10.1 Introduction

Last lecture, we introduced the global alignments of multiple sequences. In this lecture, we develop a statistical model for partial global alignments of two sequences and discuss how to find optimal alignments.

10.2 Statistical models for alignments

Recall from last time that for a partial global alignment (p.g.a.), we use the notation \((\mathcal{H}, \leq_C(\mathcal{H}))\), where \(\mathcal{H}\) is a laminar family and \(\leq_C(\mathcal{H})\) is a partial order on the components of \(\mathcal{H}\) satisfying certain nice properties.

Example. For the sequences \(\sigma_1 = GGACT\) and \(\sigma_2 = TAGAC\), consider the following partial global alignment:

\[
\begin{array}{c}
\sigma_1\
\sigma_2
\end{array}
\begin{array}{c}
GGACT\
TAGAC
\end{array}
\]

Here each of the six columns is a component of \(\mathcal{H}\) and \(\leq_C(\mathcal{H})\) is the total ordering on the columns moving from left to right. Note that for two sequences, each component of the laminar family is of size one (like column 1) or size two (like column 2).

Recall that for two sequences of length \(n\) and \(m\), there are \(\binom{n+m}{n}\) partial global alignments of these sequences. To create a statistical model, we want to assign a probability to each p.g.a. To do so, we will use an extension of the Jukes-Cantor model, which was introduced in an earlier lecture.

10.2.1 Extension of the Jukes-Cantor model for two sequences

First we will work with individual components. For components of size two, the Jukes-Cantor model has only one parameter: \(\mu\), the probability of the two aligned characters matching. Then for example, given two aligned characters \(\sigma_a^1\) and \(\sigma_b^2\), the probability that \(\sigma_a^1 = \sigma_b^2 = A\) is \(\frac{\mu}{4}\).

Since the probability that \(\sigma_a^1 \neq \sigma_b^2\) must be \(1 - \mu\) and there are 12 ways this could happen (16 total possibilities minus the four matching ones), for any \(i \neq j \in \{A, C, G, T\}\),

\[
P(\sigma_a^1 = i, \sigma_b^2 = j) = \frac{1-\mu}{12}.
\]
For components of size one (i.e. a character aligned with a dash), we simply assign a probability of $\frac{1}{4}$ to each possible character ($A, C, G, T$).

Now we need to extend this model on the components to a probability distribution on all partial global alignments.

### 10.3 The Hidden Markov Model

To give probability distributions on partial global alignments, we will use random walks to construct p.g.a.’s.

#### 10.3.1 The states

Our random walk will have 6 states: “begin”, $S$ (for “Silent”), $H$, $\begin{bmatrix} 1 \end{bmatrix}$, $\begin{bmatrix} 2 \end{bmatrix}$, and “end”. Anytime the walk is in states $H$, $\begin{bmatrix} 1 \end{bmatrix}$, or $\begin{bmatrix} 2 \end{bmatrix}$, we will create a component of our global alignment. State $H$ gives us a component of two characters, state $\begin{bmatrix} 1 \end{bmatrix}$ gives a component of size one whose character is in $\sigma^1$, and likewise state $\begin{bmatrix} 2 \end{bmatrix}$ gives a component of size one whose character is in $\sigma^2$.

#### 10.3.2 Transition probabilities

This walk will always start in “begin”. From there it will move with probability 1 to state $S$, the silent state. From state $S$, the random walk will travel to “end” with probability $\tau$ and will be equally likely to travel to $H$, $\begin{bmatrix} 1 \end{bmatrix}$, or $\begin{bmatrix} 2 \end{bmatrix}$ (so it will travel to each of these with probability $\frac{1-\tau}{3}$). From $H$, $\begin{bmatrix} 1 \end{bmatrix}$, or $\begin{bmatrix} 2 \end{bmatrix}$, the walk travels back to $S$ with probability 1. When the walk reaches “end”, it ends. We call this the “Transition diagram”; see Figure 10.1.
Whenever the random walk is in state $H$, $[1]$, or $[2]$, it will output a component for our global alignment. From state $H$, it will output a specific matching character state (such as $\sigma_i^1 = \sigma_j^2 = A$) with probability $\frac{\mu_4}{4}$ and output a specific non-matching pair (such as $\sigma_i^1 = A$, $\sigma_j^2 = C$) with probability $\frac{1 - \mu_{12}}{12}$. From states $[1]$ or $[2]$, the model will output each size one component (e.g. $\sigma_i^1 = A$ from state $[1]$) with probability $\frac{1}{4}$.

