BMI/CS 576 Fall 2015
Homework #4

Prof. Colin Dewey

Due Wednesday, November 18th, 2015 by 11:59pm

The goal of this assignment is to become more familiar with Markov chains and hidden Markov models.

To turn in your assignment, copy all relevant files to the directory:
`/u/medinfo/handin/bmi576/hw4/USERNAME`
where `USERNAME` is your account name for the BMI network. Please note the homework policies posted at `http://www.biostat.wisc.edu/bmi576/hw.html`

1. (20 points) An important property of a Markov chain is the probability distribution over the lengths of sequences it generates.

    (a) Give the distribution of lengths, $P(\ell)$, for sequences generated by the Markov chain depicted in Figure 1 below.

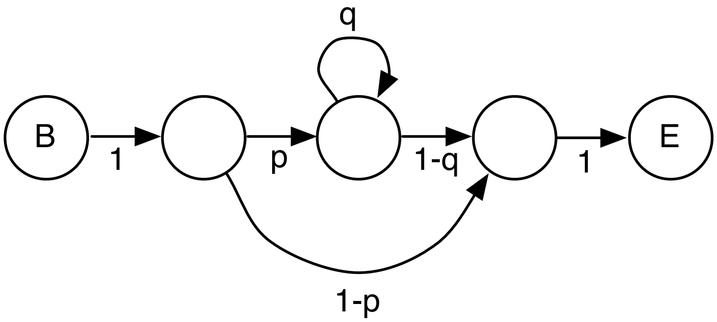

Figure 1: The state transition diagram for the Markov chain of problem 1(a). B and E are the silent begin and end states.

    (b) Give the state-transition diagram for a Markov chain with length distribution given by

$$P(\ell) = \begin{cases} 0 & \text{if } \ell = 0 \\ (1-p)(pq)^{(\ell-1)/2} & \text{if } \ell > 0, \text{ odd} \\ (1-q)p(pq)^{(\ell-2)/2} & \text{if } \ell > 0, \text{ even} \end{cases}$$

2. (20 points) The optimization problem of Homework 2, Problem 1 (partitioning a DNA sequence into segments that are predominantly composed of a single base) can be approximately modeled by a hidden Markov model.

   (a) What are the observed data?

   (b) What are the hidden states of the HMM?

   (c) What do the emission parameters represent? How do they relate to the parameters b and c in the optimization problem?

   (d) What do the transition parameters represent? How do they relate to the parameter a in the optimization problem?

   (e) Which HMM algorithm should be used to partition a DNA sequence?

3. (40 points) Write a program, `predict_exons`, that takes as input two sets of mRNA sequences, trains an exon-predicting HMM based on the first set, and outputs predictions of the exons on the second set.

   Your program will use the very simple HMM structure shown in Figure 2. The begin and end states are silent and the exon and intron states both emit one character at a time. The program should train the parameters (both emissions and transitions) of the HMM using the training sequence data and standard maximum likelihood estimates with *Laplace* smoothing (i.e., adding one "pseudocount" to each count). It should then run the Viterbi algorithm on each test sequence to predict which positions are exonic and which positions are intronic. In order to avoid numeric underflow issues, you will need to use logarithms of probabilities in the Viterbi algorithm instead of the probabilities themselves (see section 3.6 of the textbook).

   Your program should be run via a simple shell script `predict_exons.sh` that calls your program in the appropriate manner. A template script is available from:
   `http://www.biostat.wisc.edu/bmi576/hw/hw4/predict_exons.sh`
   The script will take two command-line arguments: the name of a file containing a set of mRNA sequences on which you should train your HMM, and the name of a file containing a set of mRNA sequences for which you should predict the positions of the exons. For both files, each sequence will be given on a separate single line. The training sequence file will have exon positions in UPPERCASE and intron positions in lowercase. The testing file will have all characters in lowercase (i.e., the locations of the exons are hidden). Although we are modeling mRNA, we will still use A, C,

