Numerical and Statistical Methods for Finance

Generating Continuous Random Variables

Pierpaolo De Blasi

University of Torino
email: pierpaolo.deblasi@unito.it
webpage: sites.google.com/a/carloalberto.org/pdeblasi/

Lecture no. 8
20 November 2013

Rejection Method

Suppose we have a method for simulating from a random variable having density function $g(x)$. We can use this as a basis for generating from a continuous rv with density $f(x)$.

We first generate a rv Y from $g(x)$, then accept this generated value with a probability proportional to $f(Y)/g(Y)$. Let $c > 0$ such that

$$f(x)/g(x) \leq c \quad \text{for all } x$$

The rejection algorithm is as follows:

Step 1: Generate Y having density $g(x)$.

Step 2: Generate $U \sim \text{Unif}(0, 1)$.

Step 3: If $U < f(Y)/cg(Y)$, set $X = Y$ and stop. Otherwise return to Step 1.

The number of iteration of the algorithm needed to obtain X is a geometric random variable with mean c (i.e. $\Pr(\text{Acceptance}) = 1/c$).

Beta with Uniform Envelop

Example 5d page 72, Ross (2006)

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

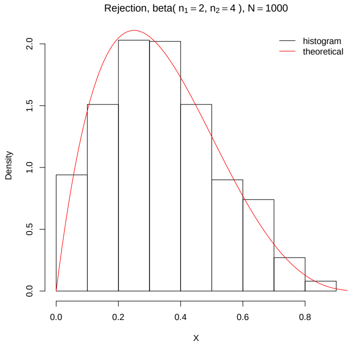
i.e. the density of beta$(2,4)$. We use as proposal density $g(x) = 1$ for $x \in (0,1)$, i.e. the density of Unif$(0,1)$.

To find c, we search for the maximum of $f(x)/g(x) = 20x(1-x)^3$ over $(0,1)$. Using first calculus one finds $\arg\max_x f(x)/g(x) = 1/4$ (*use first order calculus*), hence

$$c = 20(1/4)(1-1/4)^3 = 135/64 \approx 2.11$$

Noting that $f(x)/cg(x) = (256/27)x(1-x)^3$, the algorithm is as follows:

```
prob.accept<-0
iter<-0
U<-1
while (U>=prob.accept){
    Y<-runif(1)
    prob.accept<-(256/27)*Y*(1-Y)^3
    U<-runif(1)
    iter<-iter+1
}
X<-Y
mean(num.iter)
[1] 2.041
```



Gamma with Exponential Envelop

Example 5e page 73, Ross (2006)

Suppose we want to generate $X \sim \text{gamma}(\frac{3}{2}, 1)$, i.e. from the density

$$f(x) = \frac{1}{\Gamma(3/2)} x^{1/2} \mathrm{e}^{-x}, \quad x \geq 0$$

note that $\Gamma(3/2) = \sqrt{\pi}/2$ (*why?*).

As proposal distribution we use $Y \sim \text{Exp}(\lambda)$ with λ chosen so that $E(Y) = E(X)$, that is $\lambda = 2/3$ (*why?*). Hence

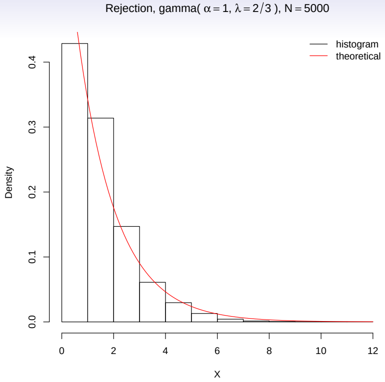
$$g(x) = \frac{2}{3} \mathrm{e}^{-\frac{2}{3}x}$$

In order to set $c = \max_x f(x)/g(x)$, note that $x^{1/2}\mathrm{e}^{-x/3}$ is maximized for $x = 3/2$ (*use first order calculus*), hence

$$c = f(3/2)/g(3/2) = \cdots = \frac{3^{3/2}}{(2\pi \mathrm{e})^{1/2}} \approx 1.2573$$

```
c.const<-3^(3/2)/(2*pi*exp(1))^(1/2)

lambda<-2/3
prob.accept<-0
iter<-0
U<-1
while (U>=prob.accept){
    Y<--log(runif(1))/lambda
    prob.accept<-f(Y)/(c.const*g(Y))
    U<-runif(1)
}
X<-Y
mean(num.iter)
[1] 1.2542
```



It can be proved that the most efficient (e.g. minimizing c) Exp(λ) to use as proposal density is the one with mean ($1/\lambda$) equal to the mean of the gamma, see derivation in Ross (2006, pp 74-75).

