Stat 361 - Recitation 4

Generating Random Variates

Orçun Oltulu

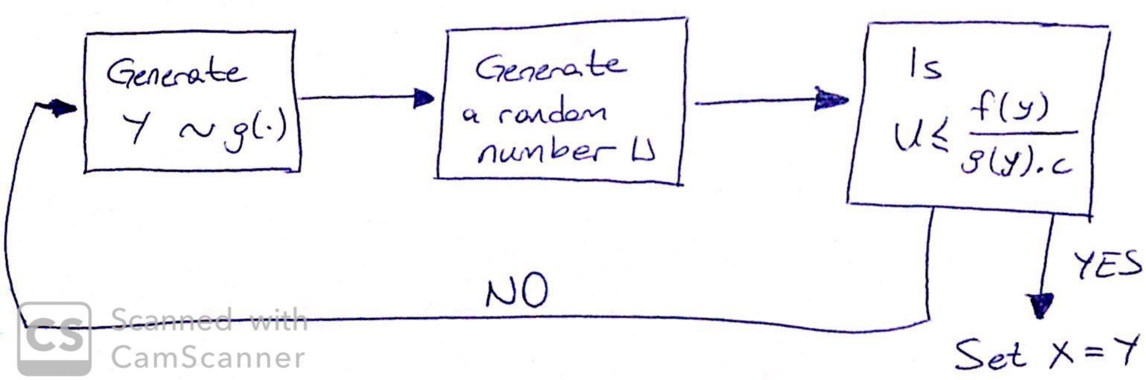
27-28 / 02 / 2020

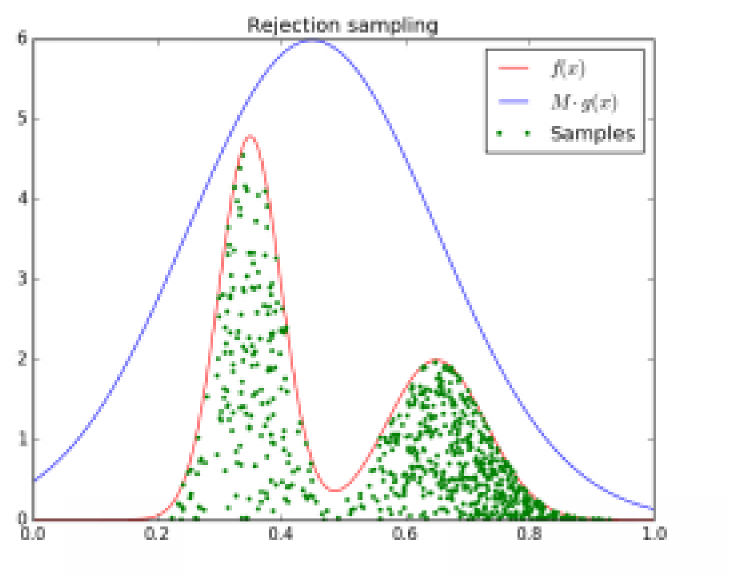
Accept - Reject Method (Continuous Case):

The basic idea is to find an alternative probability distribution G, with density function g(x), from which we already have an efficient algorithm for generating from (e.g., inverse transform method or whatever), but also such that the function g(x) is "close" to f(x). In particular, we assume that the ratio $\frac{f(x)}{g(x)}$ is bounded by a constant c > 0; $\sup_x \frac{f(x)}{g(x)} \leq c$. (And in practice we would want c as close to 1 as possible.)

Here then is the algorithm for generating X distributed as F:

1. Choose a density g(y) that is easy to sample from.
2. Find a constant c such that $\frac{f(y)}{g(y)} \leq c$.
3. Generate a random number Y from the density g(y).
4. Generate a uniform random number U.
5. If $U \leq \frac{f(y)}{cg(y)}$ then accept, else go to step 3.





Question 1:

Generate a random sample of size n = 10000 having density function

$$f(x) = 20x(1-x)^3, \quad 0 < x < 1$$

$$\text{Choose } \ g(x) = 1, \quad 0 \leq x \leq 1$$

To determine the constant 'c' such that $\frac{f(x)}{g(x)} \leq c$, find the maximum value of $\frac{f(x)}{g(x)} = 20x(1-x)^3$

Take first derivative and set this equals to 0.

