PAM and BLAST

Ulf Leser
This Lecture

- **Substitution Matrices**
  - PAM distance
  - PAM matrices

- **Scaling up Local Alignments**
  - BLAST
Substitution Matrices

• Recall
  – A scoring function is a function $s : \Sigma \times \Sigma' \rightarrow \text{Integer}$
    • We also call $s$ a substitution matrix
  – Direct similarity
    $$\text{sim}'(A, B) = \sum_{i=1}^{n} s(A[i], B[i])$$

• DNA: symmetric, simple matrices

• Protein sequences are different
  – Very heterogeneous properties
  – Very different impact on folding
  – Substitutions may change the 3D structure completely or not at all
Amino Acids

**Small**

- Glycine (Gly, G)  
  MW: 75.07
- Alanine (Ala, A)  
  MW: 91.09
- Serine (Ser, S)  
  MW: 117.11, pKₐ = 16
- Threonine (Thr, T)  
  MW: 117.16
- Cysteine (Cys, C)  
  MW: 121.15, pKₐ = 8.35

**Hydrophobic**

- Valine (Val, V)  
  MW: 99.14
- Leucine (Leu, L)  
  MW: 113.16
- Isoleucine (Ile, I)  
  MW: 113.16
- Methionine (Met, M)  
  MW: 131.19
- Proline (Pro, P)  
  MW: 97.12

**Aromatic**

- Phenylalanine (Phe, F)  
  MW: 147.19
- Tyrosine (Tyr, Y)  
  MW: 163.18
- Tryptophan (Trp, W)  
  MW: 186.21

**Acidic**

- Aspartic Acid (Asp, D)  
  MW: 115.08, pKₐ = 3.9
- Glutamic Acid (Glu, E)  
  MW: 129.12, pKₐ = 4.67

**Amide**

- Asparagine (Asn, N)  
  MW: 134.11
- Glutamine (Gln, Q)  
  MW: 128.14

**Basic**

- Histidine (His, H)  
  MW: 137.14, pKₐ = 6.04
- Lysine (Lys, K)  
  MW: 128.17, pKₐ = 10.79
- Arginine (Arg, R)  
  MW: 156.19, pKₐ = 12.48
Example

Where do all these numbers come from?
Is it Really Necessary?

- We count how often a particular AA was replaced by any other AA
  - Using “sure” sequence alignments
- Replacement rate of Alanin (A) := 100%
- Obviously no equal distribution
- Even if we assume that mutations happen more or the less at the same rate, they obviously don’t survive with the same rate
  - Mutations are suppressed to different degrees
  - W (Tryptophan): Strong suppression
  - S (Serin): Little suppression (less than A)
Filling a Substitution Matrix

• We need **app. 200 values**
  – Assuming a symmetric matrix

• **Possibility 1: Analytical**
  – Capture weight, polarity, size, ...
  – Find a scoring scheme to measure the difference between two AA
  – Needs to produce a single value per AA pair
  – Not used in practice

• **Possibility 2: Empirical**
  – Count which substitutions survived at which frequency **in reality**
  – Needs true alignments: Pairs of homologues and aligned sequences
Margaret O. Dayhoff

- Goal: “Deduce evolutionary relationships of the biological kingdoms, phyla, and other taxa from sequence evidence”
- Collection of all known protein sequences
  - First edition: 65 proteins
  - Several releases followed
  - Resulted in the Protein Information Resource (PIR)

Thanks to Antje Krause
PAM: Point-Accepted Mutations


- PAM has two meanings
  - 1 PAM – **Unit** for measuring the similarity of two AA sequences
  - PAM-X matrix – **Substitution matrix** to use when aligning two sequences that are X PAM distant
PAM as Distance Measure

• Definition
Let $S_1$, $S_2$ be two protein sequences with $|S_1| = |S_2|$. We say $S_1$ and $S_2$ are $x$ PAM distant, iff $S_1$ most probably was produced from $S_2$ with $x$ mutations per 100 AAs.