This is valid only when the shape parameter α of the gamma rv is larger than 1. In fact, when $\alpha < 1$, $f(x)/g(x) \to \infty$ as $x \to \infty$ (*why?*), so no upper bound to $f(x)/g(x)$ exists ($c = \infty$).

Normal with Exponential Envelop

Example 5f page 75, Ross (2006) Da fare favaro

To generate a standard normal random variable $Z \sim \mathrm{N}(0,1)$, note first that $X = |Z|$ has density

$$f(x) = \frac{2}{\sqrt{2\pi}} \mathrm{e}^{-\frac{x^2}{2}}, \quad x \geq 0$$

(*why?*). As proposal density, we use Exp(1) with density $g(x) = \mathrm{e}^{-x}$.

Note that

$$f(x)/g(x) = \sqrt{2/\pi}\, \mathrm{e}^{x-\frac{x^2}{2}}$$

and the maximum value of $\mathrm{e}^{x-\frac{x^2}{2}}$ is attained at the maximum value of $x - x^2/2$, i.e. for $x = 1$ (*use first order calculus*). Hence

$$c = \max_x f(x)/g(x) = f(1)/g(1) = \sqrt{2\mathrm{e}/\pi} \approx 1.32$$

Since $f(x)/(cg(x)) = \exp\{-\frac{(x-1)^2}{2}\}$, it follows that we can generate the absolute value X of a standard normal random variable as follows:

Step 1: Generate $Y \sim \text{Exp}(1)$

Step 2: Generate $U \sim \text{Unif}(0, 1)$

Step 3: If $U < \exp\{-(Y-1)^2/2\}$, set $X = Y$ and stop. Otherwise return to Step 1.

Note that in Step 3, Y is accepted when $-\log U > (Y-1)^2/2$ and $-\log U$ has distribution Exp(1). Hence we can rewrite the algorithm as

Step 1: Generate $Y_1, Y_2 \sim$ iid Exp(1)

Step 2: If $Y_2 > (Y_1 - 1)^2/2$, set $X = Y_1$ and stop. Otherwise return to Step 1.

Further, by exploiting the *lack of memory* property of the exponential distribution,

$$\Pr(Y > t + s | Y > s) = \Pr(Y > t) \tag{1}$$

we know that, when we accept Y_1, i.e. conditional to $Y_2 > (Y_1 - 1)^2/2$, $Y_2 - (Y_1 - 1)^2/2$ is also Exp(1) (and independent of X).

Summing up, we have an algorithm that generate at the same time the absolute value of a standard normal rv and an independent unit rate exponential rv.

Once we have simulated a random variable X distributed as the absolute value of a standard normal rv, we can obtain Z by letting $Z = -X$ with probability $1/2$, $Z = X$ otherwise (use auxiliary Bernoulli(1/2)).

Step 1: Generate $Y_1 \sim \text{Exp}(1)$

Step 2: Generate $Y_2 \sim \text{Exp}(1)$

Step 3: If $Y_2 > (Y_1 - 1)^2/2$, set $Y = Y_2 - (Y_1 - 1)^2/2$ and go to Step 3. Otherwise go to Step 1.

Step 4: Generate $U \sim \text{Unif}(0, 1)$ and set

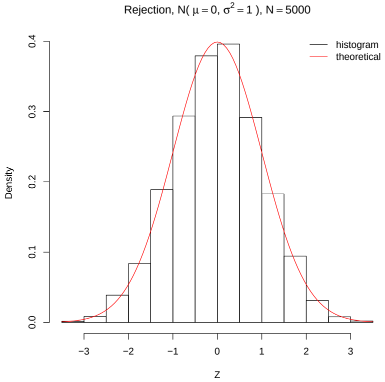
$$Z = \begin{cases} -Y_1 & \text{if } U \leq \frac{1}{2} \\ Y_1 & \text{if } U > \frac{1}{2} \end{cases}$$

If we want to generate a sequence of standard normal rv's, we can use the exponential rv Y obtained at Step 3 in Step 1 for the next normal to be generated.

