Algorithms in Computational Biology, 2021
Exercise 3 - Programming - Expectation-Maximization

Due date: 20/12/2021

Submission should include a tar file named *ex3.tar*, containing the following files:

- ZOOPS_EM.py
- Other python files are optional. Include relevant code from ex2.

EM using ZOOPS

In this exercise we will use the same ZOOPS model as in the previous exercise (ex2). Write a program that finds a motif's position and parameters using the Expectation-Maximization (EM) algorithm. The program takes as input a FASTA file with multiple sequences, and initial guesses for the model parameters, and outputs the motif's positions in all sequences, and the learned parameters.

Expectation-Maximization

Write the ZOOPS_EM.py program, that finds a motif's position and parameters unsing the Expectation-Maximizaiont (EM) algorithm. The program takes as input a FASTA file with multiple sequences and initial guesses for the model parameters and outputs the motif's positions in all sequences, as well as the learned parameters. Use the Forward & Backward implementations from your previous ex2 implementation.

Initialization

The EM algorithm requires an initial guess of the parameters: emission and transition probabilities.

Emission probabilities initialization

Your program gets as arguments a seed (motif) and a softening rate α. This is the initial guess of the motif. The emission probabilities should be constructed as follows:

- Softening parameter α: Set the initial emission probabilities for the motif states to have $1 - 3 \cdot \alpha$ for the motif letter, and α for others letters.

  For example suppose the seed is ATTA, and $\alpha = 0.1$. Then set the initial guess for emission of the motif state to be:

$\Sigma \backslash pos$	1	2	3	4
A	0.7	0.1	0.1	0.7
C	0.1	0.1	0.1	0.1
G	0.1	0.1	0.1	0.1
T	0.1	0.7	0.7	0.1

- Uniform background state probabilities.

Transition probabilities initialization

- q, p - The initial estimate for the transition probabilities are given as command line arguments.

Training:

the program should run the EM algorithm. Note that the only transition probabilities to be learned are p, q and the only emission probabilities to be learned are those of the states representing the motif (hence, background remains fixed at $(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$).

Specifications

It should be possible to invoke the program from the command line using the following format:

```
python3 ZOOP_EM.py fasta seed p q alpha convergenceThr
```

Input

- $fasta$- File name of the list of sequences in which to search for the motifs. $example1.fasta$ is a file example.
- $seed$ - The guess for the motif (e.g. ATTA).
- p, q - Initial guess for the transition probabilities
- $alpha$ - Softening parameter (α) used to create the initial profile (as described above).

- *convergenceThr* - The stopping condition for the EM is a threshold on the improvement of the log likelihood. Once the improvement of the lod likelihood is below this threshold you should stop. Meaning, when $convergenceThr > ll_t - ll_{t-1}$, where ll_t is the log-likelihood at the end of the t'th iteration.

Output

The program should output the following files:

- *ll_history.txt* - A file containing the log likelihood of the sequences at each iteration, seperated by newlines. There shoud be no header line. For example, if the algorithm converged after 50 iterations, this file should have exactly 50 lines. **Make sure they are monotonically decreasing!**
- *motif_profile.txt* - A file containing the profiles. The file has two parts:
  - Emmisions: The first four lines of the file are the profile (A,C,G and T respectively), each column is a position in the motif (tab separated), and the content of this matrix is the probability of observing this letter at this position (written with a precision of 2 digits).
  - Transitions: Line 5,6 are the predicted values of q, p probabilities, respectively (precision of 4 digits).
- *motif_positions.txt* - The i'th line is the predicted position (0-based) of the motif within the i'th sequence in the FASTA file. The positions should be found using Viterbi. For sequences in which the motif does not occur, the output position should be -1.

Tips and Implementation issues:

- Your program should run fast. Part of the grade will be the speed of your program. Avoid using pandas, instead use numpy and try to compare running times. Your forward & backward methods run many times in the EM process. Use as few nested loops as possible.
- Assume valid input, e.g. $\alpha < \frac{1}{2}$, $p > 0$, $(q = 1) \implies$ (Input sequence contains motif). valid non-empty fasta file, etc.
- Pay attention to the printing order (e.g. p vs q).
  - wrong order will be penalized

Grading

Improperly formatted tar files will be penalized. Make sure your tar file can be opened with the command 'tar -xvf my_tar_file.tar' and then without changing directory you are able to run your program with the command stated above.