CPSC 422,
Homework 2

General Instructions:

- Working in Teams: You may work with at most one other person. That person must also be enrolled in CPSC 422 this term. If you are working with a partner, the two of you must submit only one assignment, listing both of your names. Keep in mind that when you work in pairs, each of you must understand the work that you submit.
- Note: You are permitted to derive your solutions by implementing the respective algorithms, even when it is not explicitly required in the question. You are NOT permitted to take existing implementations of the algorithms and use those for your solutions. Such existing algorithms include those available from friends, CD, or off the web. You are also NOT permitted to provide your implementation to anyone else outside your team.
- Important: It is best if you type your assignment. Handwritten work will be marked only if clearly legible, and will still have to be submitted via handin.
- Make sure to include your name (or the names of the two team members if you are working with someone else) and the assignment number, at the top of the assignment. Ensure that each question and sub-question is appropriately labeled and clearly identifiable.

1 [60 points] Likelihood Weighting Particle Filtering in HMMs

Consider an HMM with $P(X_0 = t) = 0.5$, transition matrix

X_t	$P(X_{t+1} = t\|X_t)$
t	0.8
f	0.1

and observation matrix

X_t	$P(E_t = 1\|X_t)$
t	0.95
f	0.1.

Given a set of observations, $e_1, \ldots, e_T$, in this question, we will apply particle filtering (PF) and likelihood weighting (LW) to compute the filtering probabilities $p_t = P(X_t = t|e_1, \ldots, e_t)$. A series of 100 observations $e_1, \ldots, e_{100}$ (one per line) is provided in file `observations.txt`. In HMMs, we can compute the exact filtering probabilities with the forward phase of the forward-backward algorithm. For $t = 1, \ldots, 100$, we provide the exact probabilities $P(X_t = t|e_{1:t})$ in file `exact-probs.txt` (again, one per line). You will use these to evaluate the estimates of PF and LW. For the programming questions, we recommend MATLAB due to its friendliness with vector and matrix functions. However, you can use any programming language of your choice.

1. **[25 points]** Implement likelihood weighting to compute the filtering probabilities for this network. Run your implementation with $N = 1\,000$ samples to compute estimates $\hat{p}_{t\,LW}$ of p_t for $t = 10$ and $t = 100$. Also compute and report the mean absolute error $e_{LW}(t) = \sum_{i=1}^{t} |\hat{p}_{i\,LW} - p_i|$ for $t = 10$ and $t = 100$. Submit an electronic copy of your code.

2. **[25 points]** Implement a simple particle filter for this network, where particles for the next time step are sampled from the HMM transition matrix, weighted by their observation likelihood and re-sampling is used at every step. Run your implementation with $N = 1\,000$ samples, to compute estimates $\hat{p}_{t\,PF}$ of p_t for $t = 10$ and $t = 100$. Also compute and report the mean absolute error $e_{PF}(t) = \sum_{i=1}^{t} |\hat{p}_{i\,PF} - p_i|$ for $t = 10$ and $t = 100$. Submit an electronic copy of your code.

3. **[10 points]** Discuss the performance of the 2 algorithms in relation to the length of the sequences.

2 [30 points] Value Iteration

In this question, you will be using an applet to improve your understanding of value iteration. You can find the applet at http://www.cs.ubc.ca/spider/poole/demos/mdp/vi.html

