

# STAT 580 Homework 2

Yifan Zhu

February 12, 2017

```
1. #include <stdio.h>
#include <math.h>

#define P0 0.01 /*lower limit of the probability (p) */
#define P1 0.5 /* upper limit of the probability (p) */
#define PLEN 10 /* number of columns */
#define N 5 /* number of experiments (n) */

int factorial(int n)
{
    if (n <= 1)
        return 1;
    else
        return (n * factorial(n - 1));
}

double binopmf(int n, int x, double p)
{
    return (factorial(n) / (factorial(n - x) * factorial(x))) * pow(p, x)
        * pow(1 - p, n - x);
}

int main()
{
    printf("x\\p\\t");
    double step = (P1 - P0) / (double) (PLEN - 1);
    for (int i = 1; i <= PLEN - 1; i++)
    {
        printf("%.4f\\t", P0 + (i - 1)*step);
    }
    printf("%.4f\\n", P0 + (PLEN - 1)*step);

    for (int x = 0; x <= N; x++)
    {
        printf("%d\\t", x);
        for (int i = 1; i <= PLEN - 1; i++)
        {
            printf("%.4f\\t", binopmf(N, x, P0 + (i - 1)*step));
        }
        printf("%.4f\\n", binopmf(N, x, P0 + (PLEN - 1)*step));
    }
    return 0;
}
```

2. (a)  $\int_1^{10} f(x) dx = 1 \Rightarrow c \int_1^{10} \frac{1}{x} dx = c(\log 10 - 0) = 1 \Rightarrow \frac{1}{\log 10}$ . Hence

$$F(x) = \int_1^x f(x) dx = \int_1^x \frac{1}{\log 10} \frac{1}{x} dx = \frac{\log x}{\log 10} = \log_{10}(x), 1 < x < 10$$

Hence the inverse function

$$F^{-1}(u) = 10^u, 0 < u < 1$$

Thus the algorithm would be

---

**Algorithm 1** Sampling  $X$  with cdf  $F(x) = \log_{10}(x)$

---

- 1: Generate  $U \sim \text{Unif}(0, 1)$ ;
  - 2: Set  $X = 10^U$ ;
- 

(b)

```
#include <stdio.h>
#include <time.h>
#define MATHLIB_STANDALONE
#include <Rmath.h>

int main()
{
    double u, x;

    set_seed(time(NULL), 580580); /* set seed */

    u = unif_rand();
    x = pow(10.0, u);
    printf("%f\n", x);

    return 0;
}
```

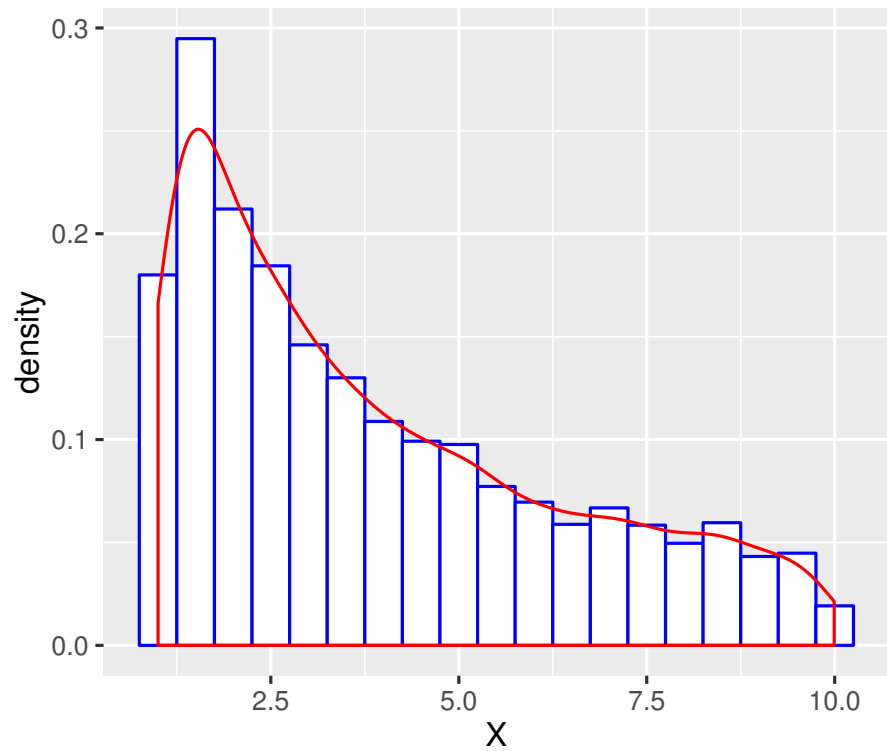
(c)

```
require(ggplot2)

U <- runif(5000, min = 0, max = 1);
X <- 10^U;

d <- data.frame(x = X);

ggplot(d, aes(x = X)) + geom_histogram(aes(y = ..density..),
    binwidth = 0.5, color = "blue", fill = "white") + geom_density(
    alpha = .2, colour = "red", fill = "#FF6666")
```