• Remarks
  – PAM is motivated by evolution
  – Assumptions: Mutations happen with the same rate at every position of a sequence
  – If mutation rate is high, mutations will occur again and again at the same position
  – PAM ≠ %-sequence-identity
PAM as Distance Measure

- No INDELS, only replacements
- The PAM distance $d$ of two sequences can be derived analytically from their %-sequence-identity $p$
  - $d = -\frac{3}{4} \ln(1-\frac{4}{3}p)$ (Jukes-Cantor model)
  - Derivation skipped
- Pairs with PAM >250 are probably not homologues
  - %-sequence-identity < 20%
  - Twilight zone
  - Which %-sequence-identity will two random protein sequences have?
**PAM Matrices**

- The **PAM-X matrix** contains measures for the probability that a given AA was replaced by another given AA in two sequences that are \( x \) PAM distant

- Estimated from data
  - Let \((S_{1,1}, S_{2,1}), \ldots, (S_{1,n}, S_{2,n})\) be \( n \) \(x\)-PAM distant pairs of aligned sequences
  - Compute \( f(i) \), the relative frequency of AA \( A_i \) in all pairs
  - Compute \( f(i,j) \), the relative substitution frequency of \( A_i \) and \( A_j \)
    - Number of positions \( k \) in any of the aligned pairs with \( S_{1,z}[k]=A_i \) and \( S_{2,z}[k]=A_j \) or vice versa
  - Then
    \[
    M_x(i, j) = \log \left( \frac{f(i, j)}{f(i) \times f(j)} \right)
    \]
Some Explanations

- **Log-likelihood ratio combining**
  - **Expectation**: chances to generate this mutation by chance given the relative frequencies of the two involved AAs
  - **Observation**: observed frequency of this mutation

\[
M_x(i, j) = \log \left( \frac{f(i, j)}{f(i) \cdot f(j)} \right)
\]

- **Meaning**
  - \(M(i,j) = 0\): **No selection**
  - \(M(i,j) < 0\): **Negative selection**, suppression of mutation
  - \(M(i,j) > 0\): **Positive selection**, mutation is favored
**Example**

Relative frequencies

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4/19</td>
<td>1/19</td>
<td>1/19</td>
<td>0/19</td>
</tr>
<tr>
<td>C</td>
<td>2/19</td>
<td>1/19</td>
<td>0/19</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>4/19</td>
<td>1/19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>5/19</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Mutation rates**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.48</td>
<td>0.10</td>
<td>-0.16</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>0.63</td>
<td>0.06</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>G</td>
<td>0.40</td>
<td>-0.20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td>0.50</td>
</tr>
</tbody>
</table>
Problems

• Depends on predefined alignments
• We need a substitution matrix to find optimal alignments
  – A hen-egg problem
  – Alternative: Do it manually using experience, 3D-structure, ..
• Makes several assumptions
  – Mutations are equally likely at every position in a sequence
  – Mutations are equally likely independent from AA neighbors
Real Substitution Matrices

- PAM requires **large n for each x** to adequately capture rare mutations
- **Cure:** Molecular clock assumption
  - Assume that mutations appear with equal rate over time
  - Then the frequencies of PAM-x mutations depend linearly on the frequencies of PAM-1 mutations
  - PAM-x matrices are computed by repeated matrix multiplication of PAM-1 with itself
- **Complete procedure**
  - Choose set of n pairs with small distance and align manually
  - Use these alignments to compute $M_1$
  - Compute $M_x = (M_1)^x$
BLOSUM

- PAM is a bit old-fashioned
- **BLOSUM**: **BLOcks SUbstitution Matrix**
  - Henikoff and Henikoff, 1993
  - Removes assumption of equal mutation rates across each sequence position by considering *conserved blocks*
  - Direct estimation for different PAM distances instead of error-propagating self multiplication
This Lecture

