Recap From Last Time:

- Bioinformatics is computer aided biology.
  - Deals with the collection, archiving, organization, and interpretation of a wide range of biological data.
- There are a large number of bioinformatics databases (see handout).
- The NCBI and EBI are major online bioinformatics service providers.
- Introduced via hands-on session the BLAST, Entrez, GENE, OMIM, UniProt, Muscle and PDB bioinformatics tools and databases.
  - Muddy point assessment (see results)
- Also covered: Course structure, Supporting course website, Ethics code, and Introductions...

Today’s Menu

<table>
<thead>
<tr>
<th>Classifying Databases</th>
<th>Primary, secondary and composite Bioinformatics databases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using Databases</td>
<td>Vignette demonstrating how major Bioinformatics databases intersect</td>
</tr>
<tr>
<td>Major Biomolecular Formats</td>
<td>How nucleotide and protein sequence and structure data are represented</td>
</tr>
<tr>
<td>Alignment Foundations</td>
<td>Introducing the why and how of comparing sequences</td>
</tr>
<tr>
<td>Alignment Algorithms</td>
<td>Hands-on exploration of alignment algorithms and applications</td>
</tr>
</tbody>
</table>

Primary, secondary & composite databases

Bioinformatics databases can be usefully classified into primary, secondary and composite according to their data source.

- **Primary databases** (or archival databases) consist of data derived experimentally.
  - GenBank: NCBI’s primary nucleotide sequence database.
  - PDB: Protein X-ray crystal and NMR structures.
- **Secondary databases** (or derived databases) contain information derived from a primary database.
  - RefSeq: non redundant set of curated reference sequences primarily from GenBank
  - PFAM: protein sequence families primarily from UniProt and PDB
- **Composite databases** (or metadatabases) join a variety of different primary and secondary database sources.
  - OMIM: catalog of human genes, genetic disorders and related literature
  - GENE: molecular data and literature related to genes with extensive links to other databases.
You have just come out a seminar about gastric cancer and one of your co-workers asks:

"What do you know about that 'Kras' gene the speaker kept taking about?"

You have some recollection about hearing of ‘Ras’ before. How would you find out more?

• Google?
• Library?
• Bioinformatics databases at NCBI and EBI!


Example Vignette Questions:

• What chromosome location and what genes are in the vicinity of a given query gene? NCBI GENE
• What can you find out about molecular functions, biological processes, and prominent cellular locations? EBI GO
• What amino acid positions in the protein are responsible for ligand binding? EBI UniProt
• What variants of this gene are associated with gastric cancer and other human diseases? NCBI OMIM
• What is known about the protein family, its species distribution, number in humans and residue-wise conservation? EBI PFAM
• Are high resolution protein structures available to examine the details of these mutations? How might we explain their potential molecular effects? RCSB PDB
1 AND 2  
\[ (1185 \text{ results}) \]  

1 OR 2  
\[ (134,872 \text{ results}) \]  

1 NOT 2  
\[ (84,448 \text{ results}) \]
Example Questions:
What chromosome location and what genes are in the vicinity?

Example Questions:
What 'molecular functions', 'biological processes', and 'cellular component' information is available?

Side-Note: Function, like beauty, is in the eye of the beholder...
GO: Gene Ontology
GO provides a controlled vocabulary of terms for describing gene product characteristics and gene product annotation data.

Why do we need Ontologies?
- Annotation is essential for capturing the understanding and knowledge associated with a sequence or other molecular entity.
- Annotation is traditionally recorded as “free text”, which is easy to read by humans, but has a number of disadvantages, including:
  - Difficult for computers to parse
  - Quality varies from database to database
  - Terminology used varies from annotator to annotator
- Ontologies are annotations using standard vocabularies that try to address these issues
- GO is integrated with UniProt and many other databases including a number at NCBI.

GO Ontologies
- There are three ontologies in GO:
  - **Biological Process**
    A commonly recognized series of events e.g. cell division, mitosis,
  - **Molecular Function**
    An elemental activity, task or job e.g. kinase activity, insulin binding
  - **Cellular Component**
    Where a gene product is located e.g. mitochondrion, mitochondrial membrane
The ‘Gene Ontology’ or GO is actually maintained by the EBI so let's switch or link over to UniProt also from the EBI.

UniProt will detail much more information for protein coding genes such as this one.

View FASTA file format
UniProt will detail much more information for protein coding genes.

Example Questions:

What positions in the protein are responsible for GTP binding?

What variants of this enzyme are involved in gastric cancer and other human diseases?

Are high resolution protein structures available to examine the details of these mutations?

