CS 3824
Homework Assignment 5

Given: October 28, 2014 **Due:** November 15, 2014

General directions. The point value of each problem is shown in [ ]. Each solution must include all details and an explanation of why the given solution is correct. In particular, write complete sentences. A correct answer without an explanation is worth no credit. The completed assignment must be turned in as a PDF through Scholar by 5:00 PM on November 15, 2014. **No late homework will be accepted.**

Digital preparation of your solutions is mandatory. Use of LaTeX is optional, but encouraged. No matter how you prepare your homework, **please include your name.**

Use of LaTeX (optional, but encouraged).

- Retrieve this LaTeX source file, named `homework5.tex`, from the course web site.
- Rename the file <*Your VT PID*>`_solvehw5.tex`, For example, for the instructor, the file name would be `heath_solvehw5.tex`.
- Use a **text editor** (such as `vi`, `emacs`, or `pico`) to accomplish the next three steps.
- Uncomment the line

  `% \setboolean{solutions}{True}`

  in the document preamble by deleting the `%`.
- Find the line

  `\renewcommand{\author}{Lenwood S. Heath}`

  and replace the instructor's name with your name.
- Enter your solutions where you find the LaTeX comments

  `% PUT YOUR SOLUTION HERE`
- Convert your solutions to PDF and submit your solutions through Scholar by 5:00 PM on November 15, 2014.

---

[50] 1. Jones and Pevzner problem 11.4.
Assume that the first state is α.

---

[50] 2. In this problem, we analyze a particular aspect of finding motifs with `RandomMotifFinder.py`. Consult the project assignment for relevant terminology and notation.

We are interested in the log-likelihood value that results from one iteration of the `while` loop in `RandomMotifFinder.py`. For simplicity, we assume that there is no planted motif, so all of the input strings are randomly generated according to the simplest null model, where each nucleotide is chosen with probability 0.25. After randomly selecting s, `RandomMotifFinder.py` is then examining an $n \times k$ matrix of nucleotides, where d columns are "don't cares". In each column that is not a "don't care", it is choosing the most frequent nucleotide to put in that location in M.

A. What is the minimum value that $LL(M, s)$ can take, as a function of k, d, and n?

B. What is the maximum value that $LL(M, s)$ can take, as a function of k, d, and n?

C. We now consider the expected (average) value for $LL(M, s)$. Looking at the formulas for $L(M, s)$ and $LL(M, s)$, it is clear that we can concentrate on the distribution of counts in an arbitrary column that is not a "don't care". There are 4^n possible values for that column, each equally likely. `RandomMotifFinder.py` chooses the nucleotide in that column that has the maximum count. Hence, the count coming from that column is between $\lceil \frac{n}{4} \rceil$ and n. Write a program called `MaxInColumnDistribution` to compute the number of times (in the 4^n possibilities) that each such maximum count occurs and convert the counts to a probability distribution. For example, the probability distribution for $n = 2$ is

count	probability
1	0.75
2	0.25

,

while for $n = 4$, it is

count	probability
1	0.09375
2	0.703125
3	0.1875
4	0.015625

.

In addition, your program should compute the expected log-likelihood for one column, given this distribution. (You can multiply by $k - d$ to get the expected log-likelihood for the best motif of length k with d "don't cares".)

Report the computed distribution and expected log-likelihood for a column for $n = 10$ and $n = 12$. Include the source code of your `MaxInColumnDistribution` in your PDF.

Such analysis programs can be made part of your toolkit to evaluate the performance of your `MotifFinder`.