Assignment 2

Instructions

1. This assignment is due at 5pm on 04th Mar 2014.
2. There are two portions to hand in for this assignment: Submit the derivations on paper to the course mailbox at the DSAP office, and submit a file that contains `R` code via the workbin on IVLE.
3. `R` has to be submitted for question 4, not any others.
4. Name your `R` file according to this convention. If your matriculation number is a999999u, then name the file `a999999u_assignment_02.R`.

Questions

1. Use the inversion-by-truncation method to derive an algorithm for generating X, when X has the following pmf:
$$p_i = \frac{3i^2 + 3i + 1}{i^3(i+1)^3}, \quad \text{i =1,2,...}$$
Be sure to explicitly write down the 2 steps of the final algorithm.

2. Suppose we have a finite, discrete distribution with pmf given by $p_0, p_1, \ldots, p_{K-1}$. Prove that there is an $i \in \{0, 1, \ldots, K-1\}$ such that $p_i < \frac{1}{K-1}$. This provides part of the basis for the alias method. *Hint: consider using a proof by contradiction.*

3. In the two-table method from the lectures, we extended table A to ensure it had M_2 elements. Here, we shall consider a modified version of the two-table algorithm, where we do not extend table A. In other words, after deriving the a_i's and b_i's, we shall leave table A as it is; we shall not extend it by adding $*$s. Table B will still have $\sum b_i$ entries as before. Using the same notation as in the lectures, prove that the algorithm below returns the value i with the following probability:
$$\Pr(i \text{ is returned}) = \frac{k_i}{M}$$
   (a) Generate $U, V \sim Unif[0, 1]$.
   (b) If $(U \leq \sum a_i / M_2)$, then
      i. Set $X = A[\lfloor \sum a_i V \rfloor + 1]$.
   (c) Else,
      i. Set $X = B[\lfloor \sum b_i V \rfloor + 1]$.
   (d) Return X

4. Consider the following discrete pmf:
$$p_0 = 0.05, \; p_1 = 0.1, \; p_2 = 0.7, \; p_3 = 0.1, \; p_4 = 0.05$$
   (a) Set up the alias method for this distribution and write a function to implement it in `R` . The `R` function should take two *vector* arguments: `u` and `v`. It should return a vector corresponding to draws from p_i.

(b) Derive the rejection algorithm to generate from p_i using the following candidate distribution:

$$q_i = 0.2 \quad i = 0, 1, 2, 3, 4$$

Be sure to write down the acceptance criterion and the rejection constant c for this algorithm. Implement this as a function in `R` . This rejection function should take in 0 arguments, and repeat a loop until the acceptance criterion passes once. The output of this function should be a vector of length 2. The first coordinate will be the value of X drawn, and the second coordinate will be the number of iterations of the rejection method for that draw.

(c) Generate 5000 draws from p_i using each of the algorithms in parts (a) and (b). Compare the running times of the two approaches, and the *total* number of random variables used in each case. Use the `table` function to check that the pmf for the sample obtained is close to the true vector for both methods.