**Remark.** This process produces a global alignment because the components are given out in order as the random walk proceeds. Also, by construction the sequences made here can be arbitrarily long. So rather than just considering sequences of a fixed length, this gives us a probability distribution of all global alignments of a priori unbounded length.

### 10.3.3 Computing the likelihood of a given alignment

What is the likelihood of an alignment $A$? For every matching component, the random walk must have traveled into $H$ and then into a matching state, which occurs with probability $\frac{1 - \tau}{3} \cdot \frac{\mu_4}{4}$ (note that these two events are independent). For every mismatching component of size two, the walk needed to travel into $H$ and then into a mismatching state, which happens with probability $\frac{1 - \tau}{3} \cdot \frac{1 - \mu_{12}}{12}$. For every component of size 1, our random walk travelled into states $[1]$ or $[2]$, then into a specific character, giving a probability $\frac{1 - \tau}{3} \cdot \frac{1}{4})$. Finally, for the walk to end, it must have travelled to “end”, with probability $\tau$. Putting this all together gives us the likelihood of $A$:

$$
\mathcal{L}(A) = \left( \frac{1 - \tau}{3} \right)^{x + m + g} \left( \frac{\mu_4}{4} \right)^m \left( \frac{1 - \mu_{12}}{12} \right)^x \left( \frac{1}{4} \right)^g \tau,
$$

where

- $m = \#\{\text{times the random walk goes into } H \text{ and outputs matching characters}\}$,
- $x = \#\{\text{times the random walk goes into } H \text{ and outputs distinct characters}\}$, and
- $g = \#\{\text{times the random walk goes into states } [1] \text{ or } [2]\}$.

Note here that $m$, $x$, and $g$ are sufficient statistics for the likelihood of $A$, and these depend only on partial global alignments (the likelihood function depends only on the values of $m$, $x$, $g$ and the fixed parameters $\tau, \mu$). To reduce this computation to a linear optimization problem, we can look at $\log(\mathcal{L}(A))$:

$$
\log(\mathcal{L}(A)) = \log(\tau) + mM + xX + gG,
$$

where $M = \log(\frac{1 - \tau}{3} \cdot \frac{\mu_4}{4})$, $X = \log(\frac{1 - \tau}{3} \cdot \frac{1 - \mu_{12}}{12})$, and $G = \log(\frac{1 - \tau}{12})$. 

10-3
10.4 Finding optimal alignments

Given \((M, X, G) = \theta^*\), we have the problem of computing which alignment \(A\) optimizes \((L(A)|\theta^*)\).

**Definition 10.1.** The alignment \(\hat{A}\) attaining \(\max_A(L(A)|\theta^*)\) is called \(\arg\max_A(L(A)|\theta^*)\).\(^1\)

To find \(\arg\max_A(L(A)|\theta^*)\), we need to find \(m, x, g\) that maximize the usual inner product

\[(m, x, g) \cdot (M, X, G).
\]

A key point to remark is that \(2(m + x) + g\) must equal the total number of characters in both sequences, i.e. \(n + m\). This enables us reduce to the case of two sufficient statistics, \(g\) and \(x\). Therefore, we have a model with two free-parameters and drawings will be easy to do.

