3.3 The Acceptance-Rejection Method

Suppose that X and Y are random variables with density or pmf f and g, respectively, and there exists a constant c such that

$$\frac{f(t)}{g(t)} \leq c$$

for all t such that $f(t) > 0$. Then the acceptance-rejection method (or rejection method) can be applied to generate the random variable X.

The Acceptance-Rejection Method

Obter realizações da v.a. X:

1. Find a random variable Y with density g satisfying $f(t)/g(t) \leq c$, for all t such that $f(t) > 0$. Provide a method to generate random Y.

2. For each random variate required:

   (a) Generate a random y from the distribution with density g.

   (b) Generate a random u from the Uniform(0, 1) distribution.

   (c) If $u < f(y)/(cg(y))$, accept y and deliver $x = y$; otherwise reject y and repeat from Step 2a.

on average each sample value of X requires c iterations.

For efficiency, Y should be easy to simulate and c small.

Example 3.7 (Acceptance-rejection method). This example illustrates the acceptance-rejection method for the beta distribution. On average, how many random numbers must be simulated to generate 1000 variates from the Beta($\alpha = 2$, $\beta = 2$) distribution by this method? It depends on the upper bound c of $f(x)/g(x)$, which depends on the choice of the function $g(x)$.

The Beta(2,2) density is $f(x) = 6x(1-x)$, $0 < x < 1$. Let $g(x)$ be the Uniform(0,1) density. Then $f(x)/g(x) \leq 6$ for all $0 < x < 1$, so $c = 6$. A random x from $g(x)$ is accepted if

$$\frac{f(x)}{cg(x)} = \frac{6x(1-x)}{6(1)} = x(1-x) > u.$$

On average, $cn = 6000$ iterations (12000 random numbers) will be required for a sample size 1000.

```
n <- 1000
k <- 0      #counter for accepted
j <- 0      #iterations
y <- numeric(n)

while (k < n) {
    u <- runif(1)
    j <- j + 1
    x <- runif(1)  #random variate from g
    if (x * (1-x) > u) {
        #we accept x
        k <- k + 1
        y[k] <- x
    }
}
```

3.4 Transformation Methods

Many types of transformations other than the probability inverse transformation can be applied to simulate random variables. Some examples are

1. If $Z \sim \text{N}(0,1)$, then $V = Z^2 \sim \chi^2(1)$.

2. If $U \sim \chi^2(m)$ and $V \sim \chi^2(n)$ are independent, then $F = \frac{U/m}{V/n}$ has the F distribution with (m, n) degrees of freedom.

3. If $Z \sim \text{N}(0,1)$ and $V \sim \chi^2(n)$ are independent, then $T = \frac{Z}{\sqrt{V/n}}$ has the Student t distribution with n degrees of freedom.

4. If $U, V \sim \text{Unif}(0,1)$ are independent, then

$$Z_1 = \sqrt{-2\log U}\ \cos(2\pi V),$$
$$Z_2 = \sqrt{-2\log U}\ \sin(2\pi V)$$

are independent standard normal variables [255, p. 86].

5. If $U \sim \text{Gamma}(r, \lambda)$ and $V \sim \text{Gamma}(s, \lambda)$ are independent, then $X = \frac{U}{U+V}$ has the Beta(r, s) distribution.

6. If $U, V \sim$ Unif(0,1) are independent, then

$$X = \left\lfloor 1 + \frac{\log(V)}{\log(1 - (1 - \theta)^U)} \right\rfloor$$

has the Logarithmic(θ) distribution, where $\lfloor x \rfloor$ denotes the integer part of x.

Example 3.8 (Beta distribution). The following relation between beta and gamma distributions provides another beta generator.

