The Problem of Probabilistic Sequence Alignment

Colby Simpson

October 24, 2015

1 Preliminaries

Definition 1. A probabilistic sequence S of length l from an alphabet A is a $|\Sigma| \times l$ matrix

$$\begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & \dots & \alpha_{1,n} \\ \alpha_{2,1} & \alpha_{2,2} & \dots & \alpha_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{|A|,1} & \alpha_{|A|,2} & \dots & \alpha_{|A|,l} \end{bmatrix}$$

Where $\alpha_{i,j}$ is the probability of the i-th letter of A being the j-th letter of a sequence. This implies that

- $0 \leq \alpha_{i,j}$ for all i,j.
- For all columns j we have $\Sigma_{a \in A} P_j(a) = \alpha_{1,j} + \alpha_{2,j} + \cdots + \alpha_{|A|,n} = 1$

For the rest of the discussion we will fix $A = \{a, c, g, t\}$, but all of what is about to be said can be said for general finite alphabets .

Note that "deterministic" sequences where we know exactly what nucleotide what at what index in the sequence are a subset of probabilistic sequences:

Example 1. The deterministic sequence $S =$ atagc can be identified with the probabilistic sequence

$$\begin{array}{c} A \\ C \\ G \\ T \end{array} \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

2 Probabilistic Alignment

Given two probabilistic sequences

$$S = \begin{array}{cc} \begin{matrix} A \\ C \\ G \\ T \end{matrix} & \begin{bmatrix} \alpha_{A,1} & \alpha_{A,2} & \dots & \alpha_{A,m} \\ \alpha_{C,1} & \alpha_{C,2} & \dots & \alpha_{C,m} \\ \alpha_{G,1} & \alpha_{G,2} & \dots & \alpha_{G,m} \\ \alpha_{T,1} & \alpha_{T,2} & \dots & \alpha_{T,m} \end{bmatrix} \end{array}$$

and

$$T = \begin{array}{cc} \begin{matrix} A \\ C \\ G \\ T \end{matrix} & \begin{bmatrix} \beta_{A,1} & \beta_{A,2} & \dots & \beta_{A,n} \\ \beta_{C,1} & \beta_{C,2} & \dots & \beta_{C,n} \\ \beta_{G,1} & \beta_{G,2} & \dots & \beta_{G,n} \\ \beta_{T,1} & \beta_{T,2} & \dots & \beta_{T,n} \end{bmatrix} \end{array}$$

an alignment of S and T is almost exactly like the notion we saw in class except we are now aligning columns of the matrices S and T as opposed to nucleotides. How do we score such an alignment?

We have no information regarding what we expect the gap distribution to look like in an optimal alignment, so let's place gaps and calculate the resulting score deterministically, the same way as in a standard alignment. We'll use a linear gap penalty $cost(l) = -cl$.

2.1 Aligning a column of S to a column of T

The columns of S and T are categorical probability distributions over $\{a, c, g, t\}$. How do we calculate the score of aligning a column of S to a column of T?

- : If both columns from S and T are certain, ie contain exactly one entry of 1 and the rest 0, then our scoring method should return the same values as in the standard case: a 1 if the nucleotides match and a -1 if they don't.

It seems that what we want our score to reflect a measure of similarity between columns. Previously our columns were either identical or they weren't, so we were able to award them discrete scores. We propose the following thought experiment:

Imagine that a probabilistic sequence acts as a template for a machine which produces bona fide DNA sequences, where a nucleotide N occurs at index i in the sequence with probability $\alpha_{N,i}$. Using this machine we repeatedly use S and T to generate sequences of nucleotides, and we repeatedly observe and record the values of S_i and T_j. If column i of S and column j of T share similar distributions, then we would expect "overlap" between our observations of S_i and T_j: ie we'd expect similar observed frequencies of nucleotides placed at S_i and T_j after many trials.

Our proposal is that we score an alignment of columns based on the "similarity" of their distributions. The more similar the observed frequencies of nucleotides are in the long run, the higher a score we award a matching of two columns, with a score of 1 being awarded if the distributions are identical, and a -1 if there is no overlap in the observations of S_i and T_j.

2.2 Hellinger distance

Definition 2. Given two discrete probability distributions $p(x)$ and $q(x)$ defined on X, the *Bhattacharyya Coefficient* is given by

$$BC(p,q) = \Sigma_{x \in X} \sqrt{p(x)q(x)}$$

A geometric interpretation of the Bhattacharyya coefficient is that it gives the cosine of the angles between $\left[\sqrt{p(x_1)}, \ldots, \sqrt{p(x_n)}\right]$ and $\left[\sqrt{q(x_1)}, \ldots, \sqrt{q(x_n)}\right]$. To examine the properties of this measure, we'll first need to prove an auxiliary inequality.