Open link in a new tab!
Let's view the 3D structure:
Can we find where in the structure our mutations are located and infer their potential molecular effects?

View PDB file format

Let's view the 3D structure:
Can we find where in the structure our mutations are located and infer their potential molecular effects?

Back to UniProt:
What is known about the protein family, its species distribution, number in humans and residue-wise conservation, etc…?

PFAM is one of the best protein family databases

Example Questions:
What is known about the protein family, its species distribution, number in humans and residue-wise conservation, etc…?
Example Questions:
What is known about the protein family, its species distribution, number in humans and residue-wise conservation, etc…?
Today’s Menu

Classifying Databases
- Primary, secondary and composite Bioinformatics databases

Using Databases
- Vignette demonstrating how major Bioinformatics databases intersect

Major Biomolecular Formats
- How nucleotide and protein sequence and structure data are represented

Alignment Foundations
- Introducing the why and how of comparing sequences

Alignment Algorithms
- Hands-on exploration of alignment algorithms and applications

ALIGNMENT FOUNDATIONS

- Why…
  - Why compare biological sequences?

- What…
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)

- How…
  - Dot matrices
  - Dynamic programming
    - Global alignment
    - Local alignment
  - BLAST heuristic approach
ALIGNMENT FOUNDATIONS

• Why...
  ‣ Why compare biological sequences?

• What...
  ‣ Alignment view of sequence changes during evolution (matches, mismatches and gaps)

• How...
  ‣ Dot matrices
  ‣ Dynamic programing
    - Global alignment
    - Local alignment
  ‣ BLAST heuristic approach

Basic Idea: Display one sequence above another with spaces (termed gaps) inserted in both to reveal similarity of nucleotides or amino acids.

Seq1: C A T T C A C
Seq2: C T C G C A G C

Basic Idea: Display one sequence above another with spaces (termed gaps) inserted in both to reveal similarity of nucleotides or amino acids.

Seq1: C A T T C A C
Seq2: C T C G C A G C

Basic Idea: Display one sequence above another with spaces (termed gaps) inserted in both to reveal similarity of nucleotides or amino acids.

Seq1: C A T - T C A - C
Seq2: C - T C G C A G C

Two types of character correspondence

Add gaps to increase number of matches
Basic Idea: Display one sequence above another with spaces (termed gaps) inserted in both to reveal similarity of nucleotides or amino acids.

Why compare biological sequences?

- To obtain functional or mechanistic insight about a sequence by inference from another potentially better characterized sequence
- To find whether two (or more) genes or proteins are evolutionarily related
- To find structurally or functionally similar regions within sequences (e.g. catalytic sites, binding sites for other molecules, etc.)
- Many practical bioinformatics applications…

Practical applications include...

- Similarity searching of databases
  - Protein structure prediction, annotation, etc...
- Assembly of sequence reads into a longer construct such as a genomic sequence
- Mapping sequencing reads to a known genome
  - "Resequencing", looking for differences from reference genome - SNPs, indels (insertions or deletions)
  - Mapping transcription factor binding sites via ChIP-Seq (chromatin immuno-precipitation sequencing)
  - Pretty much all next-gen sequencing data analysis

N.B. Pairwise sequence alignment is arguably the most fundamental operation of bioinformatics!
Sequence changes during evolution

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- Mutations/Substitutions
- Deletions
- Insertions
Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.
- **Mutations/Substitutions**
  - CTCGTTA → CACGTTA
- **Deletions**
  - CACGTTA → CACTTA
- **Insertions**
  - CACTTA → CACTGTA

There are three major types of sequence change that can occur during evolution.
- **Mutations/Substitutions**
  - CTCGTTA → CACGTTA
- **Deletions**
  - CACGTTA → CACTTA
- **Insertions**
  - CACTTA → CACTGTA

There are three major types of sequence change that can occur during evolution.
- **Mutations/Substitutions**
  - CTCGTTA → CACGTTA
- **Deletions**
  - CACGTTA → CACTTA
- **Insertions**
  - CACTTA → CACTGTA

Alignments are great tools to visualize sequence similarity and evolutionary changes in homologous sequences.
- **Mismatches** represent mutations/substitutions
- **Gaps** represent insertions and deletions (indels)
Alternative alignments

• Unfortunately, finding the correct alignment is difficult if we do not know the evolutionary history of the two sequences

Q. Which of these 3 possible alignments is best?

1. CACTGTA
   CATGTTA
   ||:||:

2. CACTGTA
   CATGTTA
   ||:||:|

3. CACTGTA
   CATGTTA
   ||:|| |

Alternative alignments

• One way to judge alignments is to compare their number of matches, insertions, deletions and mutations