If a $\text{N}(\mu, \sigma^2)$ is needed, we can take $\mu + \sigma Z$ (*why?*).

```
mu<-0;sigma<-1
Y1<--log(runif(1))
Y2<--log(runif(1))
iter<-1
while (Y2<=(Y1-1)^2/2) {
    Y1<--log(runif(1))
    Y2<--log(runif(1))
    iter<-iter+1
}
U<-runif(1)
if (U<=1/2) Z<--Y1 else Z<-Y1
Y1<-Y2-(Y1-1)^2/2
Z<-mu+sigma*Z

mean(num.iter)
[1] 1.3136
```



Gamma(2,1) Conditioned to Exceed 5

Example 5g page 77, Ross (2006)

Suppose we want to generate a gamma$(2,1)$ rv conditional on its value exceeding 5, that is from the density

$$f(x) = \frac{x\mathrm{e}^{-x}}{\int_5^\infty x\mathrm{e}^{-x}\mathrm{d}x} = \frac{x\mathrm{e}^{-x}}{6\mathrm{e}^{-5}}, \quad x \geq 5$$

where the integral was evaluated by using integration by part (*why?*).

Since a gamma$(2,1)$ has expected value 2, we will use the rejection method based on an exponential with mean 2 (i.e. with rate $1/2$) conditioned to be at least 5. That is

$$g(x) = \frac{\frac{1}{2}\mathrm{e}^{-x/2}}{\mathrm{e}^{-5/2}}, \quad x \geq 5$$

We have

$$f(x)/g(x) = (\mathrm{e}^{5/2}/3)x\mathrm{e}^{-x/2}, \quad x \geq 5$$

Since $x\mathrm{e}^{-x/2}$ is decreasing for $x \geq 5$ (*why?*),

$$c = \max_{x\geq 5} f(x)/g(x) = f(5)/g(5) = 5/3 \approx 1.667$$

In order to generate an exponential conditioned to exceed 5, we resort again to the lack of memory property (1): if $Y \sim \text{Exp}(\lambda)$, then $Y - 5 | Y > 5 \sim \text{Exp}(\lambda)$.

Step 1: Generate $U \sim \text{Unif}(0,1)$

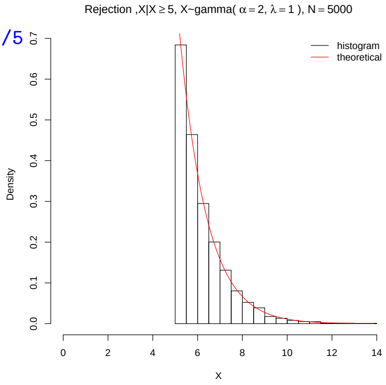
Step 2: Set $Y = 5 - \log(U)/(1/2)$

Step 3: Generate $U \sim \text{Unif}(0,1)$

Step 4: If $U \leq \frac{e^{5/2}}{5} Y e^{-Y/2}$, set $X = Y$. Otherwise return to Step 1.

```
prob.accept<-
    function(x) exp(5/2)*x*exp(-x/2)/5
Y<-5-2*log(runif(1))
iter<-1
U<-runif(1)
p<-prob.accept(Y)
while (U>p) {
    Y<-5-2*log(runif(1))
    p<-prob.accept(Y)
    iter<-iter+1
}
X<-Y

mean(num.iter)
[1] 1.6588
```



Box-Muller Algorithm for Normal rv's

Let X and Y be independent normal rv's and let R and θ denote the polar coordinates of the vector (X, Y):

$$R^2 = X^2 + Y^2, \quad \tan\theta = Y/X$$

Since X and Y are independent, their joint density is

$$f(x, y) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-x^2/2} \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-y^2/2} = \frac{1}{2\pi} \mathrm{e}^{-(x^2+y^2)/2}$$

To determine the joint density of (R^2, θ), call it $f(d, \theta)$, we make the change of variables

$$d = x^2 + y^2, \quad \theta = \tan^{-1}\left(\frac{y}{x}\right)$$

The Jacobian of this transformation (determinant of partial derivatives of d and θ with respect to x and y) is equal to 2 (*why?*). Hence the joint density of (R^2, θ) is given by

$$f(d, \theta) = \frac{1}{2} \frac{1}{2\pi} \mathrm{e}^{-d/2}, \quad 0 < d < \infty, 0 < \theta < 2\pi$$

As this is equal to the product of an exponential rv with rate $1/2$ and the uniform density on $(0, 2\pi)$, it follows that

$$R^2 \perp \theta, \quad R^2 \sim \text{Exp}(1/2), \quad \theta \sim \text{Unif}(0, 2\pi)$$

Step 1: Generate $U_1, U_2 \sim$ iid Unif$(0, 1)$

Step 2: Set $R^2 = -2 \log U_1$ and $\theta = 2\pi U_2$

Step 3: Return

$$X = R\cos\theta = \sqrt{-2\log U_1}\cos(2\pi U_2)$$
$$Y = R\sin\theta = \sqrt{-2\log U_1}\sin(2\pi U_2)$$

Also known as *Box-Muller algorithm*.

