BIOS 477/877
Bioinformatics and Molecular Evolution
Lecture 14

TODAY'S TOPICS

- BLAST & FASTA statistics
- Multiple Alignment
  - ClustalW
- Assignment 7

BLAST Statistics

- Karlin-Altschul equation (Karlin & Altschul, 1990)
  - For a pairwise alignment
    \[ P = K_{m,n} e^{-kS} \] (Lec 11 slide 7)
    - \( m, n \): lengths of the sequences compared
    - \( K \): search space
  - For database similarity searching
    \[ E = K_{m,n} e^{-kS} \] (NOTE: \( E = NP \) is not used in BLAST)
    - \( m \): length of the query
    - \( n \): length of the database (total number of residues)
    - \( E \): value: the expected number of HSPs with scores \( \geq S \)
    - \( P = 1 - e^{-E} \) (\( P \approx E < 0.01 \))
      - the probability of having at least one HSP with its score \( \geq S \)

P-value, E-value, and database search

- P-value for pairwise alignment = \( 1 - \exp\{-K_{m,n} e^{-kS}\} = K_{m,n} e^{-kS} \)
  - Probability of getting the alignment score \( \geq S \) from random pairwise comparison (\( m \) and \( n \) are the lengths of the two sequences compared)
- E-value = \( K_{m,n} e^{-kS} \)
  - Number of alignments with a score \( \geq S \) expected by chance from a database search
    - \( m \): length of the query (or effective length, \( m' \))
    - \( n \): length of the database (or effective length, \( n' \))
- P-value for a database search (Bonferroni corrected)
  - \( P = 1 - e^{-E} \) or \( E = -\ln(1 - P) \)
  - \( 0 < P < 1 \)
  - \( 0 < E < N \) (\( N \): number of random comparisons)
  - Altschul et al. (1994) & BLAST Statistics Tutorial
  - \( P = E/N \) or \( E = PN \) is used in FASTA; \( N \): database size

BLAST Statistics

Raw Score \((S)\): simply based on pairwise scores & gap penalties

Normalized Score or Bit Score \((S'_{bit})\):
\[ S'_{bit} = (\lambda S - \log_2 K) / \log_2 2, \quad [S'_{bit}] = \lambda S - \log_2 K \]
\( \lambda = 0.267, K = 0.841, S'_{bit} = (0.267 \times 465 - \log_2(0.841)) / \log_2 2 = 183.7 \)

Note: Calculation methods for length adjustment \((\ell)\) and \(m'n'\) have been changed based on a new finite-size correction (FSC). See Park et al. (2012, BMC Research Note)

BLAST Statistics

- Karlin-Altschul equation (Karlin & Altschul, 1990)
  \[ E = K_{m,n} e^{-kS} \]
  - \( m' \): effective length of the query
  - \( n' \): effective length of the database
  - \( m' = m - l \)
  - \( n' = n - l \times (\text{number of sequences in the database}) \)
  - \( l \): length adjustment
    - correction for edge effects
    - HSPs cannot occur too close to the search space edges.
    - HSPs need to be a certain length.

P-value for a database search (Bonferroni corrected)
\[ P = 1 - e^{-E} \]
\( 0 < P < 1 \)
\( 0 < E < N \)
\( N \): number of random comparisons
**BLAST Statistics**

[blastp HSP]

Mothers against decapentaplegic-like protein 4 [Tribolium castaneum]

<table>
<thead>
<tr>
<th>Range</th>
<th>0-690</th>
<th>690-1043</th>
<th>1043-1593</th>
<th>1593-2029</th>
<th>2029-2461</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Target</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Identities</td>
<td>100%</td>
<td>100%</td>
<td>87%</td>
<td>87%</td>
<td>87%</td>
</tr>
<tr>
<td>Expect</td>
<td>0.041</td>
<td>0.041</td>
<td>0.041</td>
<td>0.041</td>
<td>0.041</td>
</tr>
</tbody>
</table>

$m' = m - l = 570 - 161 = 409$ (effective length of query)

$E = 0.041 x 570 x 53,472,480,294 x e^{-0.267 x 465} = 2.49 x 10^{-42}$

$E = 0.041 x 570 x 53,472,480,294 x 2^{183} = 3.09 x 10^{-42}$

$P = 1 - e^{-E} = 1 - e^{-3.09 x 10^{-42}} = 1$ (if $E < 0.01$)

$|E| = 3e-48 (3x10^{-48})$
**BLAST search size and format options**

Before search: Restrict a search against the selected organism or limit by Entrez Query. Search space will be limited. E-values become smaller.

After search: Restrict the result shown for a selected organism or limit by Entrez Query. Search space is not affected.

