Markov Models of sequences

- Useful for locating repetitive structures in sequences: CpG islands, exons, introns, long repetitive sequences, etc.

- Model setup
- Parameter estimation

$n$-th order Markov chain:

\[ P(x_i = a) \text{ depends on what bases are at } x_{i-j}, j = 1 \ldots n, \text{ but not on any } x_{i-j}, j > n \]

\[ P(x_i = a \mid x_{i-1} = a_1, \ldots, x_{i-n} = a_n, x_{i-n-1} = a_{n-1}) = P(x_i = a \mid x_{i-1} = a_1, \ldots, x_{i-n} = a_n) \]

1-st order markov chain

\[ P(x_i = b \mid x_{i-1} = a) = p_{ab} \]

Probability of an $N$ letter sequence $x$

\[ P(x) = P(x_1) \prod_{i=2}^{N} p_{x_{i-1}x_i} \]

Modeling the beginning and end:

Extra states B and E with $p_{aB} = 0$ for all $a$ and $p_{EB} = 0$ for all $a$.

Maximum Likelihood estimate of $p_{ab}$ from observed frequencies $f_{ab}$:

\[ p_{ab} = \frac{f_{ab}}{\sum_c f_{ac}} \]

Example:

Location of CpG islands – produces two matrices $p^+_{ab}$ for CpG islands and $p^-_{ab}$ for ordinary sequence.

<table>
<thead>
<tr>
<th>+</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>$-$</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.180</td>
<td>0.274</td>
<td>0.426</td>
<td>0.120</td>
<td>A</td>
<td>0.300</td>
<td>0.205</td>
<td>0.285</td>
<td>0.210</td>
</tr>
<tr>
<td>C</td>
<td>0.171</td>
<td>0.368</td>
<td>0.274</td>
<td>0.188</td>
<td>C</td>
<td>0.322</td>
<td>0.298</td>
<td>0.078</td>
<td>0.302</td>
</tr>
<tr>
<td>G</td>
<td>0.161</td>
<td>0.339</td>
<td>0.375</td>
<td>0.125</td>
<td>G</td>
<td>0.248</td>
<td>0.246</td>
<td>0.298</td>
<td>0.208</td>
</tr>
<tr>
<td>T</td>
<td>0.079</td>
<td>0.355</td>
<td>0.384</td>
<td>0.182</td>
<td>T</td>
<td>0.177</td>
<td>0.239</td>
<td>0.292</td>
<td>0.292</td>
</tr>
</tbody>
</table>
Significance: log-odds ratio  \( S(x) = \frac{\ln(x | +)}{\ln(x | -)} = \sum_i \beta_{x_i} x_i \) where \( \beta_{ab} = \ln \left( \frac{p_{ab}}{p_{ab}} \right) \)

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-0.740</td>
<td>0.419</td>
<td>0.580</td>
<td>-0.803</td>
</tr>
<tr>
<td>C</td>
<td>-0.913</td>
<td>0.302</td>
<td>1.812</td>
<td>-0.685</td>
</tr>
<tr>
<td>G</td>
<td>-0.624</td>
<td>0.461</td>
<td>0.331</td>
<td>-0.730</td>
</tr>
<tr>
<td>T</td>
<td>-1.169</td>
<td>0.573</td>
<td>0.393</td>
<td>-0.679</td>
</tr>
</tbody>
</table>
Hidden Markov Model

- Location of CpG islands is unknown.
- Create Markov chain with two states per letter: \( A_+, G_+, C_+, T_+ \) for CpG islands and \( A_-, G_-, C_-, T_- \) for general sequence.

Put

\[
P(x_i \in \{ A_+, G_+, C_+, T_+ \} \mid x_{i-1} \in \{ A_+, G_+, C_+, T_+ \}) = q^+
\]

\[
P(x_i \in \{ A_-, G_-, C_-, T_- \} \mid x_{i-1} \in \{ A_-, G_-, C_-, T_- \}) = q^-
\]

\[
P_{ab} = \begin{cases} 
  p_{ab}q^+ & a \in \{ A_+, G_+, C_+, T_+ \}, b \in \{ A_+, G_+, C_+, T_+ \} \\
  (1 - q^+) / 4 & a \in \{ A_+, G_+, C_+, T_+ \}, b \in \{ A_-, G_-, C_-, T_- \} \\
  p_{ab}q^- & a \in \{ A_-, G_-, C_-, T_- \}, b \in \{ A_-, G_-, C_-, T_- \} \\
  (1 - q^-) / 4 & a \in \{ A_-, G_-, C_-, T_- \}, b \in \{ A_+, G_+, C_+, T_+ \} 
\end{cases}
\]

Hidden feature: No longer one-to-one correspondence between observed quantities and the states of the system.

