Iterative homology searching using PSI-BLAST, scoring statistics and performance evaluation

**Genome Analysis (Integrative Bioinformatics & Genomics)**

**Lecture 7**

**PSI (Position Specific Iterated) BLAST**
- basic idea
  - use results from BLAST query to construct a profile matrix
  - search database with profile instead of query sequence
- iterate

---

**PSI-BLAST iteration**

During iteration, new hits can come in and hits can drop out of the hit-list (also the query sequence...)

At each iteration a new profile is made of the master-slave alignment

---

**A Profile Matrix (Position Specific Scoring Matrix – PSSM)**

This is the same as a profile without position-specific gap penalties

**PSI BLAST**
- Searching with a Profile
- aligning profile matrix to a simple sequence
  - like aligning two sequences
  - except score for aligning a character with a matrix position is given by the matrix itself
  - not a substitution matrix
PSI BLAST: Determining Profile Elements

- the value for a given element of the profile matrix is given by:

\[
\text{matrix}(i,j) = \log \left( \frac{Pr(a_i \mid \text{col} = j)}{Pr(a_i)} \right)
\]

- where the probability of seeing amino acid \( a_i \) in column \( j \) is estimated as:

\[
Pr(a_i \mid \text{col} = j) = \frac{\alpha f_j + \beta g_j}{\alpha + \beta}
\]

\( e.g. \alpha = \text{number of sequences in profile,} \beta = 1 \)

PSI-BLAST steps in words

- Query sequences are first scanned for the presence of so-called *low-complexity regions* (Wootton and Federhen, 1996 – next slide), i.e., regions with a biased composition likely to lead to spurious hits, are excluded from alignment.

- The program then initially operates on a single query sequence by performing a gapped BLAST search.

- Then, the program takes significant local alignments (hits) found, constructs a multiple alignment (master-slave alignment) and abstracts a position-specific scoring matrix (PSSM) from this alignment.

- Rescan the database in a subsequent round, using the PSSM, to find more homologous sequences. Iteration continues until user decides to stop or search has converged.
Low-complexity sequences

• For example: AAAAA… or AYLAYLAYL… or AYLAYLYALY…
• Low-complexity (sub)sequences have a biased composition and contain less information than high-complexity sequences
• Because of the low information content, they often lead to spurious hits without a biological basis (for example, you can’t tell whether a poly-A sequence is more similar to a globin, an immunoglobulin or a kinase sequence).

The innovation and power of BLAST is the statistical scoring system

• (PSI-)BLAST converts raw alignment scores based on the
  – (query-database) sequence lengths
  – the size of the data base
  – the (amino acid or nucleotide) composition of the database
• It also checks to what extend a hit score is higher than randomly expected
  – BLAST has a clever and fast way for doing this
• This makes the scores really comparable, so that the hit list can be ordered based on their statistical scores (bit-scores and E-values)

Statistical scoring using scrambled sequences (Z-scores)

• Biological sequence AVTCAAG can be randomly scrambled into VCAGA TA (keeping the residue composition intact)
• For assessing the statistical significance of a given pairwise alignment (query against database sequence) the Z-score can be used:
  \[ Z = \frac{x - \text{mean}}{\text{standard deviation}} \]
  where \( x \) is the (raw) alignment score between the query and database sequence, and the mean and standard-deviation are calculated over a large number (100-1000) of pairwise alignments of scrambled sequence pairs
• Z-scores are calculated independently, each time only the query and a database sequence are needed
• Z-score calculation becomes computationally prohibitive with large sequence databases

Statistics and thresholds

• Simple idea: accept only hits above a certain threshold value \( T \)
• The likelihood of random sequences to yield a score \( > T \) increases linearly with the logarithm of the ‘search space’ \( n^m \) (query sequence length \( n \) and total database sequence length \( m \))
• This gives the following formula for accepting hits:
  \[ S > T + \log(m^*n)/\lambda \]
  where \( \lambda \) is depending upon the scoring scheme (substitution matrix, gap penalties)

Alignment Bit Score

\[ B = (\lambda S - \ln K) / \ln 2 \]
• \( S \) is the raw alignment score
• The bit score (‘bits’) \( B \) has a standard set of units
• The bit score \( B \) is calculated from the number of gaps and substitutions associated with each aligned sequence. The higher the score, the more significant the alignment
• \( \lambda \) and \( K \) are the statistical parameters of the scoring system (BLOSUM62 in Blast).
• See Altschul and Gish, 1996, for a collection of values for \( \lambda \) and \( K \) over a set of widely used scoring matrices.
• Because bit scores are normalized with respect to the scoring system, they can be used to compare alignment scores from different searches based on different scoring schemes (e.g., using different amino acid exchange matrices)

