Biological Data Analysis (CSE 182) : Assignment 2

Logistics

Submit using gradescope.

Sequence Alignment and Gap penalties

The first two questions should help you learn how to use alignment algorithms in a real-life setting when the DNA strings are quite large. Also, we have had quite a bit of discussion about the impact of different scoring functions on the alignment. In Q2 and Q3, you will study the impact of scoring functions on the length of the optimal alignment. If you can explain the behaviour theoretically, you get extra credit, and I hope some of you will attempt it. Finally, we deal with memory issues in Q4. As always, please cite all your sources and collaborators.

Problems

1. (28 pts.) Implement a local alignment program *locAL* to align two long DNA sequences with linear gap-penalty. The program should take two DNA sequences as input, along with user defined values for *match-score*, *mismatch-score*, *indel*. Preferably, it should be invoked as follows:
   `python locAL.py <seq_file> -m <match> -s <mismatch> -d <indel> -a`
   where the input *seq_file* is a FASTA-formatted text file containing two sequences to be aligned. The program should output the following:

   - Score of the best local-alignment.
   - Length of the best local-alignment
   - The alignment itself only if the '-a' option is specified

   Use the sequence file **p1seqs.txt** available on the course web-site. Apply the program to aligning the pair of sequences with the following parameters: match:1, mismatch -10, indel -1. Submit the code, and the output.

2. (25 pts.) Write a program to generate random DNA sequence ($pr[A] = Pr[C] = Pr[G] = Pr[T] = 0.25$) with a specified length. Preferably, your program should be invoked similar to the following:
   `python randomDNA.py <number of seq> <size of seq>`
   It should output the random sequences, one per line. After the sequences, the program should calculate and output a summary of the observed nucleotide frequencies in the set of sequences. Use randomDNA to generate 500 pairs of sequences. Each pair has two sequences of length 1000bp each. Align them using *locAL* using two sets of parameters:

   - **parameter P1:** match 1, mismatch -30, indel 0
   - **parameters P2:** match 1, mismatch -30, indel -20

   Plot a histogram of lengths of the local alignment using parameters P1, and P2, and *submit the plot.* Are the lengths of the optimal local alignments different? If so, why?

   To confirm your hypothesis, try the same experiment with multiple random pairs of different length. Define $l_p(n)$ as the expected length of the optimal local alignment for a pair of random sequences of length n. Your computations should give you estimates of $l_{P1}(n)$, and $l_{P2}(n)$ for different values of n. Can you guess the form of $l_{P1}(n)$, and $l_{P2}(n)$, as a function of n? **Extra Credit:** Provide a theoretical justification.

3. (25 pts.) **Phase Transition of Local Alignments:** Define $l_P(n)$ as in Problem 2. Clearly, the parameter set P can change $l_P(n)$.

(a) Fix $n = 1000$. Plot the mean values of $l_P(n)$ for a variety of parameter settings P. For example, you can choose mismatch=indel from $\{-30, -20, -10, -1, -0.5, -0.33, -0.25, 0\}$, and *submit a plot*. You can use matplotlib or another python package for plotting but you must have written the alignment code yourself.

(b) Is there an abrupt change in the value of $l_P(n)$? If so, use more parameters to check how sharp the transition is, to identify the parameters at which the change happens? **Extra credit/Research:** Provide a theoretical justification.

4. (20pts) Run locAL on the pair of sequences in the file called **p4pairs.txt** available on the course website, and output the length of the alignment, the alignment score, *and the alignment itself.* Use match:1, mismatch -10, indel -1. As the sequences are long, you must devise a method to compute a local alignment without necessarily programming the Hirschberg trick (In fact, we did NOT cover the Hirschberg method in this class). Your submission should include a succinct description of how you used only linear space.

   Hints: Note that your code outputs both the score and the length of the optimal local alignment, and you can use that for linear space computations. Second, the computation can take a long time to run. Please start early, run with smaller length sequences to estimate the time your code will take; make sure you are not using recursion instead of dynamic programming; and, do some checkpointing; the ability to stop the code and re-start from a different portion of the alignment.

5. (2pts.) What language did you use? How much time did you take to do the assignment? Who did you discuss your homework with?