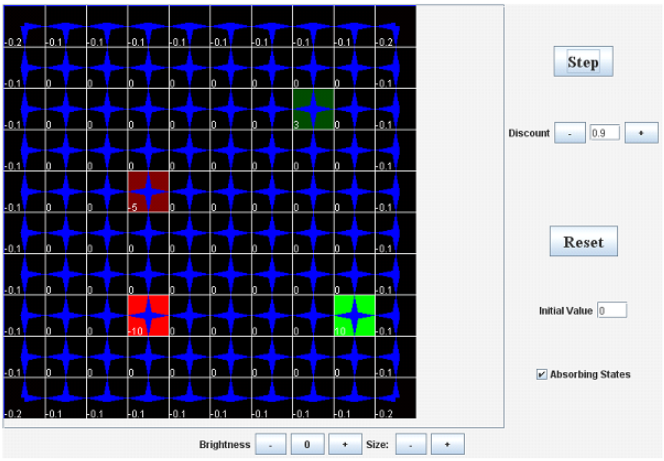
There are some questions listed on that website; for this assignment, please disregard those questions and only answer the following ones. In this assignment, we are using a discount factor of 0.9, initial values of $U^{(0)}(s) = 0$ for all s, and the "absorbing states" option (explained in detail on the website with the applet). We will refer to states as (x, y), meaning the state in the x-th column and the y-th row: e.g. $(1, 1)$ for the state at the top left, and $(10, 1)$ for the state at the top right.

(a) **[10 points]** The figure below shows the values $U^{(1)}(s)$ in each state, that is, the values after one step of value iteration. We will focus on the entry in a single state, namely state

$(10, 8)$, the state to the right of the absorbing state with reward 10 (which is located at $(9, 8)$). Show in detail how $U^{(1)}((10, 8))$ is computed using the values $U^{(0)}(s)$. [Hint: In this world, the reward in a state depends on both that state, the action taken and the resulting outcome (for instance, a move up from $(10, 8)$ has a 0.1 chance to crash into the wall and get a negative reward. With a move to the left, the probability of getting a negative reward goes up to 0.7). Youll need to modify the Bellman equation we saw in class so that it accounts for the fact that the reward does depend on the action taken]

(b) **[10 points]** The figure below shows the values $U^{(2)}(s)$ in each state, that is, the values after two steps of value iteration.

- Show in detail how $U^{(2)}((10, 8))$ is computed using the values $U^{(1)}(s)$.
- Intuitively, why is $U^{(2)}((9, 9))$ larger than $U^{(2)}((10, 8))$?
- Intuitively, why is $U^{(2)}((9, 7))$ larger than $U^{(2)}((9, 9))$?



(c) **[10 points]** We now study the importance of the discount factor. With the option absorbing states enabled and discount factors of 0.8, 0.9, and 0.999, repeatedly perform steps until value iteration converges.

- Show the optimal policies for these three discount factors (just hand in screen shots of the applet after value iteration converged.)
- Why do the optimal policies change for the states around the absorbing state with reward 3 at $(8, 3)$, depending on the discount factor?
- Why do the optimal policies change for the states $(2, 6)$, $(3, 6)$, $(2, 7)$, $(3, 7)$, depending on the discount factor?

3 [30 points] HMMs for Finding CpG Islands in the Human Genome

Hidden Markov Models (HMM) can be used to solve a variety of important problems in bioinformatics, such as sequence alignment, gene finding, and identifying so-called `CpG` islands. In

this question, we will concentrate on this latter task.

We first provide some background information on CpG islands, quoted from Durbin, Eddy, Krogh, and Mitchison's book *Biological Sequence Analysis: probabilistic models of protein and nucleic acids*, chapter 3:

> In the human genome wherever the dinucleotide CG occurs (frequenly written CpG to distinguish it from the C-G base pair across the two strands) the C nucleotide (cytosine) is typically modified by methylation. There is a relatively high chance of this methyl-C mutating into a T, with the consequence that in general CpG dinucleotides are rarer in the genome than would be expected from the independent probabilities of C and G. For biologically important reasons the methylation process is suppressed in short stretches of the genome, such as around the promoters or 'start' regions of many genes. In these regions we see many more CpG dinucleotides than elsewhere, and in fact more C and G nucleotides in general. Such regions are called CpG islands [Bird 1987]. They are typically a few hundred to a few thousand bases long.