<table>
<thead>
<tr>
<th>Alignment</th>
<th>Matches</th>
<th>Mismatches</th>
<th>Gaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Scoring alignments

• We can assign a score for each match (+3), mismatch (+1) and indel (-1) to identify the optimal alignment for this scoring scheme

<table>
<thead>
<tr>
<th>Alignment</th>
<th>Matches</th>
<th>Mismatches</th>
<th>Gaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4 (+3)</td>
<td>3 (+1)</td>
<td>0 (-1) = 15</td>
</tr>
<tr>
<td>2</td>
<td>6 (+3)</td>
<td>0 (+1)</td>
<td>2 (-1) = 16</td>
</tr>
<tr>
<td>3</td>
<td>5 (+3)</td>
<td>1 (+1)</td>
<td>2 (-1) = 14</td>
</tr>
</tbody>
</table>

Optimal alignments

• Biologists often prefer parsimonious alignments, where the number of postulated sequence changes is minimized.

<table>
<thead>
<tr>
<th>Alignment</th>
<th>Matches</th>
<th>Mismatches</th>
<th>Gaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>6</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

0 mismatches
0 gaps
Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the number of postulated sequence changes is minimized.

<table>
<thead>
<tr>
<th></th>
<th>4 matches</th>
<th>3 mismatches</th>
<th>0 gaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATGTTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTGTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA-TGTTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CATGT-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTG-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTGT-A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CATGT-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Warning:** There may be more than one optimal alignment and these may not reflect the true evolutionary history of our sequences!

Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the number of postulated sequence changes is minimized.

<table>
<thead>
<tr>
<th></th>
<th>4 matches</th>
<th>3 mismatches</th>
<th>0 gaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATGTTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTGTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA-TGTTA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CATGT-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTG-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CACTGT-A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CATGT-TA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ALIGNMENT FOUNDATIONS

- **Why**...
  - Why compare biological sequences?
- **What**...
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How**...
  - Dot matrices
  - Dynamic programming
    - Global alignment
    - Local alignment
  - BLAST heuristic approach
ALIGNMENT FOUNDATIONS

• Why...
  • Why compare biological sequences?

• What...
  • Alignment view of sequence changes during evolution (matches, mismatches and gaps)

• How...
  ‣ Dot matrices
  ‣ Dynamic programming
    - Global alignment
    - Local alignment
  ‣ BLAST heuristic approach

How do we compute the optimal alignment between two sequences?

Dot plots: simple graphical approach

• Place one sequence on the vertical axis of a 2D grid (or matrix) and the other on the horizontal

Dot plots: simple graphical approach

• Now simply put dots where the horizontal and vertical sequence values match

Dot plots: simple graphical approach

• Diagonal runs of dots indicate matched segments of sequence
Dot plots: simple graphical approach

Q. What would the dot matrix of a two identical sequences look like?

A
C
G
C
G

A
C
G
C
G

Dot plots: window size and match stringency

Solution: use a window and a threshold
– compare character by character within a window
– require certain fraction of matches within window in order to display it with a dot.
• You have to choose window size and stringency

A
C
G
C
G

A
C
G
C
G

Filter
Window = 3
Stringency = 3

A
C
G
C
G

A
C
G
C
G

Filter
Window = 3
Stringency = 2
Window size = 5 bases

A dot plot simply puts a dot where two sequences match. In this example, dots are placed in the plot if 5 bases in a row match perfectly. Requiring a 5 base perfect match is a heuristic – only look at regions that have a certain degree of identity.

Do you expect evolutionarily related sequences to have more word matches (matches in a row over a certain length) than random or unrelated sequences?

Web site used: http://www.vivo.colostate.edu/molkit/dnadot/

Window size = 7 bases

This is a dot plot of the same sequence pair. Now 7 bases in a row must match for a dot to be placed. Noise is reduced.

Using windows of a certain length is very similar to using words (kmers) of N characters in the heuristic alignment search tools.

Bigger window (kmer) fewer matches to consider

Web site used: http://www.vivo.colostate.edu/molkit/dnadot/

Ungapped alignments

Only diagonals can be followed.

Downward or rightward paths represent insertion or deletions (gaps in one sequence or the other).

Web site used: http://www.vivo.colostate.edu/molkit/dnadot/

Uses for dot matrices

- Visually assessing the similarity of two protein or two nucleic acid sequences

- Finding local repeat sequences within a larger sequence by comparing a sequence to itself
  - Repeats appear as a set of diagonal runs stacked vertically and/or horizontally
Human LDL receptor protein sequence (Genbank P01130)

W = 1
S = 1

(Figure from Mount, “Bioinformatics sequence and genome analysis”)

Human LDL receptor protein sequence (Genbank P01130)

W = 23
S = 7

(Figure from Mount, “Bioinformatics sequence and genome analysis”)

Your Turn!