If $U \sim \text{Gamma}(r, \lambda)$ and $V \sim \text{Gamma}(s, \lambda)$ are independent, then

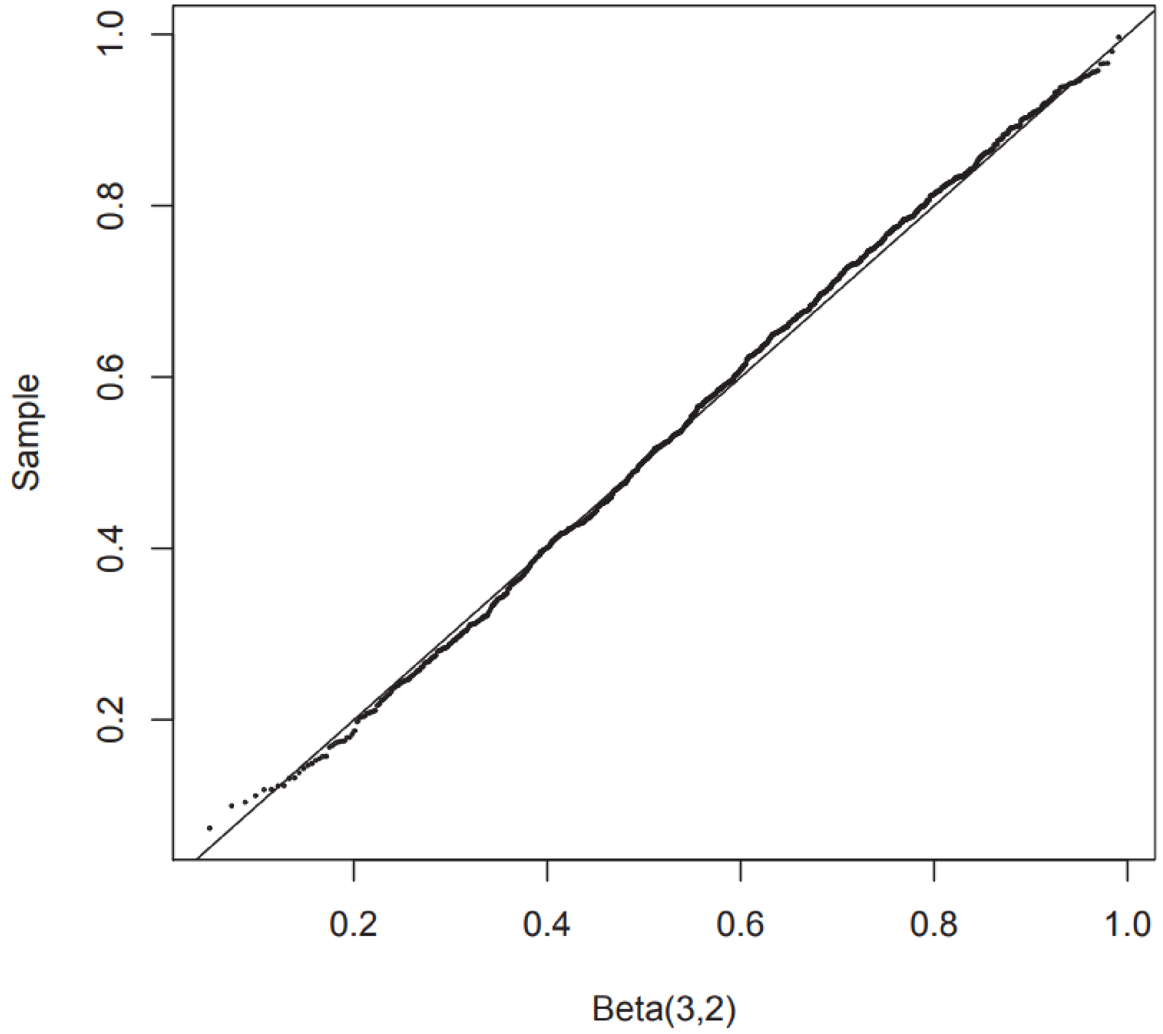
$$X = \frac{U}{U + V}$$

has the $\text{Beta}(r, s)$ distribution [255, p.64]. This transformation determines an algorithm for generating random $\text{Beta}(a, b)$ variates.