In this assignment, we will develop an HMM-based mechanism to find CpG islands in long pieces of the human genomic sequence. As input, we have one piece of the human genomic sequence, strings over the alphabet of nucleotides, C,G,A,T. Each nucleotide of these sequences was labelled by the human genome project as belonging to a CpG island or not. We will train our HMM-based CpG island-detector on one sequence and use it to predict the CpG islands in the other sequence.

First, you will need to think about how to represent this domain in the HMM framework to use an out-of-the-box implementation of the Viterbi algorithmwe provide. The 4 possible observations at each "time" step in our HMM correspond to the true nucleotides in the sequence.

Zip file `CpGislands.zip` contains the following files:

- **HMM.java** This contains a code that, given the HMM prior, number of hidden states, and a sequence of observations, learns the transition and observation matrices. The learning is done in the constructor of HMM based on the frequency counts of the sequence in the training set. More precisely, the transition probability between any two states $P(x_{t+1} = w|x_t = z)$ is determined by how frequently w occurs after z in the training sequence. A pseudocount is also added to the actual counts for smoothing the probability model. Intuitively, this counts for possible transitions that have not been seen in the training sequence. The observation probability $P(y_t = y|x_t = z)$ is the size of the samples in which y is observed from z relative to the number of samples in which z occurs. The file also includes an implementation of the Viterbi algorithm (**HMM.viterbiPath**), which you will use in this question. You do not need to modify this file.

- **CpGislandDetector.java** This reads in a sequence and splits it into training and test sets (the test set is only used for an unbiased estimation of how well the detector works). The training set is further split into k folds, which are used for cross validation. These k folds are referred to as train and test folds, and are used to learn a more robust HMM model by learning and testing it on different sets of data (more on this later). This file also calls the **HMM** constructor to learn the transition and observation matrices and calls **HMM.viterbiPath** to find the Viterbi path.

- **LabelledSequence.java** This class contains a char array of nucleotides {C,G,A,T}, as well as a boolean array indicating for each nucleotide whether it is in a CpG island or not. It also contains a stub for a method you need to write for this assignment: `extract-`

`ObservationsAndHiddenStates`, which translates this information into int arrays holding observations and hidden states of the HMM.

- **chr22.genbank.cpg.txt** and **contig1.fasta**: data files containing part of the human genome and information about where `CpG` islands are located. You should not have to worry about these since our parsing code in `CpGislandDetector.java` already extracts a `LabelledSequence` from these files.

(a) [**10 points**] Assuming the transition probabilities between any two elements of {C,G,A,T} differ, and also assuming that they are different inside and outside `CpG` islands, how many hidden states do you need to capture this domain in an HMM? (Hint: deterministic observation probabilities are ok.)

(b) [**10 points**] Implement procedure `extractObservationsAndHiddenStates` in class **LabelledSequence**, defining the observations and hidden HMM states for the labelled sequence. Hand in an electronic copy of your code.

(c) [**5 points**] We have initially set the number of folds to 5 (**kfold=5**). Run **CpGislandDetector.java** and give the HMM learned from 5-fold cross validation. In particular, print out the prior, the transition matrix, and the observation matrix. Currently, the CpGisland chooses the model with maximum performance on the test folds. How accurate is this model on the test set? What is the performance of the model on the test fold?

(e) [**3 points**] Read the main loop in **CpGislandDetector.java** and modify the file such that the selected model **hmm** is the average of the 5 models learned from the training set (i.e., test and train folds). What is the accuracy of the averaged model on the test set? Hand in your code.

(d) [**2 points**] Set the number of folds (**kfolds**) to 1 and repeat the experiment. What happens to the accuracy on both the test set and the train fold?

4 [6 points] Survey

[Note that this question is worth marks, so dont forget to do it.]

(a) [**2 points**] For each question in this assignment, say how long you spent on it.

(b) [**4 points**] Rate each question in this assignment by how useful it was to help you understand the related topics. 0 (very little) to 5 (a whole lot), and explain the reason of your score