1 Usage in Bioinformatics

1. gene finding: GLIMMER, GENSCAN
2. motif finding
3. segmentation analysis: chromHMM
4. find CpG islands

2 Simple example

1. Sample sequence data:

\[ \begin{array}{cccccccccccccccc}
Y & E_1E_2E_3E_1E_2E_3E_1E_2E_3 & I & I & I & E_1E_2E_3E_1E_2E_3 \\
\hline
\text{Exon} & \text{Intron} & \text{Exon} & \text{hidden states} & \text{observed symbols}
\end{array} \]

2. Problem: find exon and intron in this sequence

3. Assumption: exon and intron have different probability of seeing a nucleotide

4. Hidden states in this example: \{intron, exon\}
   more specifically: states=\{E_1, E_2, E_3, I\}, where E_1 is the first nucleotide in a codon, E_2 is the second nucleotide in a codon, E_3 is the third nucleotide in a codon, and I is a nucleotide in an intron.

5. Markov chain example (transition diagram, see Figure 1):

6. Five things we care about:
   (a) observed sequence
Figure 1: Markov chain example

(b) hidden state
(c) transition probability
(d) initial probability
(e) emission probability

7. Some notations:

X: observed symbols (ATCG in this example)
Y: hidden states (E₁,E₂,E₃,I in this example)
Θ: Set of parameters, including:
   (a) Transition probability, \{t_{ij}\}, i, j \in \{E₁, E₂, E₃, I\}
   (b) Emission probability, \{e(x_n|i)\}, n \in \{1, ..., L\}, i \in \{E₁, E₂, E₃, I\}
   (c) Initial probability, \{π_i\}, i \in \{E₁, E₂, E₃, I\}

8. Question:

(a) p(X|Θ)?
(b) What are the hidden states? \(Y^* = \arg \max_Y p(X|Θ)\)?
(c) how to estimate Θ?

Answers:

1. p(X|Θ)?

\[ p(X|Θ) = \sum_Y p(X,Y|Θ). \] However, simple enumeration is not computationally feasible.
To solve this problem, we use **forward algorithm**:

\[
\alpha(n, i) = p(x_1, x_2, \ldots, x_n, y_n = i | \Theta) \\
= \sum_{k \in \{E_1, E_2, E_3, I\}} [\alpha(n - 1, k) t(k, i) e(x_n | i)] \\
\text{start: } \alpha(1, i) = \pi(i)
\]

Finally, \( p(X | \Theta) = \sum_{i \in \{E_1, E_2, E_3, I\}} \alpha(L, i) \)

The computational complexity of this algorithm is \( O(L \cdot 4^2) \)

2. **What are the hidden states?** \( Y^* = \arg\max_Y p(X, Y | \Theta) \)?

Here we use **Viterbi algorithm** - a dynamic programming algorithm for finding the most likely sequence of hidden states.

\[
\Gamma(n, i) = \max_{y_1, \ldots, y_{n-1}} P(X_1, \ldots, X_n, y_1, \ldots, y_{n-1}, y_n = i | \Theta)
\]

Recursively,

\[
\Gamma(n, i) = \max_k [\Gamma(n - 1, k) t(k, i) e(X_n | i)] \Rightarrow \max_k \Gamma(L, k) = \max_y P(X, y | \Theta)
\]

Traceback:

\[
y_L^* = \arg\max_k \Gamma(L, k), \ y_{L-1}^* = \arg\max_k \Gamma(L - 1, k), \ldots
\]

computation time \( O(L \cdot 4^2) \)

What if we are more interested in \( \hat{y}_n = \arg\max_i P(y_n = i | X, \Theta) \)?

\[
P(y_n = i | X, \Theta) = \frac{P(X_1, \ldots, X_L, y_n = i | \Theta)}{P(X_1, \ldots, X_L | \Theta)} = \frac{P(X_1, \ldots, X_n, y_n = i | \Theta) P(X_{n+1}, \ldots, X_L | y_n = i, \Theta)}{P(X | \Theta)}
\]

Last time we defined \( \alpha(n, i) = P(X_1, \ldots, X_n, y_n = i | \Theta) \)

Now, \( \beta(n, i) \triangleq P(X_{n+1}, \ldots, X_L | y_n = i, \Theta) \)

**Backward algorithm**:

\[
\beta(n, i) = \sum_k [\beta(n + 1, k) e(X_{n+1} | k) t(i, k)] \\
= P(X_{n+2}, \ldots, X_L | y_{n+1} = k, \Theta) P(X_{n+1} | y_{n+1} = k, \Theta) P(y_{n+1} = k | y_n = i, \Theta) \\
= \sum_k P(X_{n+1}, \ldots, X_L, y_{n+1} = k | y_n = i, \Theta)
\]

What is \( \beta(L - 1, i) \)?

\[
\beta(L - 1, i) = P(X_L | y_{L-1} = i, \Theta) = \sum_{\gamma \in \{A, T, C, G\}} P(X_{L-1} = \gamma, X_L | y_{L-1} = i, \Theta) \\
= \sum_{k \in \{E_1, E_2, E_3, E_4\}} \sum_{\gamma} e(X_{L-1} = \gamma | i) \cdot t(i, k) \cdot e(X_L | k)
\]
Given $\alpha(n, i)$ and $\beta(n, i)$, we have

$$P(y_n = i | X, \Theta) = \frac{\alpha(n, i) \beta(n, i)}{\sum_k \alpha(n, k) \beta(n, k)} \Rightarrow \hat{y}_n = \text{argmax}_i \alpha(n, i) \beta(n, i)$$

This serves as a second way of finding hidden states (as opposed to Viterbi).

3. Estimate $\Theta$ - Training

We use **Baum-Welch algorithm**, which is similar to EM algorithm.

From the forward-backward algorithm: $P(y_n = i | X, \Theta^{(m)}) \Rightarrow \hat{y}^{(m)}_n$

E-step, m-th iteration: $\hat{y}^{(m)}_n = E[y_n | X, \Theta^{(m)}]$

M-step, $\Theta^{(m+1)} = \text{argmax}_\Theta P(X, \hat{y}^{(m)}_n | \Theta)$