Exploration of dot plot parameters (hands-on worksheet Section 1)

ALIGNMENT FOUNDATIONS

• Why...
  • Why compare biological sequences?
• What...
  • Alignment view of sequence changes during evolution (matches, mismatches and gaps)
• How...
  • Dot matrices
    ‣ Dynamic programming
      - Global alignment
      - Local alignment
    ‣ BLAST heuristic approach
The Dynamic Programming Algorithm

- The dynamic programming algorithm can be thought of an extension to the dot plot approach
  - One sequence is placed down the side of a grid and another across the top
  - Instead of placing a dot in the grid, we compute a score for each position
  - Finding the optimal alignment corresponds to finding the path through the grid with the best possible score

Different paths represent different alignments

Matches are represented by diagonal paths & indels with horizontal or vertical path segments

Algorithm of Needleman and Wunsch

- The Needleman–Wunsch approach to global sequence alignment has three basic steps:
  1. setting up a 2D-grid (or alignment matrix),
  2. scoring the matrix, and
  3. identifying the optimal path through the matrix

Scoring the alignment matrix

- Start by filling in the first row and column – these are all indels (gaps).
  - Each step you take you will add the gap penalty to the score \( (S_{i,j}) \) accumulated in the previous cell

Scoring the alignment matrix

• Start by filling in the first row and column – these are all indels (gaps).
  – Each step you take you will add the gap penalty to the score \( S_{i,j} \) accumulated in the previous cell

\[
\begin{array}{cccccc}
  & - & D & P & L & E \\
 0 & -2 & -4 & -6 & -8 \\
D & -2 & 0 & -2 & -4 & -6 & -8 \\
P & -4 & -2 & 0 & -2 & -4 & -6 & -8 \\
M & -6 & -4 & -2 & 0 & -2 & -4 & -6 & -8 \\
E & -8 & -6 & -4 & -2 & 0 & -2 & -4 & -6 & -8 \\
\end{array}
\]

\[ S_{i4} = (-2) + (-2) + (-2) + (-2) \]

\[ \text{Seq1: DPME} \]
\[ \text{Seq2: } ---- \]

\[ \text{Scores: match = +1, mismatch = -1, gap = -2} \]

Scoring the alignment matrix

• Then go to the empty corner cell (upper left). It has filled in values in up, left and diagonal directions
  – Now can ask which of the three directions gives the highest score?
  – keep track of this score and direction

\[ S(i, j) = \max \left\{ S(i-1, j-1) + \text{(mis)match}, S(i-1, j) + \text{gap penalty}, S(i, j-1) + \text{gap penalty} \right\} \]

\[ \text{Scores: match = +1, mismatch = -1, gap = -2} \]
Scoring the alignment matrix

• At each step, the score in the current cell is determined by the scores in the neighboring cells
  – The maximal score and the direction that gave that score is stored (we will use these later to determine the optimal alignment)

<table>
<thead>
<tr>
<th></th>
<th>D</th>
<th>P</th>
<th>L</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
<td>-8</td>
</tr>
<tr>
<td>D</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
</tr>
<tr>
<td>P</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
</tr>
<tr>
<td>M</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
</tr>
<tr>
<td>E</td>
<td>-15</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
</tr>
</tbody>
</table>

Scores: match = +1, mismatch = -1, gap = -2

1. (-2)+(-1) = -3 <= (D-P) mismatch!
2. (-4)+(-2) = -6 <= (D-M) mismatch!
3. (-1)+(-2) = -3 <= (D-E) mismatch!

Alignment

Scoring the alignment matrix

• For the highlighted cell, the corresponding score ($S_{ij}$) refers to the score of the optimal alignment of the first $i$ characters from sequence1, and the first $j$ characters from sequence2.

<table>
<thead>
<tr>
<th></th>
<th>D</th>
<th>P</th>
<th>L</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
<td>-8</td>
</tr>
<tr>
<td>D</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
</tr>
<tr>
<td>P</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
</tr>
<tr>
<td>M</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
</tr>
<tr>
<td>E</td>
<td>-15</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
</tr>
</tbody>
</table>

Scores: match = +1, mismatch = -1, indel = -2

1. (-1)+(-1) = -2 <= (D-D) mismatch!
2. (-3)+(-2) = -5 <= (D-P) mismatch!
3. (2)+(-2) = 0 <= (D-M) mismatch!