$$\frac{d}{dx}[\frac{f(x)}{g(x)}] = 20[(1-x)^3 - 3x(1-x)^2]$$

From here you will get x as 0.25 and thus,

$$\frac{f(x)}{g(x)} \leq 20(\frac{1}{4})(\frac{3}{4})^3 = \frac{135}{64} = c$$

Hence,

$$\frac{f(x)}{cg(x)} = \frac{256}{27}x(1-x)^3$$

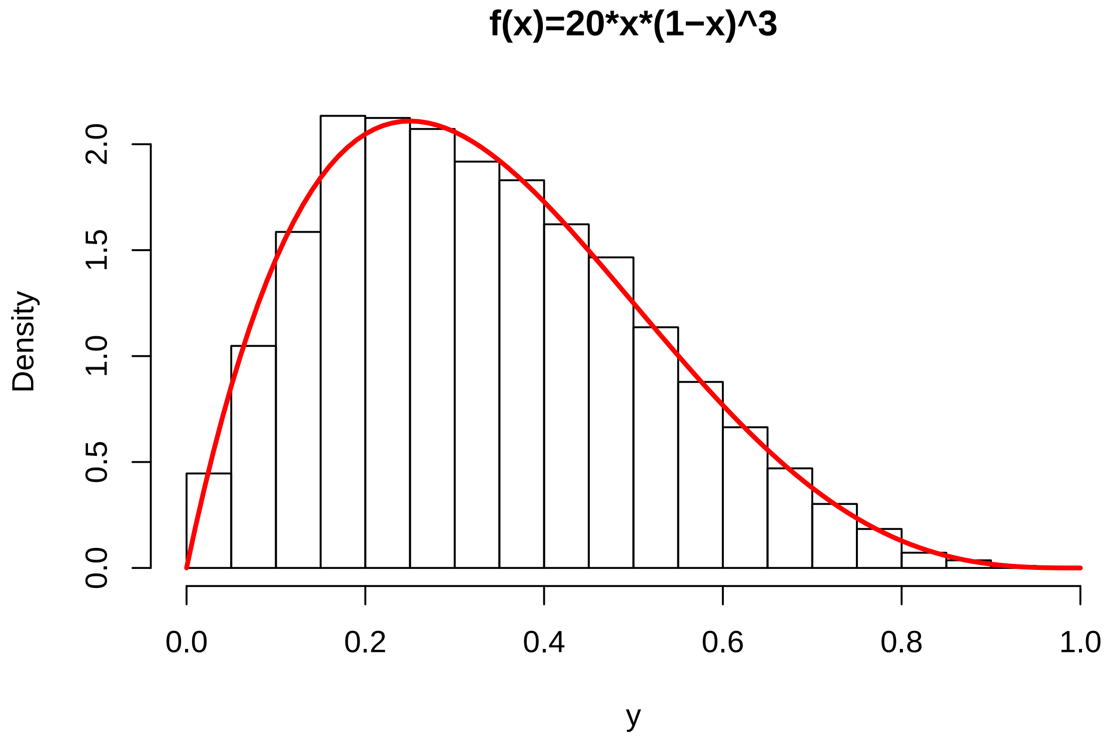
Step 1: Generate random numbers U_1 and U_2 from Uniform(0,1).

Step 2: If $U_2 \leq \frac{256}{27} U_1 (1 - U_1)^3$, stop and set $X = U_1$. Otherwise return to step 1.

```r
n <- 10^4
i <- 0 # iteration
k <- 0 # counter for accepted
y <- numeric(n) # random sample

set.seed(1234)

while (k < n) {
  u <- runif(1)
  i <- i + 1
  x <- runif(1)

    if (256 / 27 * x * (1-x)^3 > u) {
      k <- k + 1
      y[k] <- x
    }
}

i;k;length(y)
```

```
## [1] 21297

## [1] 10000

## [1] 10000
```

```r
hist(y, prob = TRUE, main ="f(x)=20*x*(1-x)^3")
a = seq(0,1,0.01)
lines(a,20*a*(1-a)^3,col="red",lwd = 2.5)
```




```
summary(y)
```

