

**Homework 2**  
**Building Profile HMMs**  
**Due: February 19, 2009**  
**50 points**

Your task is to write and test a program in C, C++, Java, Perl, Python or Matlab for Andrew Unix, Linux, MacOs or Windows that will find build models of sequence features using profile Hidden Markov Models (HMMs). Your program should:

1. parse from the command line the following inputs
  - a. name of a FASTA file containing a set of aligned sequences
  - b. a character indicating whether the sequences are amino acid (A) or nucleic acid (N)
  - c. the number of match states to be used in the HMM
  - d. the number of folds of cross-validation to use
  - e. the name of an output file
2. read the FASTA file
3. build a profile HMM for the specified number of match states using all sequences (see below)
4. calculate the best match between the profile HMM and the sequences, print the score for each, and remember them
5. divide the sequences into the specified number of folds
6. for each fold, use all other folds as training data, build a profile HMM, calculate the best match between that profile and the sequences in the test fold **ONLY**, remember the ratio between that score and the score for the corresponding sequence found in step 4, and print the average ratio for all test sequences in that fold
7. print the average ratio for all test sequences for all folds

Be sure to use a modular design, follow good programming practice and provide good internal documentation for your code. **Be sure to develop an adequate test procedure for your program and use it to test your program thoroughly.**

Your program can use any implementation of profile HMMs that you wish, but (other than the two recommendations below) you will need to include any code that you call with your submission.

The two recommended implementations of profile HMMs are HMMER and the version in the Matlab bioinformatics toolbox. You can get HMMER from <http://hmmer.janelia.org/>

Create a single zip, gzip, tar, or jar file containing all of the items below and submit it through the Blackboard drop box. Name the file with your Andrew username and the homework number, e.g., smith-hw2.zip

Provide the following files:

1. The source files (including required shell scripts listed below, include files, etc.)
2. A thorough description of the test procedures you used to verify that the program operates properly, including results from these tests and how that you know the results are correct
3. Any scripts or files that you used in testing

### **About program grading**

The homework will be graded using grading scripts so specific naming and input/output conventions must be followed in order to receive credit.

You must include a shell script named “compile” that compiles the program (if compilation is not necessary, make an empty “compile” file) and a shell script named “runme” that takes the arguments from the command line and runs your program using them.

Example command “runme fs00002.fa A 32 8 output.txt”

### **Testing restriction (added February 12, 2009)**

Given that HMMER does not allow the number of match states to be specified, we will adopt the following restriction for testing of your programs. The third argument (number of match states) will always be given as the value that HMMER would choose for the alignment file given in the first argument. For example, HMMER chooses 301 as the number of match states for file pf00002.fa. Thus we would invoke your programs with (assuming two-fold cross-validation)

```
runme pf00002.fa A 301 2 output.txt
```

Those using the Matlab toolbox should then get comparable results to those using HMMER. Note: If you use HMMER, your program is responsible for checking that the number of match states determined by HMMER corresponds to the number specified on the command line. If not, an error message should be given and your program should exit without writing to the output file.

### **Extra credit (15 points each)**

Modify your program so that it finds profile HMMs for numbers of match states from 2 to 100 (using only the original sequences) and generates a plot of the average ratio versus the number of match states.

Modify your program so that it finds profile HMMs for different numbers of training examples (using the sequences from the input file) and plots the average ratio versus the number of training examples. The number of training examples should start at 1 and increase by 1 to one less than the total number of sequences in the input file. The average score should be an average over all

sequences (including training examples), and averaged over 5 random trials of which sequences are used for each number of training examples.

Write a program to build a PSSM from the aligned sequences in the FASTA file, print it out in a formatted table, and score each of the original sequences using the PSSM. Use a scoring scheme in which the score at each position is calculated using the log likelihood method described in the lecture notes.