Alignment

Scoring the alignment matrix

• At each step, the score in the current cell is determined by the scores in the neighboring cells
  – The maximal score and the direction that gave that score is stored

<table>
<thead>
<tr>
<th></th>
<th>D</th>
<th>P</th>
<th>L</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
<td>-8</td>
</tr>
<tr>
<td>D</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
</tr>
<tr>
<td>P</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
</tr>
<tr>
<td>M</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
</tr>
<tr>
<td>E</td>
<td>-15</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
</tr>
</tbody>
</table>

Scores: match = +1, mismatch = -1, indel = -2

1. (2)+(-1) = 1 <= mismatch
2. (0)+(-2) = -2 <= mismatch
3. (0)+(-2) = -2 <= mismatch

Alignment

Scoring the alignment matrix

• We will continue to store the alignment score ($S_{ij}$) for all possible alignments in the alignment matrix.

<table>
<thead>
<tr>
<th></th>
<th>D</th>
<th>P</th>
<th>L</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
<td>-8</td>
</tr>
<tr>
<td>D</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
<td>-9</td>
</tr>
<tr>
<td>P</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
<td>-10</td>
</tr>
<tr>
<td>M</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
<td>-11</td>
</tr>
<tr>
<td>E</td>
<td>-15</td>
<td>-14</td>
<td>-13</td>
<td>-12</td>
</tr>
</tbody>
</table>

Scores: match = +1, mismatch = -1, gap = -2

1. (-4)+(-1) = -5 <= (D-L) mismatch
2. (-6)+(-2) = -8 <= (D-PL) mismatch
3. (-1)+(-2) = -3 <= (D-E) mismatch

Alignment
Scoring the alignment matrix

- The score of the best alignment of the entire sequences corresponds to $S_{n,m}$ where $n$ and $m$ are the length of the sequences.

Scores: match = +1, mismatch = -1, indel = -2

Scores: match = +1, mismatch = -1, indel = -2

Questions:

- What is the optimal score for the alignment of these sequences and how do we find the optimal alignment?

Alignments:

DPME
DPLE

DPME
DPLE
Questions:
- To find the best alignment we retrace the arrows starting from the bottom right cell.

The alignment and score are dependent on the scoring system
- Here we increase the gap penalty from -2 to -3

More than one alignment possible
- Sometimes more than one alignment can result in the same optimal score

Key point: Optimal alignment solutions and their scores are not necessarily unique and depend on the scoring system!
Your Turn!

Hands-on worksheet **Sections 2 & 3**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>T</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</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>
</tr>
<tr>
<td>T</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>
</tr>
<tr>
<td>G</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>
</tr>
</tbody>
</table>

**Match:** +2  
**Mismatch:** -1  
**Gap:** -2

**NW DYNAMIC PROGRAMMING**

```
A  G  T  T  C
A  -2 +2  0  -2  -4
T  -4  0  +1  +2  0  -6
T  -6 -2 -1  +3  +4  +2
G  -8 -4  0  +1  +2  +3
C -10 -6 -2 -1  0  +4
```

**Match:** +2  
**Mismatch:** -1  
**Gap:** -2

ALIGNMENT FOUNDATIONS

- **Why...**
  - Why compare biological sequences?
- **What...**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How...**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

**Global vs local alignments**

- **Needleman-Wunsch is a global alignment algorithm**
  - Resulting alignment spans the complete sequences end to end
  - This is appropriate for closely related sequences that are similar in length
- **For many practical applications we require local alignments**
  - Local alignments highlight sub-regions (e.g. protein domains) in the two sequences that align well
Local alignment: Definition

• Smith & Waterman proposed simply that a local alignment of two sequences allow arbitrary-length segments of each sequence to be aligned, with no penalty for the unaligned portions of the sequences. Otherwise, the score for a local alignment is calculated the same way as that for a global alignment.


The Smith-Waterman algorithm

• Three main modifications to Needleman-Wunsch:
  – Allow a node to start at 0
  – The score for a particular cell cannot be negative
    • if all other score options produce a negative value, then a zero must be inserted in the cell
  – Record the highest-scoring node, and trace back from there

$$S(i, j) = \max \begin{cases} S(i-1, j-1) + \text{match} \\ S(i-1, j) - \text{gap penalty} \\ S(i, j-1) - \text{gap penalty} \\ 0 \end{cases}$$

Local alignments can be used for database searching

• Goal: Given a query sequence (Q) and a sequence database (D), find a list of sequences from D that are most similar to Q
  – Input: Q, D and scoring scheme
  – Output: Ranked list of hits

