Elements of Bioinformatics (autumn 2010)

Lecturer: Veli Mäkinen

Exercise 4

Tue 30.11, 16-18, C222, Esa Pitkänen

Choose any 5 assignments from below (each assignment gives 1 point, 5 points is maximum for each week; you can do more for better learning, of course).

1. **Motif discovery and statistical significance.**

   Given a set of N promoter sequences each of length L, an *exact motif finding* problem can be formulated as the task of finding k-mers that occur in n out of N promoter sequences (at least once in each) and have small probability of occurring that many times in a random set of sequences following the same distribution as the promoter sequences.

   Let C_w denote the number of promoter sequences containing k-mer $w = w_1 w_2 \cdots w_k$.

   a) Derive an estimate for the expected value of C_w assuming the background follows the i.i.d. model.

   b) Why C_w divided by its expected value does not give a good ranking for reporting the statistically most significant k-mer motifs?

   c) Find out what kind of different rankings (statistical tests) are used in this kind of contexts. What do you need to know about the distribution of values C_w to use them?

2. **Generating DNA sequences with higher-order Markov chains.**

   Write a python program to read the k-th order distribution of a given DNA sequence (for given k), and to generate a new sequence of the same length simulating the same distribution.

3. **Extracting training set for HMM parameter estimation.**

   Write a python program that, given a gene name, extracts the corresponding DNA sequence with each position labeled with the information of whether belonging to exon or to intron (according to any chosen transcript). (Use SQL to find the exon/intron intervals, then extract the corresponding sequences from the chromosome files; see previous exercises for a related example).

4. **Training coding/non-coding HMM.**

   Assume a set of DNA sequences with coding/non-coding labeling (as given by the preceding assignment). Write a python program that trains the emission/transition probabilities for the coding/non-coding HMM considered at lectures, given the training data.

5. **Implementing viterbi.**

   Write a python program that implements the viterbi algorithm in the special case of the coding/non-coding HMM (or directly the general case). (Now if you have completed also preceding two assignments, you can try out whether the gene prediction works at all with this simple HMM.)

6. **Better HMM for gene prediction.**

   Sketch a visual representation of a HMM that would recognize complete genes and not just coding/non-coding areas. Take into account start/end codons, required dinucleotides at exon/intron boundaries, CAI index for frequent codons, etc. Notice, you can enhance the standard HMM definition by emissions of several symbols at a time, etc.

7. **Correctness of UPGMA.**

   Prove that UPGMA algorithm constructs an ultrametric tree if the distances are ultrametric.