**FASTA**

- [http://fasta.bioch.virginia.edu/fasta_www2/fasta_list2.shtml](http://fasta.bioch.virginia.edu/fasta_www2/fasta_list2.shtml) (includes also SSEARCH)
- [http://www.ebi.ac.uk/Tools/sss/fasta/](http://www.ebi.ac.uk/Tools/sss/fasta/) (includes also SSEARCH)
- With graphic output
- Results can be obtained through email
- [http://fasta.genome.jp/](http://fasta.genome.jp/)

**FASTA Similarity Search**

- Default DB: PIR (small, limited)
- SwissProt for better results

**FASTA Statistics**

Distribution based on database sequences can be used instead of shuffling a sequence.
**FASTA Statistics**

*Observed distribution* vs. *Expected distribution*

- **Smith-Waterman score**
- **Against**: SwissProt (~84,000 seq)

---

**Distribution based on database sequences**
can be used instead of shuffling a sequence

\[
P(S \geq x) = 1 - \exp\left[\frac{-K_{\text{me}}}{l}x\right]
\]

---

**Pairwise alignments between the query vs. all database entries are used to generate the null score distribution**

\[
E(553,474) = P \times 553,474
\]

---

**Optimum raw score**

- Standardized score: \(P = E / 553,474\)

- Bit score: \(P = 3.5 \times 10^{-15} / 553,474 = 6.32 \times 10^{-21}\)

---

**Consistent domains in both sequences**

- **High scores across the region**
FASTA Statistics

Only SH3 domain region has a high score, and aligned correctly.
Other domain regions do not contribute to the score but alignment is overextended.

See Mills and Pearson (2013)

FASTA Similarity Search

http://www.ebi.ac.uk/Tools/ss/s/fasta

FASTA Functional Prediction

Multiple alignment as an extension of pairwise alignment

Dynamic programming algorithm
Guarantees to find the optimal alignment based on the scoring system
Optimal alignments are searched based on alignment score
Match/mismatch ($S_{ij}$) and gap penalties

Why multiple alignments?

- To examine evolutionary relationships between sequences
- To reconstruct phylogenetic trees
- To identify protein families
- To predict protein functions (conserved regions, functional or structural domains)
- For homology modeling (structural prediction)
- To identify gene structures (exons, introns)
- To design PCR primers

etc. etc. ...
**Multiple alignment: complexity**

- **Dynamic programming algorithm**
  - Pairwise alignment
  - Multiple alignment (3x)

  ![Search Space](image)

  $O(mn)$ or $O(n^2)$

  Complexity can be expressed with big-O notation

  $O(n^3)$

**How to score multiple alignment**

- **Sum of pairs score**
  \[ S(A) = \sum i,j S(A_{ij}) \]

  $A_{ij}$: the score of the pairwise alignment between $i$ and $j$

  \[ S(4) = S(4_{12}) + S(4_{13}) + S(4_{23}) \]

  $S(\Lambda)$ has no statistical justification

  There is no single good method that can measure the overall quality of multiple alignments!

**Multiple alignment: Divide and Conquer (DCA)**

- Breaks down a problem into small sub-problems,
- Solves each sub-problem independently,
- Combines the solutions to the sub-problems to give a solution to the original large problem.


[https://bibiserv.cebitec.uni-bielefeld.de/dca](https://bibiserv.cebitec.uni-bielefeld.de/dca)

For multiple alignment:
1. Each sequence is cut into two.
2. Each group of subsequences is recursively cut into two until they are short enough.
3. Each group of short subsequence is aligned optimally (dynamic programming algorithm).
4. Short alignments are concatenated, yielding a solution of the original multiple alignment problem.
**Multiple alignment: Divide and Conquer (DCA)**

DCA:
- Uses (almost) exact multiple alignment method on shorter fragments
- How can the suitable cutting positions be found?

DEALIGN (local fragment alignment) can be used to find cutting positions (based on highly conserved fragments)


[Not implemented.]

http://bibiserv.techfak.uni-bielefeld.de/dalign/

**Multiple alignment: Heuristic methods**

**Pairwise alignment:**
- fast approximation or full dynamic programming
- Generate a distance matrix (% identities converted to distances)
- Construct a guide tree (neighbor-joining phylogenetic tree)
- Progressive alignment following the guide tree (scoring matrix, sequence weight, gap penalties, etc.)

**Multiple alignment: ClustalW**

**Pairwise alignment:**
- fast approximation

**Generate a distance matrix from % identities**

**Multiple alignment: ClustalW**

**Pairwise alignment:**
- fast approximation

**Generate a distance matrix from % identities**
**Pairwise alignment**

- Fast approximation or full dynamic programming
- Generate a distance matrix (% identities converted to distances)
- Construct a guide tree (neighbor-joining phylogenetic tree)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**


**Sequence weights: based on branch lengths**

- Shorter branch lengths ➔ Fewer changes
  - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

  - Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

---

**Pairwise alignment**

- Fast approximation or full dynamic programming
- Generate a distance matrix (% identities converted to distances)
- Construct a guide tree (neighbor-joining phylogenetic tree)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.

---

**Multiple alignment: ClustalW**

- Sequence weights: based on branch lengths
  - Shorter branch lengths ➔ Fewer changes
    - (e.g., S1 and S2 are more similar than S1 and S3)

**Progressive alignment following the guide tree**

- Scoring matrix, sequence weight, gap penalties, etc.
A sequence (S3) is aligned against the alignment (S1, S2)

Profile alignment: alignment vs. sequence, alignment vs. alignment

Multiple alignment: ClustalW

Progressive alignment following the guide tree

S1 peeksav
S2 geekaav
S3 egewglv

From Gonnet scoring matrix

S1 vs. S3 = S(P, E)
S2 vs. S3 = S(G, E)

S1/S2 vs. S3 = (S(P, E) + S(G, E))/2

(gap penalty = -g)

Average from all pairwise scores

Profile alignment: alignment vs. sequence, alignment vs. alignment