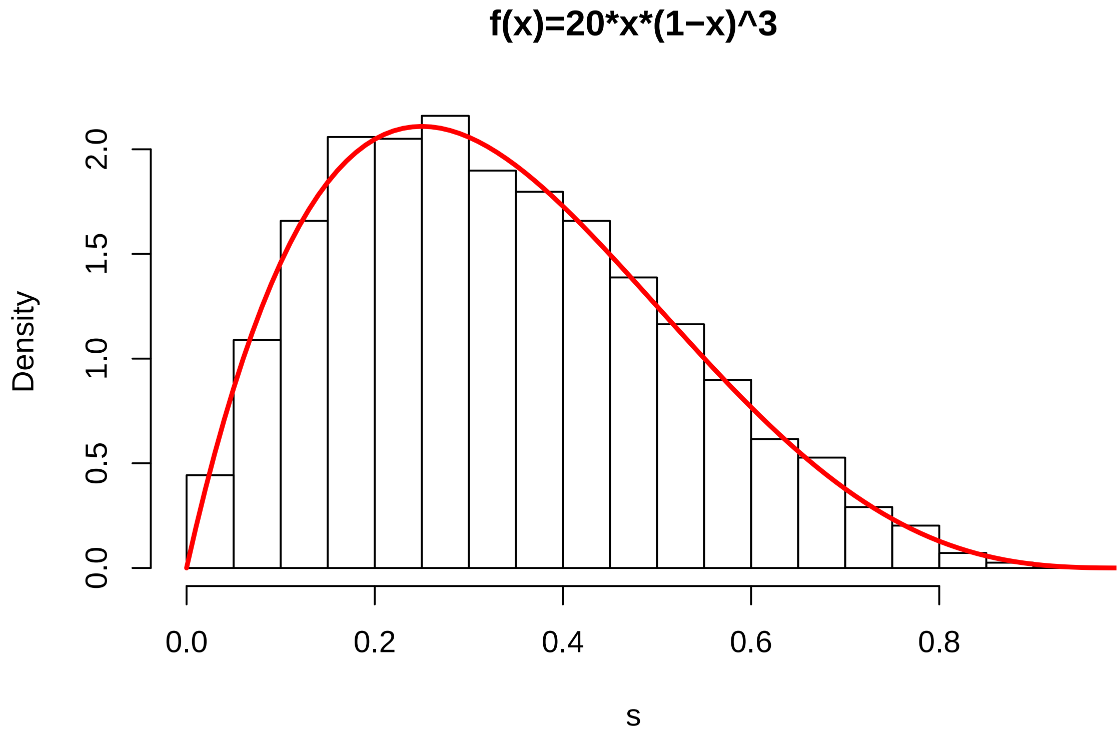
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.003268 0.195170 0.314911 0.335047 0.457559 0.955364
```

Alternatively;

```
set.seed(1234)
n = 10000
x2 = runif(n)
y2 = runif(n)

# acceptance condition
s = x2[y2 < 256 / 27 * x2 * (1-x2)^3]

length(s)
```

```
## [1] 4741
```

```
hist(s, prob = TRUE, main ="f(x)=20*x*(1-x)^3")
a = seq(0,1,0.01)
lines(a,20*a*(1-a)^3,col="red",lwd = 2.5)
```

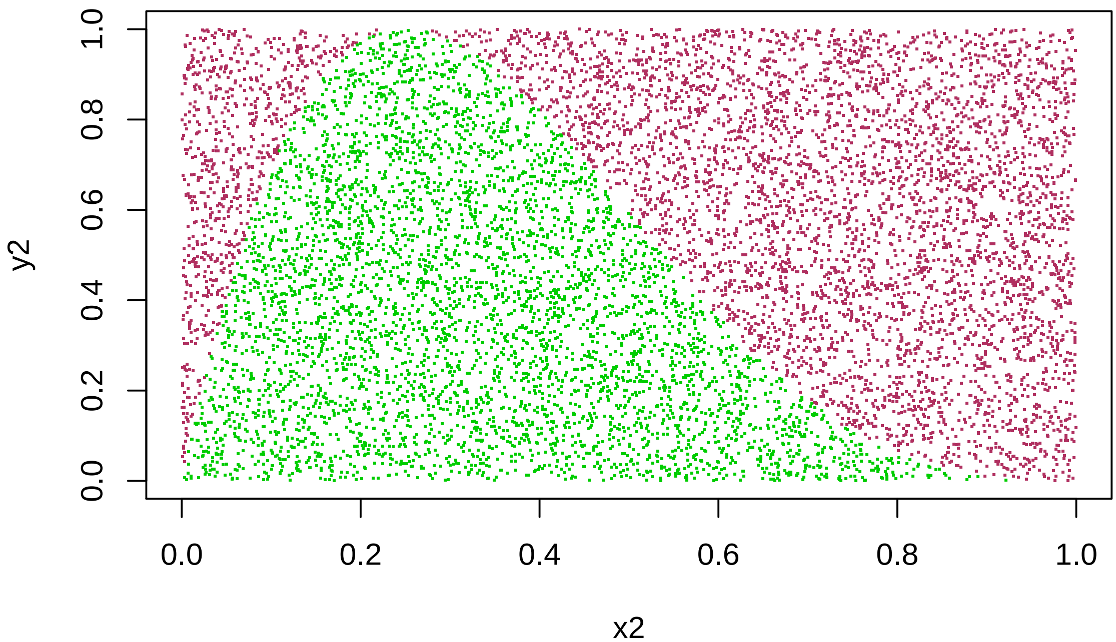



