -2 | -2 | 0  | -1 | -2 | -1 | 4  | 1  | -3 | -2 | -2 |
| T | 0  | 1  | 0  | -1 | -1 | 1  | -2 | -1 | -2 | -1 | -1 | -1 | 1  | 5  | -2 | -2 | 0  |    |    |    |
| W | -3 | -3 | -4 | -4 | -4 | -2 | -2 | -3 | -2 | -3 | -2 | -3 | -1 | 1  | -4 | -3 | -2 | 11 | 2  | -3 |
| Y | -2 | -2 | -2 | 2  | -1 | -2 | -3 | 2  | -1 | -1 | -2 | -1 | 3  | -3 | -2 | -2 | 7  | -1 |    |    |
| 0  | -3 | -3 | -3 | -1 | -2 | -2 | -3 | -3 | 3  | 1  | -2 | 1  | -1 | -2 | -2 | 0  | -3 | 1  | 4  |    |

# BLOSUM Clustered Scoring Matrix in 1/2 Bit Units
# Blocks Database = /data/blocks_5.0/blocks.dat
# Cluster Percentage: >= 62
# Entropy = 0.6979, Expected = -0.5209
Linear vs. affine scoring

<table>
<thead>
<tr>
<th>Scoring</th>
<th>Gap</th>
<th>introduction</th>
<th>extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>-2</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>Affine</td>
<td>-3</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>

...and +1 for match

Seq1 G T A - - G - T - A
Seq2 - - A T G - A T G -

Linear -2 -2 1 -2 -2 (SUM = -7) -2 -2 1 -2 -2 (SUM = -7)
Affine -3 -1 1 -3 -1 (SUM = -7) -3 -3 1 -3 -3 (SUM = -11)

The algorithm

- Goal: find the maximal scoring alignment
- Scores: m match, -s 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 \( M[i, j] \)

\[
X_1 \ldots X_{i-1} - - X_i \\
Y_1 \ldots - Y_{j-1} Y_j -
\]
The algorithm

- Goal: find the maximal scoring alignment
- Scores: \(m\) match, \(-s\) mismatch, \(-g\) for insertion/deletion
- The best alignment for \(X[1...i]\) and \(Y[1...j]\) is called \(M[i, j]\)
- 3 ways to extend the alignment

\[
M[i,j] = \begin{cases} 
M[i-1, j-1] + m & \text{match} \\
M[i-1, j-1] - s & \text{mismatch} \\
M[i-1, j] - g & \text{for insertion/deletion} \\
M[i, j-1] - g & \text{for deletion} \\
M[i-1, j-1] - g & \text{for insertion} \\
\end{cases}
\]

The algorithm – final equation

Corresponds to:

\[
M[i, j] = \max \left\{ M[i-1, j-1] + \text{score}(X[i], Y[j]), M[i, j-1] - g, M[i-1, j] - g \right\}
\]
Field work: *global* alignment of two sequences

- Align two DNA sequences:
  - GAGTGA
  - GAGGCAGA (note the length difference)
- Parameters of the algorithm:
  - Match: $\text{score}(A,A) = 1$
  - Mismatch: $\text{score}(A,T) = -1$
  - Gap: $g = 2$

\[
M[i, j] = \max \begin{cases} 
M[i-1, j-1] \pm 1 \\
M[i, j-1] - 2 \\
M[i-1, j] - 2 
\end{cases}
\]

The algorithm. Step 1: init

- Create the matrix
- Initiation
  - 0 at [0,0]
  - Apply the equation...
The algorithm. Step 1: init

\[ M[i, j] = \max \begin{cases} 
  M[i-1, j-1] + 1 \\
  M[i, j-1] - 2 \\
  M[i-1, j] - 2 
\end{cases} \]

- Initiation of the matrix:
  - 0 at pos \([0,0]\)
  - Fill in the first row using the “←” rule
  - Fill in the first column using “↑”

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
<td>-8</td>
<td>-10</td>
<td>-12</td>
</tr>
<tr>
<td>G</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The algorithm. Step 2: fill in

\[ M[i, j] = \max \begin{cases} 
  M[i-1, j-1] + 1 \\
  M[i, j-1] - 2 \\
  M[i-1, j] - 2 
\end{cases} \]

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

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
<td>-8</td>
<td>-10</td>
<td>-12</td>
</tr>
<tr>
<td>G</td>
<td>-2</td>
<td>-1</td>
<td>-3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-4</td>
<td>-1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>-12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The algorithm. Step 2: fill in

\[ M[i, j] = \text{max} \begin{cases} M[i-1, j-1] \pm 1 \\ M[i, j-1] - 2 \\ M[i-1, j] - 2 \end{cases} \]

- We are done...
- Where’s the result?

\[
\begin{array}{cccccccc}
  & - & G & A & G & T & G & A \\
i & - & 0 & -2 & -4 & -6 & -8 & -10 & -12 \\
G & -2 & 1 & -1 & -3 & -5 & -7 & -9 \\
A & -4 & -1 & 2 & 0 & -2 & -4 & -6 \\
G & -6 & -3 & 0 & 3 & 1 & -1 & -3 \\
G & -8 & -5 & -2 & 1 & 2 & 2 & 0 \\
C & -10 & -7 & -4 & -1 & 0 & 1 & 1 \\
G & -12 & -9 & -6 & -3 & -2 & 1 & 0 \\
A & -14 & -11 & -8 & -5 & -4 & -1 & 2 \\
\end{array}
\]

The algorithm. Step 3: backtrace

- 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: backtrace

• Extract the alignments

a)
GAGT-GA
GAGGCCGA

b)
GA-GTGA
GAGGCCGA

<table>
<thead>
<tr>
<th></th>
<th>-</th>
<th>G</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>G</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td>a</td>
<td>b</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>0</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
<td>-8</td>
<td>-10</td>
</tr>
<tr>
<td>G</td>
<td>-2</td>
<td>1</td>
<td>-1</td>
<td>-3</td>
<td>-5</td>
<td>-7</td>
<td>-9</td>
</tr>
<tr>
<td>A</td>
<td>-4</td>
<td>-1</td>
<td>2</td>
<td>0</td>
<td>-2</td>
<td>-4</td>
<td>-6</td>
</tr>
<tr>
<td>G</td>
<td>-6</td>
<td>-3</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>-1</td>
<td>-3</td>
</tr>
<tr>
<td>G</td>
<td>-8</td>
<td>-5</td>
<td>-2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>-10</td>
<td>-7</td>
<td>-4</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>G</td>
<td>-12</td>
<td>-9</td>
<td>-6</td>
<td>-3</td>
<td>-2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>-14</td>
<td>-11</td>
<td>-8</td>
<td>-5</td>
<td>-4</td>
<td>-1</td>
<td>2</td>
</tr>
</tbody>
</table>

Variation on global alignment

• *Global* alignment: the previous algorithm is called *global* alignment, because it uses all letters from both sequences.

```
CAGCACTTGATTCTCGG
CAGC--------G-T----GG
```

• *Semi-global* alignment: don’t penalize for start/end gaps (omit the start/end of sequences).

```
CAGCA--CTTGGATTCTCGG
--------CAGCGTGG--------
```

– Applications of *semi-global*:
  – Finding a gene in genome
  – Placing marker onto a chromosome
  – One sequence much longer than the other

– Danger! – really bad alignments for divergent seqs

| seq X: | seq Y: |
Take-home message

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