**Example.** Recall our two sequences, \(\sigma^1 = GGACT\) and \(\sigma^2 = TAGAC\). For the partial global alignment

\[
\begin{array}{c}
_ & GGACT \\
TAGAC & _
\end{array}
\]

we have two components of size one and one component of size two with distinct characters, so this p.g.a. has sufficient statistics \((g, x) = (2, 1)\). The p.g.a.

\[
\begin{array}{c}
GGACT & _ & _ & _ & _ & _ \\
_ & _ & _ & _ & TAGAC
\end{array}
\]

corresponds to \((g, x) = (10, 0)\), whereas the p.g.a.

\[
\begin{array}{c}
GGACT \\
TAGAC
\end{array}
\]

yields \((g, x) = (0, 5)\).

Taking the convex hull of all such \((g, x)\) gives us a 2-dimensional polytope \(P\). All \((g, x)\) maximizing \((L(A)|\theta^*)\) for some \(\theta^*\) must be the vertices of this polytope. In general, they will lie in faces of the polytope, since they will maximize the inner product with \(\omega = (M, X, G)\). Thus \((m, g, x) \in \text{face}_\omega(P)\).

**Remark.** Rather than use the extension of the Jukes-Cantor Model developed earlier, biologists often use the weights \(M = 1, X = -1, G = -2\). Although these values are arbitrary, they roughly correspond to computing \(\log(L(A))/L_{\text{R}}(A)\), rather than \(L(A)\), where \(L_{\text{R}}(A)\) (called “random likelihood”) is our model with \(\mu = \frac{1}{4}\) (completely random characters). For this reason, it is wise not to restrict ourselves to the set \(M, X, G < 0\) (by our original construction they correspond to log values of real numbers between zero and one).

\(^1\)In agreement with the standard Computer Science notation.
10.5 Three main questions

1. Given $\sigma_1$, $\sigma_2$, what is $P(\sigma_i \sim \sigma_j)$? In other words, what is the probability that two characters are in the same component?

2. Given $\theta^* = (X, M, G)$, what is $\arg\max_A (L(A)|\theta^*)$?

3. What is $\{\arg\max_A (L(A)|\theta^*) : \theta^* \in \mathbb{R}^3\}$? In other words, what is the set of integer points on the boundary of the polytope we constructed above?

10.5.1 Good news: one algorithm solves them all

Luckily, three different versions of the same algorithm solve all three questions. The actual algorithm will be given in the next lecture. For now we discuss its inputs. In general, the algorithm takes in objects and two binary operators on these objects, denoted by $\oplus$ and $\odot$.

To answer questions 1-3, we use three different inputs (Objects, $\oplus$, $\odot$). Namely

1. $(\mathbb{R}, +, \cdot)$,
2. $(\mathbb{R}, \min, +)$, and
3. (Polytopes, convex hull, Minkowski sum).

10.5.2 Polytope Algebra

Let $P_d$ be the set of polytopes in $\mathbb{R}^d$. Consider $P, Q \in P_d$.

**Definition 10.2.** The **convex hull** of $P$ and $Q$ is the convex hull of all points in $P \cup Q$. More precisely,

$$P \oplus Q = \{\lambda p + (1 - \lambda)q : p \in P, q \in Q, \lambda \in [0, 1]\}.$$

**Definition 10.3.** The **Minkowski sum** of $P$ and $Q$ is given by

$$P \odot Q = \{p + q : p \in P, q \in Q\}.$$

By using the $V$-representation (in the case of $\oplus$) and the $H$-representation (in the case of $\odot$) of polytopes, we see that $P \oplus Q$ and $P \odot Q$ are again polytopes.

Note that with these definitions we have the distributive law,

$$(P \oplus Q) \odot R = (P \odot R) \oplus (Q \odot R),$$

since for any $p \in P, q \in Q, r \in R$ and $0 \leq \lambda \leq 1$ we have the equality

$$\lambda p + (1 - \lambda)q + r = \lambda(p + r) + (1 - \lambda)(q + r).$$

**Remark.** For $d = 1$ the semiring of the third input (Polytopes, convex hull, Minkowski sum) is simply a collection of intervals (i.e. 1-dim polytopes), with

$$[a, b] \oplus [c, d] = [\min\{a, c\}, \max\{b, d\}], \text{ and } [a, b] \odot [c, d] = [a + c, b + d].$$