3. (a)

```
require(ggplot2)

sample1 <- function (n)
{
  size <- 0;
  O <- rep(0,5000);
  while (size < 5000)
  {
    U <- runif(1);
    X <- rexp(1);
    if (U <= 1/(1 + X^2))
    {
      size <- size + 1;
      O[size] <- X;
    }
  }

  return(O)
}

sample2 <- function (n)
{
  size <- 0;
  O <- rep(0,5000);
  while (size < 5000)
  {
    U <- runif(1);
    X <- abs(rcauchy(1));
  }
}
```

```

    if (U <= exp(-X))
    {
      size <- size + 1;
      O[size] <- X;
    }
  }

  return(O)
}

d1 <- data.frame(x = sample1(5000));
ggplot(d1, aes(x = x)) + geom_histogram(aes(y = ..density..),
  binwidth = 0.1, color = "blue", fill = "white") + geom_density(
  alpha = .2, colour = "red", fill = "#FF6666") + xlim(0,5)

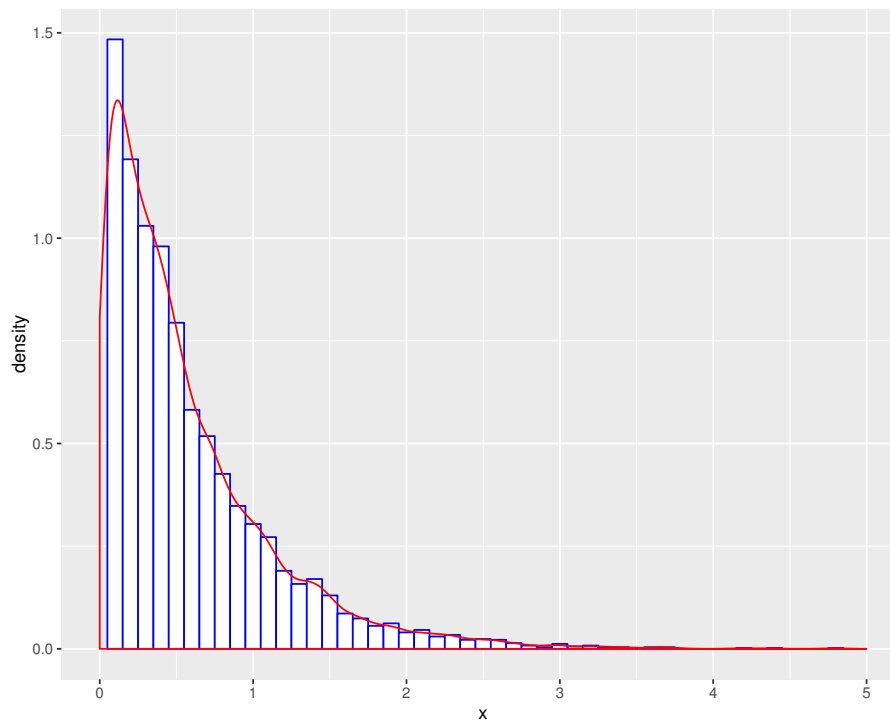
d2 <- data.frame(x = sample2(5000));
ggplot(d2, aes(x = x)) + geom_histogram(aes(y = ..density..),
  binwidth = 0.1, color = "blue", fill = "white") + geom_density(
  alpha = .2, colour = "red", fill = "#FF6666") + xlim(0,5)