```
summary(y2)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## 0.0004263 0.2586527 0.5030983 0.5045714 0.7482903 0.9999746
```

Alternative visual;

```
plot(x2,y2, pch=".", col="maroon")
points(s,y2[y2 < 256 / 27 * x2 * (1-x2)^3], pch=".", col="green3")
```



Question 2:

Generate a random sample of size n = 10000 having density function

$$f(x) = 3x^2, \quad 0 \le x \le 1$$

$$\text{Choose } \; g(x) = 1, \quad 0 \le x \le 1$$

$$\max[\frac{f(x)}{g(x)}] = 3 = c$$

Hence,

$$\frac{f(x)}{cg(x)} = x^2$$

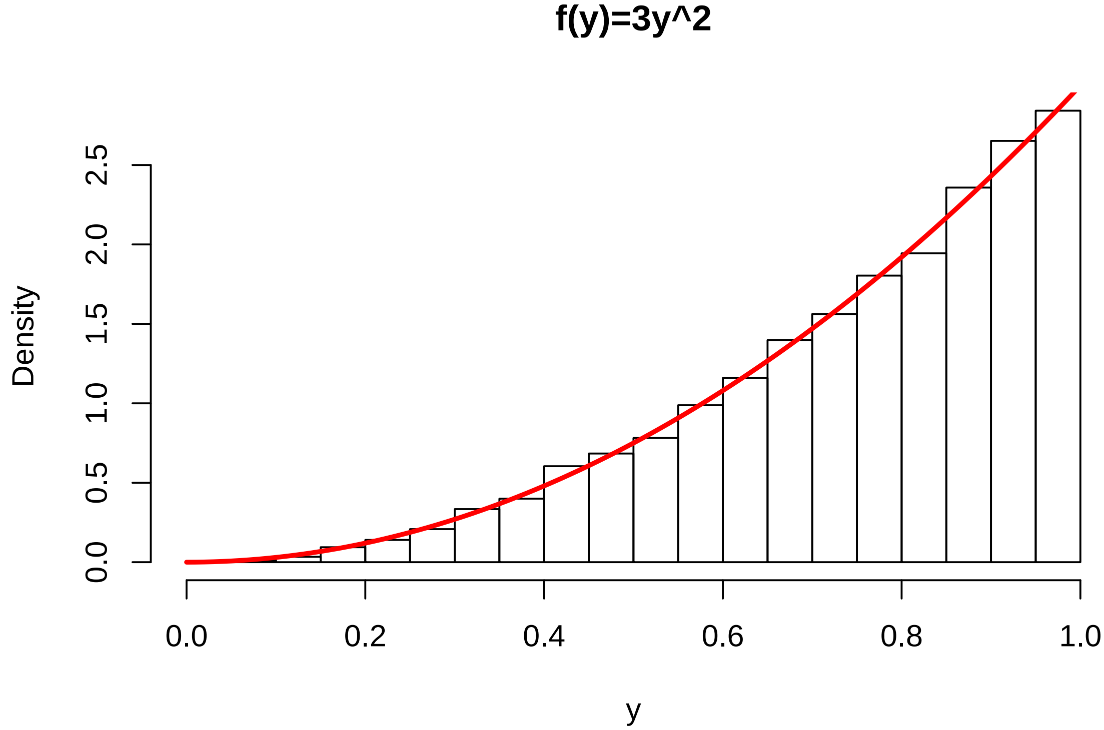
Step 1: Generate random numbers U_1 and U_2 from Uniform(0,1).

Step 2: If $U_2 \le U_1^2$, stop and set $X = U_1$. Otherwise return to step 1.

```r
n <- 10^4
i <- 0 # iteration
k <- 0 # counter for accepted
y <- numeric(n) # random sample

set.seed(1234)
while(TRUE){
  u <- runif(1)
  i <- i + 1
  x <- runif(1)

    if(u < x^2){
      k <- k + 1
      y[k] <- x
    }

    if(k >= n){
      break
    }
}

i;k;length(y)
```

```
## [1] 30060

## [1] 10000

## [1] 10000
```

```r
hist(y, prob = TRUE, main ="f(y)=3y^2")
a = seq(0,1,0.01)
lines(a,3*a^2,col="red",lwd = 2.5)
```