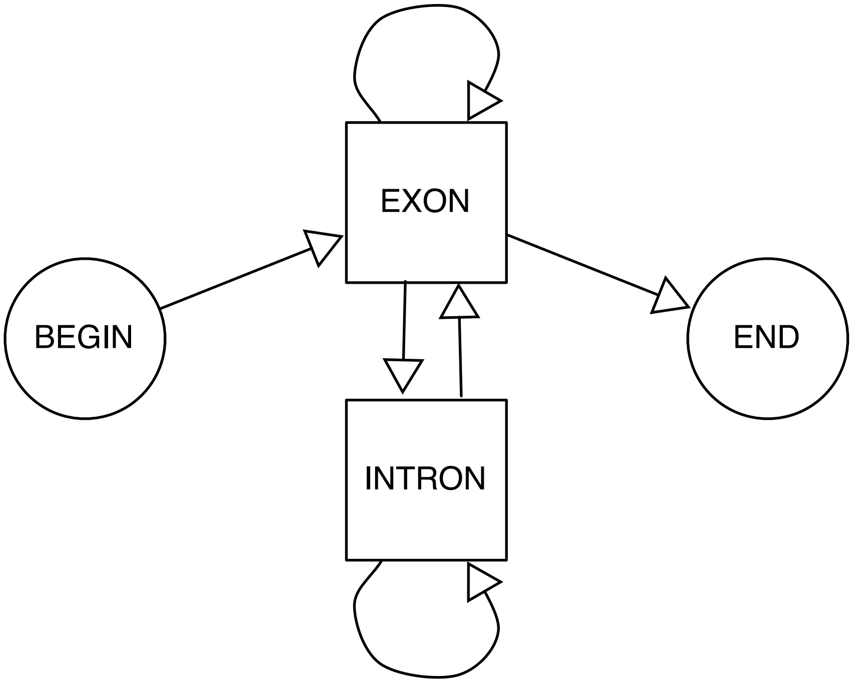

Figure 2: The state-transition diagram of the simple exon-intron HMM you will use for Problem 3.

G, and T as our characters. Your program should output its predictions by printing to the standard output stream each test sequence on a separate single line, with predicted exon positions in UPPERCASE and predicted intron positions in lowercase (just like the training data).

For example, using the small example input provided at `http://www.biostat.wisc.edu/bmi576/hw/hw4`, below is a command we should be able to run with its expected output:

```
$ sh predict_exons.sh small.train small.test
ATGTTAAgtggccagTTAATGA
ATGAAAAgtggggccagTAATGA
ATGGTCAGTAG
```

Sample input and output files, are provided at: `http://www.biostat.wisc.edu/bmi576/hw/hw4/` Your program must run in less than 5 minutes on mi1.biostat.wisc.edu for the training and test sets on the Web site.

4. (10 points) Write a program, `exon_accuracy`, that takes as input true and predicted exon annotations for a set of mRNAs, and outputs the accuracy, recall, and precision

of the exon predictions. The definitions of the three measures will be as follows:

$$\begin{aligned} accuracy &= \frac{\text{\# of correctly predicted positions}}{\text{total \# of positions}} \\ recall &= \frac{\text{\# of correctly predicted exonic positions}}{\text{total \# of true exonic positions}} \\ precision &= \frac{\text{\# of correctly predicted exonic positions}}{\text{total \# of predicted exonic positions}} \end{aligned}$$

Your program should be run via a simple shell script `exon_accuracy.sh` that calls your program in the appropriate manner. A template script is available from:
`http://www.biostat.wisc.edu/bmi576/hw/hw4/exon_accuracy.sh`
The script will take two command-line arguments: the name of a file containing a set of mRNA sequences with *true* exon annotations and the name of a file containing the same set of mRNA sequences, but with *predicted* exon classifications. Both files will be of the same format as for the `predict_exons` program, with exon nucleotides in UPPERCASE. The program should write to the standard output stream the accuracy, recall, and precision values, in that order, with one value per line. The values should be rounded to three decimal places.

For example, using the small example input provided at `http://www.biostat.wisc.edu/bmi576/hw/hw4`, below is a command we should be able to run with its expected output:

```
$ sh exon_accuracy.sh small.test.truth small.out
0.821
1.000
0.737
```

5. (10 points) Run your `predict_exons` program with the human mRNA training and test data on the Web site and evaluate the accuracy of your predictions with your `exon_accuracy` program.

   (a) What are the accuracy, recall, and precision of your predictions?

   (b) Examine the emission probabilities of your trained model. What signal is your model using for predicting exons?