• Substitution Matrices
  – PAM distance
  – PAM matrices

• Scaling up Local Alignments
  – BLAST
Growth of EMBL
Scaling Up Local Alignment

• Searching similar sequences (with a high local alignment score) is a fundamental operation in Bioinformatics
• Sequence databases grow exponentially
• Naïve method does not scale at all
• We need faster algorithms, even if they sometimes fail
Similarity Search Problems and their Accuracy

- **Task:** Given a sequence $s$ and a database $D$, find all sequences $T$ in $D$ that are sufficiently similar to $s$
  - Often, exactly computing $T$ is not feasible and not necessary (think of the WWW and search engines)
- **Assume a method that finds a set $X$ of answers for $s$**
- **How good is this method?**
  - Some sequences will be in $X$ and $T$ – true positives
  - Some will be in $X$ but not $T$ – false positives
  - Some will be in $T$ but not $X$ – false negatives
  - Some will be neither in $X$ nor $T$ – true negatives

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Reality</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>TruePositive (TP)</td>
</tr>
<tr>
<td>-</td>
<td>FalseNegative (FN)</td>
</tr>
</tbody>
</table>
Precision and Recall

• **Precision** = \( \frac{TP}{TP + FP} \)
  - What is the fraction of correct answers in \( X \)?
  - Related to specificity

• **Recall** = \( \frac{TP}{TP + FN} \)
  - Which fraction of correct answers from \( T \) are also in \( X \)?
  - Also called sensitivity

• **Trade-Offs**
  - Usual methods compute a score per element of \( D \)
  - All sequences with a score above a threshold \( t \) are returned as \( X \)
  - Increasing \( t \): higher precision, lower recall
  - Lowering \( t \): lower precision, higher recall
  - ... if the score correlates with correctness ...

\[ \begin{array}{|c|c|c|}
\hline
\text{Reality} & \text{Prediction} \\
\hline
+ & + & \text{TruePositive (TP)} & \text{FalsePositive (FP)} \\
- & - & \text{FalseNegative (FN)} & \text{TrueNegative (TN)} \\
\hline
\end{array} \]
Example

- Let $|DB| = 1000$, $|X| = 15$, $|T| = 20$, $|X \cap T| = 9$

<table>
<thead>
<tr>
<th></th>
<th>Real: Positive</th>
<th>Real: Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alg: Positive</td>
<td>TP = 9</td>
<td>FP = 6</td>
</tr>
<tr>
<td>Alg: Negative</td>
<td>FN = 11</td>
<td>TN = 974</td>
</tr>
</tbody>
</table>

- Precision $= \frac{TP}{TP+FP} = \frac{9}{15} = 60\%$
- Recall $= \frac{TP}{TP+FN} = \frac{9}{20} = 45\%$

- Assume we increase t: $|X| = 10$, $|X \cap T| = 7$

<table>
<thead>
<tr>
<th></th>
<th>Real: Positive</th>
<th>Real: Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alg: Positive</td>
<td>TP = 7</td>
<td>FP = 3</td>
</tr>
<tr>
<td>Alg: Negative</td>
<td>FN = 13</td>
<td></td>
</tr>
</tbody>
</table>

- Precision: 70%, recall = 35%
BLAST

- Altschul, Gish, Miller, Myers, Lipman: „Basic Local Alignment Search Tool“, J Mol Bio, 1990
  - A heuristic algorithm for sequence similarity search
  - Very fast, high recall, not perfect
  - Very successful: You “blast” a sequence
  - NCBI runs thousands of BLAST searches every day

- A family of tools
  - Gapped-BLAST, PSI-BLAST, MegaBlast, BLAST-ALL, PATHBLAST, Name-BLAST, ...
  - BLAST for DNA, protein, DNA-protein, protein-DNA, ...
  - We only look at the simple DNA-DNA version
  - We skip several heuristic and domain-specific tricks
Fundamental Idea