**Input**

<table>
<thead>
<tr>
<th>Query sequence</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTATGGTCA</td>
<td>100</td>
</tr>
<tr>
<td>GTATGGTCA</td>
<td>90</td>
</tr>
<tr>
<td>TATGGTCA</td>
<td>40</td>
</tr>
<tr>
<td>CGATCTGCA</td>
<td>38</td>
</tr>
<tr>
<td>TCGTTGCTA</td>
<td></td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Score</th>
<th>Ranked hit list</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>GTATGGTCA</td>
<td>Ras</td>
</tr>
<tr>
<td>90</td>
<td>TATGGTCA</td>
<td>Ras</td>
</tr>
<tr>
<td>40</td>
<td>CGATCTGCA</td>
<td>HSP90</td>
</tr>
<tr>
<td>38</td>
<td>TCGTTGCTA</td>
<td>P450</td>
</tr>
</tbody>
</table>
The database search problem

- Due to the rapid growth of sequence databases, search algorithms have to be both efficient and sensitive
  - Time to search with SW is proportional to $m \times n$ ($m$ is length of query, $n$ is length of database), too slow for large databases!

To reduce search time heuristic algorithms, such as BLAST, first remove database sequences without a strong local similarity to the query sequence in a quick initial scan.

ALIGNMENT FOUNDATIONS

- Why...
  - Why compare biological sequences?
- What...
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- How...
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

Rapid, heuristic versions of Smith–Waterman: BLAST

- BLAST (Basic Local Alignment Search Tool) is a simplified form of Smith-Waterman (SW) alignment that is popular because it is fast and easily accessible
  - BLAST is a heuristic approximation to SW - It examines only part of the search space
  - BLAST saves time by restricting the search by scanning database sequences for likely matches before performing more rigorous alignments
  - Sacrifices some sensitivity in exchange for speed
  - In contrast to SW, BLAST is not guaranteed to find optimal alignments
Rapid, heuristic versions of Smith–Waterman: BLAST

• BLAST (Basic Local Alignment Search Tool) is a simplified form of Smith-Waterman (SW) alignment that is popular because it is fast and easily accessible.
  – BLAST finds regions of similarity between biological sequences.
  – BLAST saves time by restricting the search by scanning database sequences for likely matches before performing more rigorous alignments.
  – BLAST sacrifices some sensitivity in exchange for speed.
  – In contrast to SW, BLAST is not guaranteed to find optimal alignments.

"The central idea of the BLAST algorithm is to confine attention to sequence pairs that contain an initial word match." - Altschul et al. (1990)

How BLAST works

• Four basic phases
  – **Phase 1**: compile a list of query word pairs (w=3)

    Query sequence
    
    RGGVKRI
    RGG
    GGV
    GVK
    VKR
    KRI

    generate list of w=3 words for query

  – **Phase 2**: expand word pairs to include those similar to query (defined as those above a similarity threshold to original word, i.e. match scores in substitution matrix)

    Query sequence
    
    RGGVKRI
    RGG
    RAG
    RIG
    RLG
    ...
    GGV
    GAV
    GTV
    GCV
    ...
    GVK
    GAK
    GIK
    GGK
    ...
    VKR
    VRR
    VHR
    VER
    ...
    KRI
    KKI
    KHI
    KDI
    ...

    extend list of words similar to query

  – **Phase 3**: search database sequences for similar matches

  – **Phase 4**: refine alignments of similar matches

• BLAST uses this pre-screening heuristic approximation resulting in an approach that is about 50 times faster than the Smith-Waterman algorithm.
Blast

**Phase 3:** a database is scanned to find sequence entries that match the compiled word list

**Phase 4:** the initial database hits are extended in both directions using dynamic programming
BLAST output

• BLAST returns the highest scoring database hits in a ranked list along with details about the target sequence and alignment statistics

<table>
<thead>
<tr>
<th>Description</th>
<th>Max score</th>
<th>Query cover</th>
<th>E value</th>
<th>Max ident</th>
<th>Accession</th>
</tr>
</thead>
<tbody>
<tr>
<td>kinesin-1 heavy chain [Homo sapiens]</td>
<td>677</td>
<td>100%</td>
<td>0</td>
<td>100%</td>
<td>NP_004512.1</td>
</tr>
<tr>
<td>Kif5b protein [Mus musculus]</td>
<td>676</td>
<td>100%</td>
<td>0</td>
<td>98%</td>
<td>AAA20133.1</td>
</tr>
<tr>
<td>Kinesin-14 heavy chain [Danio rerio]</td>
<td>595</td>
<td>88%</td>
<td>0</td>
<td>78%</td>
<td>XP_00320703</td>
</tr>
<tr>
<td>hypothetical protein EGK_18589</td>
<td>48.2</td>
<td>40%</td>
<td>0.03</td>
<td>32%</td>
<td>ELK35081.1</td>
</tr>
<tr>
<td>mKIAA4102 protein [Mus musculus]</td>
<td>42.7</td>
<td>38%</td>
<td>3.02</td>
<td>24%</td>
<td>EHH28205.1</td>
</tr>
</tbody>
</table>