1. Generate a random u from $\text{Gamma}(a, 1)$.
2. Generate a random v from $\text{Gamma}(b, 1)$.
3. Deliver $x = \frac{u}{u+v}$.

```
n <- 1000
a <- 3
b <- 2
u <- rgamma(n, shape=a, rate=1)
v <- rgamma(n, shape=b, rate=1)
x <- u / (u + v)
```

```
q <- qbeta(ppoints(n), a, b)
qqplot(q, x, cex=0.25, xlab="Beta(3, 2)", ylab="Sample")
abline(0, 1)
```



3.5 Sums and Mixtures

Sums and mixtures of random variables are special types of transformations. In this section we focus on sums of independent random variables (convolutions) and several examples of discrete and continuous mixtures.

Convolutions

Let $X_1, \ldots, X_n$ be independent and identically distributed with distribution $X_j \sim X$, and let $S = X_1 + \cdots + X_n$. The distribution function of the sum S is called the n-fold convolution of X and denoted $F_X^{*(n)}$. It is straightforward to simulate a convolution by directly generating $X_1, \ldots, X_n$ and computing the sum.

Example 3.10 (Chisquare). This example generates a chisquare $\chi^2(\nu)$ random variable as the convolution of ν squared normals. If $Z_1, \ldots, Z_\nu$ are iid N(0,1) random variables, then $V = Z_1^2 + \cdots + Z_\nu^2$ has the $\chi^2(\nu)$ distribution. Steps to generate a random sample of size n from $\chi^2(\nu)$ are as follows:

1. Fill an $n \times \nu$ matrix with $n\nu$ random N(0,1) variates.
2. Square each entry in the matrix (1).
3. Compute the row sums of the squared normals. Each row sum is one random observation from the $\chi^2(\nu)$ distribution.
4. Deliver the vector of row sums.

An example with $n = 1000$ and $\nu = 2$ is shown below.

```
n <- 1000
nu <- 2
X <- matrix(rnorm(n*nu), n, nu)^2 #matrix of sq. normals
#sum the squared normals across each row: method 1
y <- rowSums(X)
#method 2
y <- apply(X, MARGIN=1, FUN=sum)  #a vector length n
```

Mixtures

A random variable X is a discrete mixture if the distribution of X is a weighted sum $F_X(x) = \sum \theta_i F_{X_i}(x)$ for some sequence of random variables $X_1, X_2, \ldots$ and $\theta_i > 0$ such that $\sum_i \theta_i = 1$. The constants θ_i are called the mixing weights or mixing probabilities. Although the notation is similar for sums and mixtures, the distributions represented are different.

A random variable X is a continuous mixture if the distribution of X is $F_X(x) = \int_{-\infty}^{\infty} F_{X|Y=y}(x) f_Y(y)\, dy$ for a family $X|Y = y$ indexed by the real numbers y and weighting function f_Y such that $\int_{-\infty}^{\infty} f_Y(y)\, dy = 1$.

Example 3.11 (Convolutions and mixtures). Let $X_1 \sim \text{Gamma}(2, 2)$ and $X_2 \sim \text{Gamma}(2, 4)$ be independent. Compare the histograms of the samples generated by the convolution $S = X_1 + X_2$ and the mixture $F_X = 0.5F_{X_1} + 0.5F_{X_2}$.

```
n <- 1000
x1 <- rgamma(n, 2, 2)
x2 <- rgamma(n, 2, 4)
s <- x1 + x2                 #the convolution
u <- runif(n)
k <- as.integer(u > 0.5)   #vector of 0's and 1's
x <- k * x1 + (1-k) * x2   #the mixture
```

```
par(mfcol=c(1,2))           #two graphs per page
hist(s, prob=TRUE, xlim=c(0,5), ylim=c(0,1))
hist(x, prob=TRUE, xlim=c(0,5), ylim=c(0,1))
par(mfcol=c(1,1))           #restore display
```