Stat580 - Homework 1

Alex Shum

February 12, 2014

Problem 1

$X_1, X_2, \ldots, X_n \sim Unif(0,1)$.
$(\Sigma_{i=1}^n X_i) mod 1 = \Sigma_{i=1}^n X_i - \lfloor \Sigma_{i=1}^n \rfloor \sim Unif(0,1)$.

Proof:

Problem 2

Let F be a cumulative distribution function and let $F^{-1} = min\{x|F(x) \geq u\}$. If $U \sim Unif(0,1)$ then $F^{-1}(U) \sim F$. We start with the cumulative distribution function for $F^{-1}(U)$: $P(F^{-1}(U) \leq x)$

$$\text{Applying F to both sides (F is monotonic): } P(F^{-1}(U) \leq x) = P(U \leq F(x))$$
$$\text{But since U is uniform: } P(U \leq F(x)) = F(x)$$

Problem 3

Part a

We know that $U_1, U_2 \sim Unif(0,1)$ and $X = \sqrt{-2log(U_1)}cos(2\pi U_2)$ and $Y = \sqrt{-2log(U_1)}sin(2\pi U_1)$. We will transform U_1 and U_2 using the above functions and show that it yields a normal.

$$\begin{aligned}
X = \sqrt{-2log(U_1)}cos(2\pi U_2) &\text{ and } Y = \sqrt{-2log(U_1)}sin(2\pi U_1) \\
X^2 + Y^2 = -2log(U_1) &\longrightarrow U_1 = exp\{\frac{-1}{2}(X^2+Y^2)\} \\
\frac{Y}{X} = tan(2\pi U_2) &\longrightarrow U_2 = \frac{1}{2\pi}tan^{-1}(\frac{Y}{X}) \\
|J| = |det\begin{bmatrix} \frac{\partial U_1}{\partial X} & \frac{\partial U_1}{\partial Y} \\ \frac{\partial U_2}{\partial X} & \frac{\partial U_2}{\partial Y} \end{bmatrix}| &= |\begin{bmatrix} exp\{\frac{-1}{2}(X^2+Y^2)\}(-X) & exp\{\frac{-1}{2}(X^2+Y^2)\}(-Y) \\ \frac{1}{2\pi}\frac{X^2}{X^2+Y^2}\frac{-Y}{X^2} & \frac{1}{2\pi}\frac{X^2}{X^2+Y^2}\frac{1}{X} \end{bmatrix}| \\
&= \frac{1}{2\pi}\frac{X^2}{X^2+Y^2}exp\{-\frac{X^2+Y^2}{2}\}(1+\frac{Y^2}{X^2}) \\
&= \frac{1}{2\pi}exp\{-\frac{X^2+Y^2}{2}\} \\
&= \frac{1}{\sqrt{2\pi}}exp\{-X^2/2\}\frac{1}{\sqrt{2\pi}}exp\{-Y^2/2\}
\end{aligned}$$

Thus we have that $f_{X,Y}(x,y) = \frac{1}{\sqrt{2\pi}}exp\{-x^2/2\}\frac{1}{\sqrt{2\pi}}exp\{-y^2/2\}$ thus we have two independent standard normal variables.

Part b

Problem 4

We sampled from a standard normal tail as follows: we generated 1,000,000 iid samples from $U_1 \sim Unif(0,1)$ and 1,000,000 iid samples from $U_2 \sim Unif(0,1)$. Then we accept if $U_2\sqrt{d^2 - 2log(U_1)} \leq d$. We tried d from 1 to 20. We repeated this experiment by generating 1,000,000 iid samples from $X \sim N(0,1)$ and we accepted if $X \leq d$.

We found that when we sample directly from standard normal we accept 100% of the time when $d > 4$. When we sampled using the uniforms as described above we found that as d increases the acceptance probability increases but even with $d = 20$ we did not have a 100% acceptance rate. With 1,000,000 samples we still rejected about 2200 times.

Problem 5

Part a

The following is a basic outline of our algorithm. (A, R, C) are the same triples as described in the prompt. Note that we index the matrix and the vector starting from $index = 0$. If we want y_j, the jth element of $y = Wx$. We must find the jth row of our matrix W and multiply with the correct element in vector x:

1. Iterate through the columns (elements in C).
2. For each column i, find the rows of non-zero elements.
3. If the element in row j is non-zero we multiply it with the ith element in x.

We include the following pseudo code:

```
sum = 0;
r = correct row;
for(int i = 0; i < c.length - 1; i++) {
  for(int j = C[i]; j < C[i + 1]; j++) {
    if(R[j] == r) sum = sum + x[i]*A[j];
  }
}
return(sum)
```

Part b

Our strategy is similar to our method in part a. However, with a symmetric matrix we will only store the lower-triangular part of the matrix. A stores the elements in column-wise order; but only the elements in the diagonal and lower-triangular part of the matrix. R stores the corresponding row of the elements in A. And elements in C indicate the element in A that starts the columns of the lower triangular part of our matrix.

We follow the same outline as above, the key difference is that when we check an element we also check it's reflection: when we check if element (i,j), we will also check element (j,i). We include the following pseudo code:

```
sum = 0;
r = correct row;
for(int i = 0; i < c.length - 1; i++) {
  col = i;
  for(int j = c[i]; j < c[i + 1]; j++) {
    row = R[j];
    if(row == r) sum = sum + y[i]*A[j];
    if(col != row && col == r) sum = sum + y[row]*A[j];
  }
}
```

Problem 6

Part a

Part b

Part c

Problem 7

See readme.txt file in /class/stat580/ashum/homework1/

Part a

Code available on impact2.stat.iastate.edu under /class/stat580/ashum/homework1/temp.c

Part b

Code available on impact2.stat.iastate.edu under /class/stat580/ashum/homework1/mult.c