Statistical significance of results

• An important feature of BLAST is the computation of statistical significance for each hit. This is described by the E value (expect value)

<table>
<thead>
<tr>
<th>Description</th>
<th>Max score</th>
<th>Query cover</th>
<th>E value</th>
<th>Max ident</th>
<th>Accession</th>
</tr>
</thead>
<tbody>
<tr>
<td>kinesin-1 heavy chain [Homo sapiens]</td>
<td>677</td>
<td>100%</td>
<td>0</td>
<td>100%</td>
<td>NP_004512.1</td>
</tr>
<tr>
<td>Kif5b protein [Mus musculus]</td>
<td>676</td>
<td>100%</td>
<td>0</td>
<td>98%</td>
<td>AAA20133.1</td>
</tr>
<tr>
<td>Kinesin-14 heavy chain [Danio rerio]</td>
<td>595</td>
<td>88%</td>
<td>0</td>
<td>78%</td>
<td>XP_00320703</td>
</tr>
<tr>
<td>hypothetical protein EGK_18589</td>
<td>48.2</td>
<td>40%</td>
<td>0.03</td>
<td>32%</td>
<td>ELK35081.1</td>
</tr>
<tr>
<td>mKIAA4102 protein [Mus musculus]</td>
<td>42.7</td>
<td>38%</td>
<td>3.02</td>
<td>24%</td>
<td>EHH28205.1</td>
</tr>
</tbody>
</table>

BLAST scores and E-values

• The E value is the expected number of hits that are as good or better than the observed local alignment score (with this score or better) if the query and database are random with respect to each other – i.e. the number of alignments expected to occur by chance with equivalent or better scores

• Typically, only hits with E value below a significance threshold are reported
  – This is equivalent to selecting alignments with score above a certain score threshold
• Ideally, a threshold separates all query related sequences (yellow) from all unrelated sequences (gray)

• Unfortunately, often both score distributions overlap
  – The E value describes the expected number of hits with a score above the threshold if the query and database are unrelated

• Unfortunately, often both score distributions overlap
  – The E value describes the expected number of hits with a score above the threshold if the query and database are unrelated

<table>
<thead>
<tr>
<th>Description</th>
<th>Max score</th>
<th>Query cover</th>
<th>E value</th>
<th>Max ident</th>
<th>Accession</th>
</tr>
</thead>
<tbody>
<tr>
<td>kinesin-1 heavy chain [Homo sapiens]</td>
<td>677</td>
<td>100%</td>
<td>0</td>
<td>100%</td>
<td>NP_004512.1</td>
</tr>
<tr>
<td>Kif5b protein [Mus musculus]</td>
<td>676</td>
<td>100%</td>
<td>0</td>
<td>98%</td>
<td>AAA20133.1</td>
</tr>
<tr>
<td>Kinesin-14 heavy chain [Danio rerio]</td>
<td>595</td>
<td>88%</td>
<td>0</td>
<td>78%</td>
<td>XP_00320703</td>
</tr>
<tr>
<td>hypothetical protein EGK_18589</td>
<td>42.7</td>
<td>40%</td>
<td>0.03</td>
<td>32%</td>
<td>ELK35081.1</td>
</tr>
</tbody>
</table>

A score of 42.7 or better is expected to occur by chance 3 in 100 times (E-value = 0.03)
### Practical database searching with BLAST

To find out more about $E$ values see: “The Statistics of Sequence Similarity Scores” available in the help section of the NCBI BLAST site:


In general $E$ values < 0.005 are usually significant.

### Your Turn!

**Hands-on worksheet Sections 4 & 5**

- Please do answer the last lab review question (Q19).
- We encourage discussion and exploration!