Calculating sines and cosines can be expensive (in terms of the time required), but there is a version of the Box and Muller algorithm that avoids this.

Suppose that the point (A, B) is uniformly distributed over the unit circle. Let (S, Ψ) be the polar coordinates of (A, B). It can be shown that $S^2 \sim \text{Unif}(0, 1)$ and $\Psi \sim \text{Unif}(0, 2\pi)$ independently of S (*why?*).

figure here

Thus, $(\sqrt{-2\log(S^2)}, \Psi)$ has the same distribution of the polar coordinates (R, θ) of (X, Y). The advantage of this representation is that we can easily calculate $\cos\Psi$ and $\sin\Psi$:

$$\cos\Psi = A/S, \quad \sin\Psi = B/S \quad (\textit{why?})$$

Step 1: Generate U, V uniformly over the unit circle
Step 2: Set $S^2 = A^2 + B^2$
Step 3: Return

$$X = \sqrt{-2\log(S^2)}\frac{A}{S} = A\sqrt{\frac{-2\log(S^2)}{S^2}}$$

$$Y = \sqrt{-2\log(S^2)}\frac{B}{S} = B\sqrt{\frac{-2\log(S^2)}{S^2}}$$

To simulate a point from the unit circle, use a simple rejection algorithm: generate $U, V \sim$ iid Unif$(-1, 1)$, then accept the point $(A, B) = (U, V)$ if it is inside the unit circle, that is, if $U^2 + V^2 < 1$.

Step 1: Generate $U_1, U_2 \sim$ iid Unif$(0, 1)$ and set $U = 2U_1 - 1$, $V = 2U_2 - 1$

Step 2: Accept $S^2 = U^2 + V^2$ provided $S^2 < 1$, else return to Step 1.

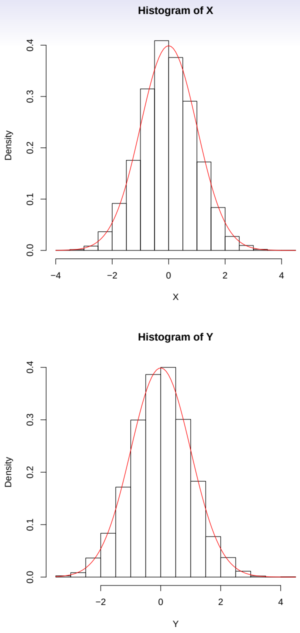
Step 3: Set $W = \sqrt{-2\log(S^2)/S^2}$

Step 4: Return $X = UW$ and $Y = VW$

Since the probability that a random point in the square $[-1, 1] \times [-1, 1]$ will fall within the unit circle is equal to $\pi/4$ (*why?*), it follows that, on average, the (improved) Box-Muller algorithm will require $4/\pi \approx 1.273$ iterations of Step 1, hence, on average, $2(4/\pi) \approx 2.546$ random numbers.

```
S.sq<-1
iter<-0
while (S.sq>=1){
    U<-2*runif(1)-1
    V<-2*runif(1)-1
    S.sq<-U^2+V^2
    iter<-iter+1
}
W<-sqrt(-2*log(S.sq)/S.sq)
X<-U*W
Y<-V*W

mean(num.iter)
[1] 1.279
```



Exercises

- Chapter 5, Ross (2006)
  Ex 13, Ex 14, Ex 15, Ex 16, Ex 17, Ex 18
- Chapter 18, Owen, et al. (2007)
  Ex 18, Example 18.4.1 page 341:

  > *Use the rejection method to generate from the triangular distribution using as proposal* $Unif(0, 2)$

Resources

- BOOKS
  - Owen J., Maillardet R. and Robinson A. (2009).
    Introduction to Scientific Programming and Simulation Using R.
    Chapman & Hall/CRC.
  - Ross, S. (2006).
    Simulation. 4th edn. Academic Press.
- WEB
  - R software:
    `http://www.r-project.org/`
  - Owen, et al. (2009): `http://www.ms.unimelb.edu.au/spuRs/`