Formal definition:

- States \( \pi_i \),
- alphabet \( x_j \),
- transition matrix \( p_{\alpha \beta} = P(\pi_i = \beta \mid \pi_{i-1} = \alpha) \)
- emmission probabilities \( e_{\alpha} (b) = P(x_i = b \mid \pi_i = \alpha) \)
- Beginning and end state labeled with 0

Joint probability of an observed sequence \( x \) and state sequence \( \pi \):

\[
P(x, \pi) = p_{0 \pi_1} \prod_{i=1}^{L} e_{\pi_i} (x_i) p_{\pi_i \pi_{i+1}}
\]

Problems:

1. Given observed sequence \( x \) find the most probable path \( \pi \) (Viterbi algorithm)
2. Calculate efficiently \( P(x) \) (forward algorithm)
3. Calculate the posterior probability \( P(\pi_i = \alpha \mid x) \) (backward algorithm)
4. Find the parameters \( p_{\alpha \beta} \) and \( e_{\alpha} (b) \) that maximize \( P(x) \). (Baum-Welch algorithm, Viterbi training)
Viterbi algorithm

- Finds most probable sequence of states \( \pi \) given the observed sequence \( x \):

\[
\pi^* = \arg \max \limits_{\pi} P(x, \pi)
\]

- Uses recursive relation for \( v_\alpha(i) = \max \limits_{\pi} P(x_1...x_i, \pi_1...\pi_{i-1}\alpha) \):

\[
v_\alpha(i + 1) = e_\alpha(x_{i+1}) \max \limits_\beta \{ v_\beta(i) p_{\beta\alpha} \}
\]

- Initial condition: \( v_\alpha(0) = 1 \), \( v_\alpha(0) = 0 \) for \( \alpha \neq 0 \)

- Most probable path is found by backtracking

Ex: Calculate the most probable path for \( \text{CGCG} \), assume \( p_{0\beta} = 0.125 \), \( q^+ = q^- = 0.9 \)

<table>
<thead>
<tr>
<th>( v_\alpha(i) )</th>
<th>0</th>
<th>C</th>
<th>G</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A_+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C_+</td>
<td>0</td>
<td>0.125</td>
<td>0</td>
<td>0.0094</td>
<td>0</td>
</tr>
<tr>
<td>G_+</td>
<td>0</td>
<td>0</td>
<td>0.031</td>
<td>0</td>
<td>0.0023</td>
</tr>
<tr>
<td>T_+</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A_</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C_</td>
<td>0</td>
<td>0.125</td>
<td>0</td>
<td>0.0019</td>
<td>0</td>
</tr>
<tr>
<td>G_</td>
<td>0</td>
<td>0</td>
<td>0.0088</td>
<td>0</td>
<td>0.00014</td>
</tr>
<tr>
<td>T_</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Forward algorithm

• Calculates efficiently $P(x) = \sum_{\pi} P(x, \pi)

• Enumeration of all paths inefficient.

• Recursive formula for $f_\alpha(i) = P(x_1...x_i, \pi_i = \alpha)$:

  $$f_\alpha(i+1) = e_\alpha(x_{i+1}) \sum_{\beta} f_\beta(i) p_{\beta\alpha}$$

• Initial condition: $f_0(0) = 1$, $f_\alpha(0) = 0$ for $\alpha \neq 0$

Backward algorithm

• Calculates the posterior probability $P(\pi_i = \alpha | x)$

• Note $P(\pi_i = \alpha | x) = \frac{P(x, \pi_i = \alpha)}{P(x)} = \frac{f_\alpha(i) b_\alpha(i)}{P(x)}$ where $b_\alpha(i) = P(x_{i+1}...x_L | \pi_i = \alpha)$ obeys recursive formula

  $$b_\alpha(i-1) = \sum_{\beta} p_{\alpha\beta} e_\beta(x_i) b_\beta(i)$$

• Initial condition: $b_\alpha(L) = p_{\alpha0}$

Ex: Location of CpG islands:

• as the most probable path

• by posterior probability

Both roughly equivalent in performance
Parameter estimation

**Known sequence of states**

- Suppose $A_{\alpha\beta}, E_{\alpha}(b)$ are the numbers of transitions and emissions in training data:

  $$p_{\alpha\beta} = \frac{A_{\alpha\beta}}{\sum_{\gamma} A_{\alpha\gamma}}, \quad e_{\alpha}(b) = \frac{E_{\alpha}(b)}{\sum_{c} E_{\alpha}(c)}$$

**Unknown sequence of states:**

**Baum-Welch iterative procedure**

1. Estimate $A_{\alpha\beta}, E_{\alpha}(b)$ by considering probable paths for the training sequences using current values of $p_{\alpha\beta}, e_{\alpha}(b)$

   $$P(\pi_i = \alpha, \pi_{i+1} = \beta \mid x) = \frac{f_{\alpha}(i)p_{\alpha\beta}e_{\beta}(x_{i+1})b_{\beta}(i+1)}{P(x)}$$

   $$A_{\alpha\beta} = \sum_{ij} P(\pi_i = \alpha, \pi_{i+1} = \beta \mid x^j)$$

   $$E_{\alpha}(b) = \sum_{j} \sum_{i \mid x_i^j = b} P(\pi_i = \alpha \mid x^j)$$

2. Recalculate $p_{\alpha\beta}, e_{\alpha}(b)$ using equations above

3. Go back to step 1.

**Viterbi training**

1. Find the most probable paths for training sequences using Viterbi algorithm and current values of $p_{\alpha\beta}, e_{\alpha}(b)$

2. Recalculate $A_{\alpha\beta}, E_{\alpha}(b)$ using those paths

3. Recalculate $p_{\alpha\beta}, e_{\alpha}(b)$ using equations above

4. Go back to step 1