PyMotif

An implementation of the Gibbs sampling algorithm

Martijn Vermaat
mvermaat@cs.vu.nl

October 23, 2005

1 Introduction

Gibbs sampling is a general method used in mathematics and physics for probabilistic inference. We apply Gibbs sampling to the problem of detecting subtle patterns in DNA sequence data. Patterns common to multiple sequences are called a motif and often reflect a biological importance of the pattern.

Our instantiation of the problem is as follows: given a set of sequences S and a motif width W, find in each sequence S_i a word of width W so that the words together form a motif.

2 Implementation

The basic algorithm is explained in [1]. Our implementation runs the algorithm for a maximum of t non-improving iterations. Improvement is measured using the relative entropy of the motif found, calculated as

$$\sum_{j=1}^{W} \sum_{r \in \{A,T,G,C\}} p_{rj} \log_2 \frac{p_{rj}}{b_r},$$

where p_{rj} is the frequency of nucleotide r in position j of the alignment and b_r is the background frequency of r [2]. The motif with the highest relative entropy found is shown.

Phase Shifts

A disadvantage of basic Gibbs sampling is that it is susceptible to the "phase" problem. Consider a strong pattern at positions 9, 4, 13, . . . within the given sequences. If after a few iterations the algorithm happens to choose positions 11 and 6 in the first two sequences, it is likely to settle with position 15 for the third sequence (and so on). Because of the nature of the algorithm, it will get locked into this shifted form of the optimal pattern.

As proposed in [1], we added an extra step to the algorithm. Every f iterations, the current motif is checked for phase shifts of a certain maximum width s. Based on the relative entropies calculated from these phase shifts, a random phase shift is chosen and the algorithm continues.

Initial Positions Heuristic

Testing our implementation made clear another problem of the algorithm. Patterns likely to occur based on background frequencies of nucleotides are easy to find but often not significant for our purposes. Using background frequencies in the sampling step and entropy calculation takes care for this to some extend, but the randomly chosen initial positions for the motif introduce the problem again.

Our solution is to choose the initial positions for the motif based on the following heuristic. Highly significant patterns would in general contain one or more occurrences of the nucleotide with the lowest background frequency. Our algorithm searches for words of width v having at least n occurences of this nucleotide and chooses initial positions based on one of these words.

3 Comparison with AlignACE Results

We compared the results of our implementation with those of the AlignACE program [3] (version 4.0 dated 05/13/04).

Implementation Details

The approach taken by the AlignACE program differs in a number of ways from ours. First, it employs a multiple local alignment search, meaning that the motif may occur zero or more times in each sequence either on the forward or reverse strand. Our implementation only yields motifs occurring exactly once in each sequence's forward strand.

Second, running the AlignACE program results in a series of motifs found, ordered by descending MAP score, the metric for motif strength used by AlignACE. Our implementation results in exactly one motif.

Finally, our implementation needs a fixed parameter W for the motif width, whereas the AlignACE program independently searches for motifs of different widths.

Results

Running our implementation on 33 upstream regions of yeast genes with a motif width of 10 and default settings (no phase shifts or heuristics) typically yields a result in only 5 to 7 seconds on our workstation. The motif found usually contains 6 or 7 strongly preserved nucleotide positions (out of 10).

Varying input parameters can lead to a typical number of 7 to 8 highly preserverd nucleotide positions, but can also easily make the running time more than twice as long. Overall, we are not entirely satisfied with the effects of these variations on the algorithm. The process of finding the best parameter values tends to be cumbersome while result improvements are minimal at best.

The execution of AlignACE on the same set of sequences takes about 30 to 40 seconds and typically yields 12 to 20 motifs of differing strength. Best motifs are usually very strong, whereas some tend to be rather weak.

Comparing with the results of our implementation, the strongest motifs found by AlignACE seem to be unobtainable using single local alignment. Due to this difference in implementation it is hard to meaningly compare results. It is clear however, that our implementation usually finds motifs which are more significant than the motifs found by AlignACE with the lowest reported MAP score.

References

[1] C. E. Lawrence, S.F. Altschul, M.S. Boguski, J.S. Liu, A.F. Neuwald, J.C. Wootton. *Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment*, Science, 262(5131):208-214, 1993.

[2] Neil C. Jones, Pavel A. Pevzner. *An Introduction To Bioinformatics Algorithms*, Bradford Books (MIT Press), 2004, ISBN 0262101068.

[3] Jason D. Hughes et al. `AlignACE`. World Wide Web: `http://atlas.med.harvard.edu/`.