Lemma 1. *Jensen's Lemma*: If $f(x) : D \to \mathbb{R}$ is a real concave function, then for any $x_1, \ldots, x_n \in D$ and for any positive weights $a_1, \ldots, a_n$ we have

$$f\left(\frac{\sum a_i x_i}{\sum a_i}\right) \geq \frac{\sum a_i f(x_i)}{\sum a_i}$$

Proof. We proceed by induction on n. Jensen's inequality holds trivially in the case n=1, so we first consider n=2. If α_1, α_2 are arbitrary non-negative real numbers such that $\alpha_1 + \alpha_2 = 1$, then by the concavity of f we have

$$f(a_1 x_1 + a_2 x_2) \geq a_1 f(x_1) + a_2 f(x_2)$$

from which, since $a_1 + a_2 = 1$ it follows that

$$f\left(\frac{a_1 x_1 + a_2 x_2}{a_1 + a_2}\right) \geq \frac{a_1 f(x_1) + a_2 f(x_2)}{a_1 + a_2}$$

Now assume that Jensen's inequality holds for n $\geq$ 2. Given $a_1, \cdots, a_{n+1}$ such that $\sum_1^{n+1} a_i = 1$ at least one of the a_i is strictly positive, say a_1. Then

$$\sum_2^{n+1} a_i = 1 - a_i \to \sum_2^{n+1} \frac{a_i}{1-a_1} = 1$$

.

We thus have

$$f\left(\sum_1^{n+1} a_i x_i\right) = f\left(a_1 x_1 + (1-a_1)\sum_2^{n+1} \frac{a_i x_i}{1-a_1}\right)$$

$$\begin{aligned}
&\geq a_1 f(x_1) + (1-a_1) f\left(\sum_2^{n+1} \frac{a_i x_i}{1-a_i}\right) \text{(concavity of} f\text{)} \\
&= a_1 f(x_1) + (1-a_1) f\left(\sum_2^{n+1} \frac{\frac{a_i x_i}{1-a_1}}{\sum_2^{n+1} \frac{a_i}{1-a_1}}\right) \left(\because \sum_2^{n+1} \frac{a_i}{1-a_1} = 1\right) \\
&\geq a_1 f(x_1) + (1-a_1) \frac{\sum_2^{n+1} \frac{a_i}{1-a_1} f(x_i)}{\sum_2^{n+1} \frac{a_i}{1-a_1}} \text{(I.H)} \\
&= \sum_1^{n+1} a_i f(x_i) \left(\because \sum_2^{n+1} \frac{a_i}{1-a_1} = 1\right) \\
&= \frac{\sum_1^{n+1} a_i f(x_i)}{\sum_1^{n+1} a_i} \left(\because \sum_1^{n+1} a_i = 1\right) \\
&\therefore
\end{aligned}$$

$$f\left(\frac{\sum_1^{n+1} a_i x_i}{\sum_1^{n+1} a_i}\right) \geq \frac{\sum_1^{n+1} a_i f(x_i)}{\sum_1^{n+1} a_i}$$

which completes the proof. □

From Jensen's inequality we see that

$$0 \leq BQ(p,q) = \sum_{x \in X} \sqrt{p(x)q(x)} = \sum_{x \in X} p(x) \sqrt{\frac{q(x)}{p(x)}} \leq \sqrt{\sum_{x \in X} q(x)} = 1$$

If $p(x)$ and $q(x)$ are identical distributions then

$$BQ(p,q) = \sum_{x \in X} \sqrt{p^2(x)} = 1$$

and if $p(x)$ and $q(x)$ are distributions with no overlap (ie $p(x)q(x) = 0$ for all $x \in X$) we have

$$BQ(p,q) = \sum_{x \in X} \sqrt{0} = 0$$

It seems reasonable to ask that our measure of similarity between distributions be a metric. The Bhattacharyya coefficient does not define a metric, since it fails to assign identical distributions a distance of 0. However, the *Hellinger Distance* function given by

$$d(p,q) = \sqrt{(1 - BC(p,q))}$$

does define a metric. We see that if p and q are identical than $d(p,q) = 0$ and if there is no overlap between p and q then $d(p,q) = 1$. The other properties are verified in (Cite Comaniciu, D., Ramesh, V. & Meer, P. (2003). Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence , 25(5) , 564577.)

Let $P_i(x)$ and $Q_j(x)$ be the probability distributions of nucleotides at index i of S and index j of T respectively. Since we have $0 \leq d(p,q) \leq 1$, it follows that $-1 \leq 2d(P_i(x), Q_j(x) - 1 \leq 1$, with a score of 1 being assigned if the distributions are identical and a score of -1 being assigned if there is no overlap between the distributions. We propose using the function $M(S_i, T_j) = 2d(P_i(x), Q_j(x)) - 1$ as our scoring function for aligning two columns of the probabilistic sequences S and T.

3 Dynamic Programming Algorithm

Our algorithm for a global alignment of two probabilistic sequences S and T will proceed almost exactly the same as the Needleman-Wunsch algorithm, except now our recurrence will be

$$X(i,j) = \begin{cases} X(i-1,j-1) + 2d(P_i(x), Q_j(x)) - 1 & \text{Match} \\ X(i-1,j) - c & \text{Insertion?} \\ X(i,j-1) - c & \text{deletion?} \end{cases}$$

The initialization of the dynamic programming array $X(i, j)$ is the same as in NW:

$$X(i, o) = -ci \text{ and } X(0, j) = -cj$$