Normalised sequence similarity

The \( p \)-value is defined as the probability of seeing at least one unrelated score \( S \) greater than or equal to a given score \( x \) in a database search over \( n \) sequences.
This probability follows the Poisson distribution (Waterman and Vingron, 1994):

\[ P(x; n) = 1 - e^{-n \cdot \rho(B > x)} \]

where \( n \) is the number of sequences in the database Depending on \( x \) and \( n \) (fixed)
Normalised sequence similarity

Statistical significance

The **E-value** is defined as the expected number of non-homologous sequences with score greater than or equal to a score \( x \) in a database of \( n \) sequences:

\[
E(x, n) = n \cdot P(S \geq x)
\]

For example, if \( E \)-value = 0.01, then the expected number of random hits with score \( S \geq x \) is 0.01, which means that this \( E \)-value is expected by chance only once in 100 independent searches over a randomised database.

If the \( E \)-value of a hit is 5, then five fortuitous hits with \( S \geq x \) are expected within a single randomised database search, which renders the hit not significant.

A model for database searching

**score probabilities**

- Scores resulting from searching with a query sequence against a database follow the Extreme Value Distribution (EDV) (Gumbel, 1955).
- Using the EDV, the raw alignment scores are converted to a statistical score (E value) that keeps track of the database amino acid composition and the scoring scheme (a.a. exchange matrix).

**Extreme Value Distribution (EVD)**

You know that an optimal alignment of two sequences is selected out of many suboptimal alignments, and that a database search is also about selecting the best alignment(s). This bodes well with the EVD which has a right tail that falls off more slowly than the left tail. Compared to using the normal distribution, when using the EVD an alignment has to score further beyond the expected mean value to become a significant hit.

**Extreme Value Distribution**

The probability of a score \( S \) to be larger than a given value \( x \) can be approximated following the EVD as:

\[
P(S \geq x) = 1 - \exp(-e^{-Kmne^{-x}})
\]

where \( \mu = (\ln Kmne)/\lambda \), and \( K \) a constant that can be estimated from the background amino acid distribution and scoring matrix (see Altschul and Gish, 1996, for a collection of values for \( \lambda \) and \( K \) over a set of widely used scoring matrices).

**Extreme Value Distribution**

Using the equation for \( \mu \) (preceding slide), the probability for the raw alignment score \( S \) becomes

\[
P(S \geq x) = 1 - \exp(-Kmne^{-x}).
\]

In practice, the probability \( P(S \geq x) \) is estimated using the approximation \( 1 - \exp(-e^{-x}) = e^{-x} \), which is valid for large values of \( x \). This leads to a simplification of the equation for \( P(S \geq x) \):

\[
P(S \geq x) = e^{-Kmne^{-x}} = Kmne^{-x}.
\]

The lower the probability (\( E \)-value) for a given threshold value \( x \), the more significant the score \( S \).
Normalised sequence similarity
Statistical significance

- Database searching is commonly performed using an E-value threshold in between 0.1 and 0.001.
- Low E-value thresholds decrease the number of false positives in a database search, but increase the number of false negatives, thereby lowering the sensitivity of the search.
- Each time the database grows, the BLAST team has to recalculate the $\mu = (\ln K_{mn})/\lambda$ and $\lambda$ values (one time per DB update)

Words of Encouragement

- “There are three kinds of lies: lies, damned lies, and statistics” – Benjamin Disraeli
- “Statistics in the hands of an engineer are like a lamppost to a drunk – they’re used more for support than illumination”
- “Then there is the man who drowned crossing a stream with an average depth of six inches.” – W.I.E. Gates
Making things even faster - indexing the complete database (or genome sequence)

- SSAHA – Sequence Search and Alignment by Hashing Algorithms (Ning et al., 2001)
- BLAT – BLAST-like Alignment Tool (Kent, 2002)
- PatternHunter (Ma et al., 2002)
- BLASTZ – alignment of genomic sequences (Schwartz et al., 2003)

BLAT – BLAST-Like Alignment Tool

- Blat produces two major classes of alignments:
  - at the DNA level between two sequences that are of 95% or greater identity, but which may include large inserts
  - at the protein or translated DNA level between sequences that are of 80% or greater identity and may also include large inserts.
  - The output of BLAT is flexible. By default it is a simple tab-delimited file which describes the alignment, but which does not include the sequence of the alignment itself.