- Fundamental idea: If two sequences have a good local alignment, then the match-area contains, with very high probability, a sub-area where the match is even better (or even exact)
- These sub-areas are called seeds

```
TTGACTCGATTATAGTGC CGGATATACTATCG
CCTATCAACAAGAATATA GTCCCTGATCCAGC

TTGACTC GAT TATAGTGC GGAT ATACTATCG
CCTATCAACA GAATATAGTCCCTGAT CCAGC

TTGACTC GAT TATAGTGC GGAT ATACTATCG
CCTATCAACA GAATATAGTCCCTGAT CCAGC
```
Algorithm

- Given query sequence \( s \) and sequence database \( D = \{d_i\} \)
- 1. Compute all substrings \( s_i \) of \( s \) of length \( q \)
  - Also called q-grams
  - How many?
- 2. Find all approximate occurrences of all \( s_i \) in all \( d_j \)
  - Gap-free alignment with matrix; score must be above threshold \( t \)
  - Hits are called seeds - approx. occurrences of some \( s_i \) in some \( d_j \)
- 3. Extend seeds to left and right in \( s_i \) and \( d_j \) until
  - [Constantly update the similarity score]
  - ... the score drops sharply
  - ... \( s_i \) or \( s_j \) ends
  - ... the score gets too bad compared to other hits found earlier
Example

q=5, t= 3, Matrix: M=+1, R=-1
s=ACGTGATA
d=GATTGACGTGACTGCTAGTGATACTATAT

\[ s_1 = \text{ACGTG}, \quad s_2 = \text{CGTGA}, \quad s_3 = \text{GTGAT}, \quad s_4 = \text{TGATA} \]

\[
\begin{align*}
\text{GATTGA} & \text{CGTGA} \\
\text{GATTGA} & \text{CGTGA} \\
\text{GATTGA} & \text{GTGACGTGACT} \\
\text{GATTGA} & \text{GTGACGTGACTG} \\
\text{GATTGA} & \text{GTGACGTGACTGCAAG} \\
\text{GATTGA} & \text{GTGACGTGACTGCAAGA} \\
\text{GATTGA} & \text{GTGACGTGACTGCAAGA} \\
\text{GATTGA} & \text{GTGACGTGACTGCAAGA} \\
\end{align*}
\]

\[
\begin{align*}
\text{ACGTGATA} & : 5 \\
\text{ACGTGATA} & : 5+1=6 \\
\text{ACGTGATA} & : 6-1=5 \\
\ldots & : \ldots \\
\end{align*}
\]
Visualization
Properties

- Finding **seeds efficiently** requires more work
  - Pre-compute all q-grams of all \( d_i \)
  - Group by q-gram
  - Called a **hash-index** (should be kept in main memory)
  - Lookup: Given \( s \), find all matching q-grams (as seeds)

- **Exclusion method**
  - Vast majority of all sequences in DB are never looked at because they do not contain a seed
  - The “seed” idea is the basis of nearly all fast alignment methods

- **Where it fails**
  - **Sensitive to \( t \):** Too high – missing hits; too low – slow
  - Does not consider gaps
Speed – Precision - Recall

- Increasing $t$
  - Higher requirements for any seed
  - Less seeds, less extensions
  - **Lower recall, higher speed**, precision stays

- Increasing $q$ (and adapting $t$)
  - Higher requirements for any seed
  - Less seeds, less extensions
  - Lower recall, higher speed, precision stays
BLAST Screenshots
BLAST-2


- **Faster**
  - BLAST: 90% of time spend in extensions
  - BLAST2: *Two seeds* in short distance
    - Needs a decrease in t

- **Higher recall**
  - BLAST didn’t even consider gaps in the extension phase
  - BLAST2: *Full local alignment* starting from seeds
    - Allows an increase of t
Further Reading

- Substitution matrixes: Krane & Raymer, Chapter 3
- BLAST, BLAST2: Merkl & Waack, Chapter 12