---

<table>
<thead>
<tr>
<th>Description</th>
<th>Max score</th>
<th>Total score</th>
<th>Query cover</th>
<th>E value</th>
<th>Max ident</th>
<th>Accession</th>
</tr>
</thead>
<tbody>
<tr>
<td>kinesin-1 heavy chain [Homo sapiens]</td>
<td>677</td>
<td>677</td>
<td>100%</td>
<td>0</td>
<td>100%</td>
<td>NP_004512.1</td>
</tr>
<tr>
<td>Kif5b protein [Mus musculus]</td>
<td>676</td>
<td>676</td>
<td>100%</td>
<td>0</td>
<td>98%</td>
<td>AAA30153.1</td>
</tr>
<tr>
<td>Kinesin-14 heavy chain [Danio rerio]</td>
<td>595</td>
<td>595</td>
<td>88%</td>
<td>0</td>
<td>78%</td>
<td>XP_00320703</td>
</tr>
<tr>
<td>Hypothetical protein EGK_18589</td>
<td>42.7</td>
<td>52</td>
<td>40%</td>
<td>0.03</td>
<td>32%</td>
<td>ELK35081.1</td>
</tr>
</tbody>
</table>

### Practical database searching with BLAST

- There are four basic components to a traditional BLAST search
  - (1) Choose the sequence (query)
  - (2) Select the BLAST program
  - (3) Choose the database to search
  - (4) Choose optional parameters
- Then click “BLAST”
Step 1: Choose your sequence

- Sequence can be input in FASTA format or as accession number

Step 2: Choose the BLAST program

DNA potentially encodes six proteins

```
5’ CAT CAA
5’ ATC AAC
5’ TCA ACT

5’ CAT CACT CACT CCAAAAAGACACCCCTACACATCAACACAC CCCAC
3’
3’ GTAGTTGATGTTGAGGTTTCTGTGGGAATGTGTAGTTGTTTGGATGGGTG
```

```
5’ GTG GGT
5’ TGG GTA
5’ GGG TAG
```
**Step 3: Choose the database**

nr = non-redundant (most general database)
dbest = database of expressed sequence tags
dbsts = database of sequence tag sites
gss = genomic survey sequences

**Step 4a: Select optional search parameters**

- choose the organism to search
- change the substitution matrix
- change the expect (E) value
- change the word size
- change the output format
Different output formats are available

E.g. Query anchored alignments

... and alignments with dots for identities

Common problems

• Selecting the wrong version of BLAST
• Selecting the wrong database
• Too many hits returned
• Too few hits returned
• Unclear about the significance of a particular result - are these sequences homologous?
### How to handle too many results

- **Focus on the question you are trying to answer**
  - select “refseq” database to eliminate redundant matches from “nr”
  - Limit hits by organism
  - Use just a portion of the query sequence, when appropriate
  - Adjust the expect value; lowering $E$ will reduce the number of matches returned

### How to handle too few results

- **Many genes and proteins have no significant database matches**
  - remove Entrez limits
  - raise E-value threshold
  - search different databases
  - try scoring matrices with lower BLOSUM values (or higher PAM values)
  - use a search algorithm that is more sensitive than BLAST (e.g. PSI-BLAST or HMMer)

### Summary of key points

- Sequence alignment is a fundamental operation underlying much of bioinformatics.
- Even when optimal solutions can be obtained they are not necessarily unique or reflective of the biologically correct alignment.
- Dynamic programming is a classic approach for solving the pairwise alignment problem.
- Global and local alignment, and their major application areas.
- Heuristic approaches are necessary for large database searches and many genomic applications.

---

**FOR NEXT CLASS…**

Check out the online:

- **Reading**: Sean Eddy’s “What is dynamic programming?”
- **Homework**: (1) Quiz, (2) Alignment Exercise.
Homework Grading
Both (1) quiz questions and (2) alignment exercise carry equal weights (i.e. 50% each).

(Homework 2) Assessment Criteria

<table>
<thead>
<tr>
<th>Points</th>
<th>(Homework 2) Assessment Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup labeled alignment matrix</td>
</tr>
<tr>
<td>1</td>
<td>Include initial column and row for GAPs</td>
</tr>
<tr>
<td>1</td>
<td>All alignment matrix elements scored (i.e. filled in)</td>
</tr>
<tr>
<td>1</td>
<td>Evidence for correct use of scoring scheme</td>
</tr>
<tr>
<td>1</td>
<td>Direction arrows drawn between all cells</td>
</tr>
<tr>
<td>1</td>
<td>Evidence of multiple arrows to a given cell if appropriate</td>
</tr>
<tr>
<td>1</td>
<td>Correct optimal score position in matrix used</td>
</tr>
<tr>
<td>1</td>
<td>Correct optimal score obtained for given scoring scheme</td>
</tr>
<tr>
<td>1</td>
<td>Traceback path(s) clearly highlighted</td>
</tr>
<tr>
<td>1</td>
<td>Correct alignment(s) yielding optimal score listed</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>