Indexing (hashing) the database

- BLAT - The Blast-Like Alignment Tool
- For large-scale genome comparison
  - query can be as large as a complete genome

Preprocessing phase:
- BLAST: indexes only the query sequence
- BLAT: indexes the complete database
Hashing – associative arrays (recap)
• Indexing with the object, the
• Hash function:
  ![Hash function diagram]
• Objects should be “well spread”

Hashing – widely used implementation
• T9 Predictive Text in mobile phones
  – “hello” in Multitap:
    4, 4, 3, 3, 5, 5, 5,
    (pause) 5, 5, 6, 6, 6
  – “hello” in T9:
    4, 3, 5, 5, 6
  – Collisions: 4, 6:
    “in”, “go”

BLAT step 1- indexing the database: Find ”exact” matches with hashing
• Preprocess the database
  – Hash the database with k-words
  – For each k-word store in which sequences it appears

BLAT step 1- indexing the database: Find “exact” matches with hashing
• The database is preprocessed only once! (independent from the query)
• In a constant time we can get the sequences with a certain k-word

BLAT – step2: scanning the DB
• Hit criteria
• In a constant time we can get the
• Sequences with a certain k-word
• Relaxing hit definition -> improve sensitivity
  - allow imperfect hits
  - costly, huge hash grows a few times!
  ➔ shorten k (would lead to FP), but expect two hits (see BLAST two-hit method)

BLAT, Step 3 – Identifying homologous regions
• Exclude common k-words
• For all k-words from query
  – find out the position in db
• For results (qpos, dbpos):
  – split into buckets (64kbp)
  – sort on the diagonal (diag=qpos|dbpos)
BLAT, Step 3 – Identifying homologous regions (Continued)

- from diagonally close hits (gap limit) create "pre-clusters"
  - sort each "pre-cluster" on dbpos
  - create clusters from close hits
  - run Local Alignment for each cluster

Seeds – improving sensitivity

- More general form of k-word is a seed
- The seed CT.GT.AT.
gives "hits" with both sequences
  ...CTCGTTATA...
  ...CTAGTAAATG...

HMM-based homology searching

- Most widely used HMM-based profile searching tools currently are SAM-T2K (Karplus et al., 2000) and HMMER2 (Eddy, 1998)
- Formal probabilistic basis and consistent theory behind gap and insertion scores
- HMMs good for profile searches, not as good for alignment
- HMMs are slow

The Framework

Databank searching can be split into three phases:

1. Matching phase
   The query sequence is compared by (partial) alignment with the databank sequence. Most programs pre-select potential hits at this level. The comparison is usually heuristic and fast (see previous lecture).

2. Scoring phase
   If the database sequence passed the matching phase, the query-hit sequence pair is re-aligned and scored, mostly using a pairwise scoring matrix. (see previous lecture)

3. Selection phase
   Based on a statistical criterion, sequences with a score above a user-defined score cutoff are returned as hits. The hit list is the result of the databank searching and each hit is a potential homologue.

Some Tricky Problems

- Repeats
- Multi-domain proteins
- Low-complexity regions
- Redundancy
- Very short quotes
- Very distant sequences
- Un-annotated sequences ("conserved hypothetical" proteins)
- Profile wander

Multi-domain Proteins

It can be disadvantageous to search with multi-domain proteins. If a multi-domain protein contains domains A and B, and B is a common domain, many other (multi-domain) proteins containing this domain will be matched. The interesting domain could be A, but the majority of reported hit matches B.

In iterative sequence searching, the multi-domain proteins BC that were detected in the first round will produce hits to other proteins containing CD. The search may drift from AB to CD.

A

B

C

D

E

9
Multi-domain Proteins (cont.)

• A common conserved protein domain such as the tyrosine kinase domain can make weak but relevant matches to other domain types appear very low in the hit list, so that they are missed (e.g. only appearing after 5000 kinase hits).
• Sequences containing low-complexity regions, such as coiled coils and transmembrane regions, can cause an explosion of the search rather than convergence because of the absence of any strong sequence signals.
• Conversely, some searches may lead to premature convergence; this occurs when the PSSM is too strict only allowing matches to very similar proteins, i.e., sequences with the same domain organization as the query are detected but no homologues with different domain combinations. In this case the power of iteration is not used fully.

How to assess homology search methods

• We need an annotated database, so we know which sequences belong to what homologous families.
• Examples of databases of homologous families are PFAM, Homstrad or Astral.
• The idea is to take a protein sequence from a given homologous family, then run the search method, and then assess how well the method has carried out the search.
• This should be repeated for many query sequences and then the overall performance can be measured.

Sequence searching

How to assess homology search methods

• We need an annotated database, so we know which sequences belong to what homologous families.
• Examples of databases of homologous families are PFAM, Homstrad or Astral.
• The idea is to take a protein sequence from a given homologous family, then run the search method, and then assess how well the method has carried out the search.
• This should be repeated for many query sequences and then the overall performance can be measured.
Receiver Operator Curve (ROC)

- Plot Sensitivity (TP/(TP+FN)) against 1-specificity (1-TN/(FP+TN)), where the latter is called error

Sensitivity is also called Coverage

Going down the prediction list: correct prediction - one step up; false prediction - one step to the right
Sequence identity scoring zones

- >25-30%: homology zone
- 15-25%: twilight zone
- <15%: midnight zone (Rost, 1999)

Is midnight zone properly definable?