```r
summary(y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.03529 0.63172 0.79380 0.75167 0.91065 0.99998
```

Accept - Reject Method (Discrete Case):

Now, suppose we want to generate an X that is a discrete random variable with probability mass function(PMF) $p(k) = P(X = k)$. Further suppose that we can easily generate a discrete random variable Y with PMF $q(k) = P(Y = k)$ such that $\frac{p(k)}{q(k)} \leq c$.

Step 1: Generate a random variable Y which is distributed as q(k).

Step 2: Generate U (independent from Y).

Step 3: If $U \leq \frac{p(Y)}{cq(Y)}$, then set X = Y; otherwise go back to step 1.

Question 3:

Simulate a discrete random variable X that has PMF given by

$$P(X = 1) = 0.23$$
$$P(X = 2) = 0.32$$
$$P(X = 3) = 0.17$$
$$P(X = 4) = 0.28$$

Since the domain of X is $1, 2, 3, 4$ we can assume that random variable Y has discrete uniform distribution between (1,4). The PMF of discrete uniform distribution is

$$q(k) = P(Y = k) = \frac{1}{4}, \quad k = 1, 2, 3, 4$$

Therefore,

Since the biggest pmf for X is 0.32, we can choose minimum c by the equation:

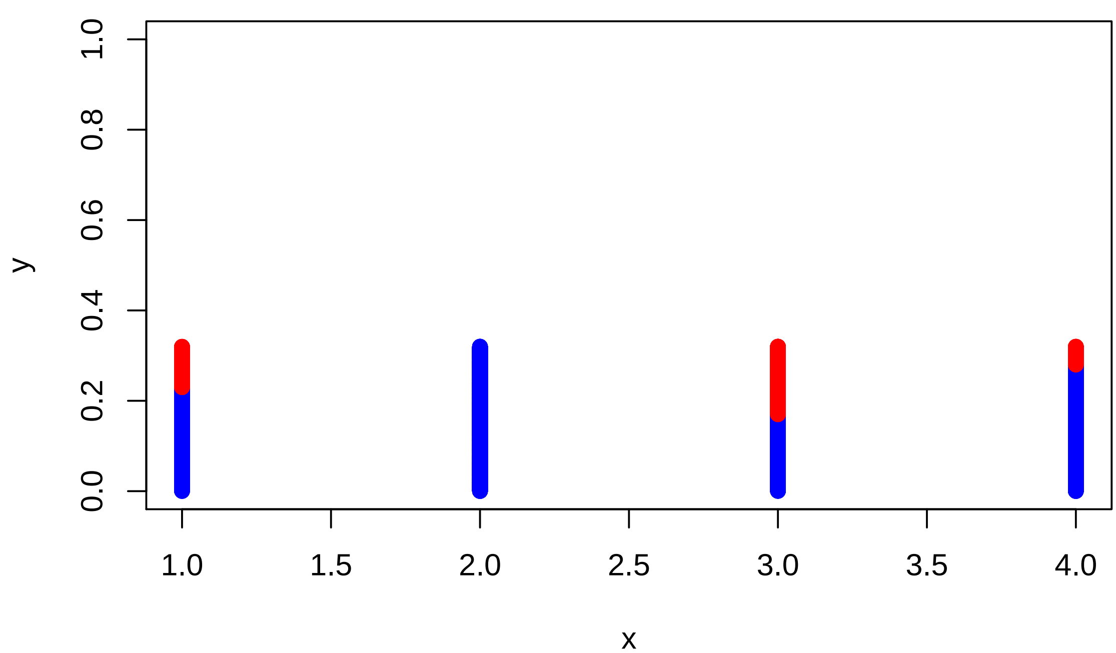
$$\frac{\max(p_x)}{0.25} = \frac{0.32}{0.25} = c = 1.28$$

```
n <- 10000
probs <- c(0.23, 0.32, 0.17, 0.28) # probabilities for desired density
constant <- 1.28

cq <- constant * 0.25
cq
```

```
## [1] 0.32
```

```
# for accepted variates
x <- numeric(n)
y <- numeric(n)
# for rejected variates
rej_x <- numeric(n)
rej_y <- numeric(n)

i <- 1 # initialization of index for accepted r.v's
j <- 1 # initialization of index for rejected r.v's

set.seed(1234)
while(i <= n){

  k <- sample(1:4, size = 1) # random number for q(k)
```

```r
  u <- runif(1) # random number for comparison

  if (u <= probs[k]/0.32){

    x[i] <- k
    y[i] <- u * 0.32
    i <- i + 1

  }else{

    rej_x[j] <- k
    rej_y[j] <- u * 0.32
    j <- j + 1

  }
}

plot(x, y, type="p", col="blue",ylim=c(0,1))
points(rej_x, rej_y,col="red")
```



```r
table(x)/n
```

```
## x
##      1      2      3      4
## 0.2322 0.3237 0.1675 0.2766
```