system.time(sample1(5000))

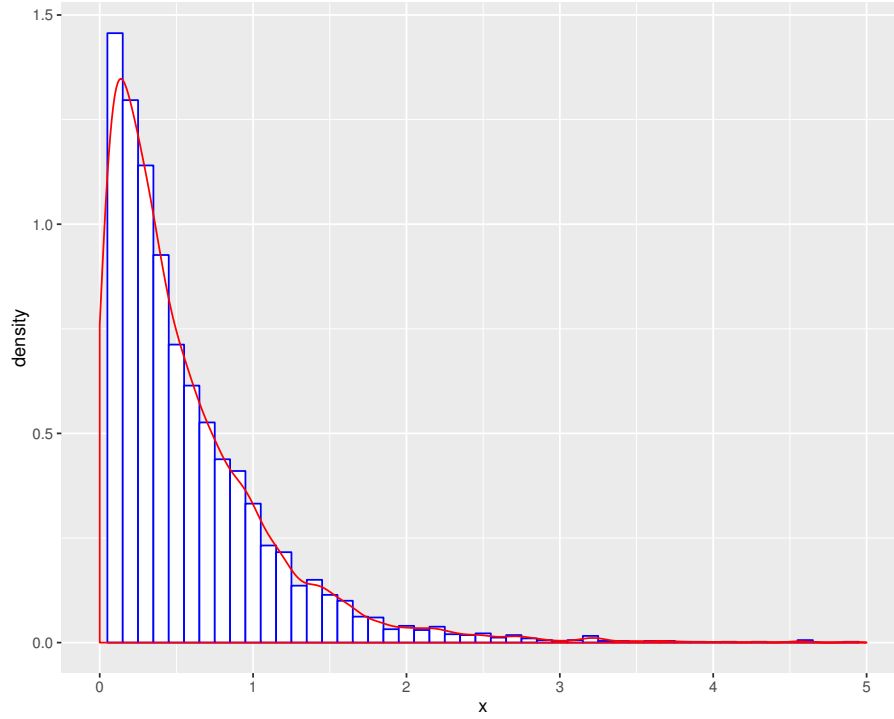
system.time(sample2(5000))

```

The density when using  $g_1(x) = e^{-x}$ :



The density when using  $g_2(x) = \frac{\pi(1+x^2)}{2}$ :



- (b) The density we get using  $g_1$  and  $g_2$  are almost the same. And the time using  $g_2$  is longer than the time using  $g_1$ .

The `system.time(sample1(5000))` returns the time when using  $g_1$ :

```
user  system elapsed
0.04   0.00   0.04
```

The `system.time(sample2(5000))` returns the time when using  $g_2$ :

```
user  system elapsed
0.07   0.00   0.08
```

4.

---

**Algorithm 2** Sampling from  $f(x, y) \propto x^\alpha y$  with support being  $x^2 + y^2 \leq 1$  in the first quadrant

---

- 1: Generate  $X \sim \text{Beta}(\alpha + 1, 1)$  and  $Y \sim \text{Beta}(2, 1)$  independently;
  - 2: **if**  $X^2 + Y^2 > 1$  **then**  
     Go to Step 1;
  - 3: **else**  
     Return  $(X, Y)$ ;
  - 4: **end if**
-