Practical 2 – Simulation and Statistical Programming

Try to reach Questions and 4 and 5. Q5 is a bigger question (you have to write two functions) but worth trying.

Q1. Write a function that uses a `for` loop to sum a geometric series with `n` terms, first term `a` and common ratio `r`. The function should take three arguments : `a`, `r` and `n` and should return the sum. Check that it works using the formula for the sum of a geometric series.

Q2. Sieve of Eratosthenes. Here is a plan I wrote for an R function to find all primes between 2 and n inclusive:

a) Create a vector s of numbers from 2 to n.
b) The first entry 2 is a prime. Find all multiples of 2 and remove them from the vector s.
c) The vector is shorter now because we removed 2 and all its multiples. It has a new first entry (ie 3). This first entry hasn't been eliminated so it cant be divisible by any smaller number. It must be a prime.
d) Repeat this until the vector s is empty, saving the primes as we go.

The following code implements this, but is out of order. Put it in order!

```
while (length(s)>0) {
i=which(s%%p==0)
primes=c() #start with no primes
}
primes=c(primes,p)
if(n < 2) stop("Input value must be >= 2")
Eratosthenes = function(n) {
s=s[-i]
}
p=s[1]
return(primes)
s = 2:n
# return all prime numbers up to n
```

Q3. (optional) Write an R function that returns the real roots of the quadratic $ax^2 + bx + c$. The function should take `a`, `b` and `c` as arguments and return appropriate messages if the values entered don't specify a quadratic or if there are no real roots. Use the function to determine the roots of $2x^2 - x – 3$.

Q4. Last week we saw we could sample a discrete distribution p=(p_1,p_2,…p_m) by simulating u~U[0,1] (`u=runif(1)`) and looking for the smallest x in {1,2,…,m} such that

$$u < p\_1 + p\_2 + \ldots + p\_x$$

For example

```
> p<-c(0.1,0.2,0.3,0.4)
> cp<-cumsum(p)
> min(which(runif(1)<cp))
```

Make sure you understand how the above code works. Write a function which takes as input a pmf `p=c(p[1],p[2],...,p[m])` and a number `n` and returns `n` samples `X[1],X[2],...,X[n]` distributed according to `p`. Comment on how to test your function.

Q5. Write an R function to simulate X~N(0,1) using rejection with proposal Y~exp(-|x|).

i. Write a function to simulate n iid values of Y. Make the default n-value n=1.
ii. Write a function implementing rejection for X. Recall the algorithm from Q4 PS1:
    [1] simulate Y~exp(-|x|) and U~U(0,1)
    [2] if U<exp(-y^2/2+|y|-1/2) accept X=y and stop. Otherwise repeat [1].

    Hint: you can do this using a `while` statement. You should call the function you wrote in Q2.i to simulate Y.

    Your function should have no inputs, and return the simulated value of X.
iii. Test your rejection sampler by simulating 1000 samples and checking they are normal using the `qqnorm()` function.

Q6. The equation $0 = x^7 + 10000x^6 + 1.06x^5 + 10600x^4 + 0.0605x^3 + 605x^2 + 0.0005x + 5$ has exactly one real root.

(a) Plot the function to try to get a sense of where the root might be?
(b) Write an R function that applies Newton's method to find the root. The function should have 2 arguments : the initial value `x0` and the tolerance value. The function should return the estimated solution, the function value at the estimate and the number of iterations.
(c) What happens when you set `x0 = 0`?
(d) What happens when you set `x0 = 1`?