Generating Continuous Random Variables

Introduction

Each of the techniques for generating a discrete random variable has its analogue in the continuous case. In Sections 5.1 and 5.2 we present the inverse transform approach and the rejection approach for generating continuous random variables. In Section 5.3 we consider a powerful approach for generating normal random variables, known as the polar method. Finally, in Sections 5.4 and 5.5 we consider the problem of generating Poisson and nonhomogeneous Poisson processes.

5.1 The Inverse Transform Algorithm

Consider a continuous random variable having distribution function F. A general method for generating such a random variable—called the inverse transformation method—is based on the following proposition.

Proposition *Let U be a uniform (0, 1) random variable. For any continuous distribution function F the random variable X defined by*

$$X = F^{-1}(U)$$

has distribution F. [$F^{-1}(u)$ is defined to be that value of x such that $F(x) = u$.]

Proof Let F_X denote the distribution function of $X = F^{-1}(U)$. Then

$$\begin{aligned} F_X(x) &= P\{X \leqslant x\} \\ &= P\{F^{-1}(U) \leqslant x\} \end{aligned} \tag{5.1}$$

Simulation. DOI: http://dx.doi.org/10.1016/B978-0-12-415825-2.00005-X

Now since F is a distribution function it follows that $F(x)$ is a monotone increasing function of x and so the inequality "$a \leqslant b$" is equivalent to the inequality "$F(a) \leqslant F(b)$." Hence, from Equation (5.1), we see that

$$\begin{aligned} F_X(x) &= P\{F(F^{-1}(U)) \leqslant F(x)\} && \text{since } F(F^{-1}(U)) = U \\ &= P\{U \leqslant F(x)\} && \text{since } U \text{ is uniform } (0, 1) \\ &= F(x) \end{aligned}$$ □

The above proposition thus shows that we can generate a random variable X from the continuous distribution function F by generating a random number U and then setting $X = F^{-1}(U)$.

Example 5a Suppose we wanted to generate a random variable X having distribution function

$$F(x) = x^n, \quad 0 < x < 1$$

If we let $x = F^{-1}(u)$, then

$$u = F(x) = x^n \quad \text{or, equivalently,} \quad x = u^{1/n}$$

Hence we can generate such a random variable X by generating a random number U and then setting $X = U^{1/n}$. □

The inverse transform method yields a powerful approach to generating exponential random variables, as is indicated in the next example.

Example 5b If X is an exponential random variable with rate 1, then its distribution function is given by

$$F(x) = 1 - e^{-x}$$

If we let $x = F^{-1}(u)$, then

$$u = F(x) = 1 - e^{-x}$$

or

$$1 - u = e^{-x}$$

or, taking logarithms,

$$x = -\log(1 - u)$$

Hence we can generate an exponential with parameter 1 by generating a random number U and then setting

$$X = F^{-1}(U) = -\log(1 - U)$$

A small savings in time can be obtained by noting that $1 - U$ is also uniform on $(0, 1)$ and thus $-\log(1 - U)$ has the same distribution as $-\log U$. That is, the negative logarithm of a random number is exponentially distributed with rate 1.

In addition, note that if X is exponential with mean 1 then, for any positive constant c, cX is exponential with mean c. Hence, an exponential random variable X with rate λ (mean $1/\lambda$) can be generated by generating a random number U and setting

$$X = -\frac{1}{\lambda} \log U \qquad \square$$

Remark The above also provides us with another algorithm for generating a Poisson random variable. To begin, recall that a Poisson process with rate λ results when the times between successive events are independent exponentials with rate λ. (See Section 2.9 of Chapter 2.) For such a process, $N(1)$, the number of events by time 1, is Poisson distributed with mean λ. However, if we let X_i, $i = 1, \ldots$, denote the successive interarrival times, then the nth event will occur at time $\sum_{i=1}^{n} X_i$, and so the number of events by time 1 can be expressed as

$$N(1) = \text{Max}\left\{n: \sum_{i=1}^{n} X_i \leqslant 1\right\}$$

That is, the number of events by time 1 is equal to the largest n for which the nth event has occurred by time 1. (For example, if the fourth event occurred by time 1 but the fifth event did not, then clearly there would have been a total of four events by time 1.) Hence, using the results of Example 5b, we can generate $N = N(1)$, a Poisson random variable with mean λ, by generating random numbers $U_1, \ldots, U_n, \ldots$ and setting

$$\begin{aligned} N &= \text{Max}\left\{n: \sum_{i=1}^{n} -\frac{1}{\lambda} \log U_i \leqslant 1\right\} \\ &= \text{Max}\left\{n: \sum_{i=1}^{n} \log U_i \geqslant -\lambda\right\} \\ &= \text{Max}\{n: \log(U_1 \cdots U_n) \geqslant -\lambda\} \\ &= \text{Max}\{n: U_1 \cdots U_n \geqslant e^{-\lambda}\} \end{aligned}$$

Hence, a Poisson random variable N with mean λ can be generated by successively generating random numbers until their product falls below $e^{-\lambda}$, and then setting N equal to 1 less than the number of random numbers required. That is,

$$N = \text{Min}\{n: U_1 \cdots U_n < e^{-\lambda}\} - 1 \qquad \square$$

The results of Example 5b along with the relationship between the gamma and the exponential distribution can be used to efficiently generate a gamma (n, λ) random variable.

Example 5c Suppose we wanted to generate the value of a gamma (n, λ) random variable. Since the distribution function F of such a random variable is given by

$$F(x) = \int_0^x \frac{\lambda e^{-\lambda y}(\lambda y)^{n-1}}{(n-1)!}\, dy$$

it is not possible to give a closed form expression for its inverse. However, by using the result that a gamma (n, λ) random variable X can be regarded as being the sum of n independent exponentials, each with rate λ (see Section 2.9 of Chapter 2), we can make use of Example 5b to generate X. Specifically, we can generate a gamma (n, λ) random variable by generating n random numbers $U_1, \ldots, U_n$ and then setting

$$\begin{aligned} X &= -\frac{1}{\lambda}\log U_1 - \cdots - \frac{1}{\lambda}\log U_n \\ &= -\frac{1}{\lambda}\log(U_1 \cdots U_n) \end{aligned}$$

where the use of the identity $\sum_{i=1}^{n} \log x_i = \log(x_1 \cdots x_n)$ is computationally time saving in that it requires only one rather than n logarithmic computations. □

The results of Example 5c can be used to provide an efficient way of generating a set of exponential random variables by first generating their sum and then, conditional on the value of that sum, generating the individual values. For example, we could generate X and Y, a pair of independent and identically distributed exponentials having mean 1, by first generating $X + Y$ and then using the result (Exercise 36 of Chapter 2) that, given that $X + Y = t$, the conditional distribution of X is uniform on $(0, t)$. The following algorithm can thus be used to generate a pair of exponentials with mean 1.

STEP 1: Generate random numbers U_1 and U_2.
STEP 2: Set $t = -\log(U_1 U_2)$.
STEP 3: Generate a random number U_3.
STEP 4: $X = tU_3$, $Y = t - X$.

Comparing the above with the more direct approach of generating two random numbers U_1 and U_2 and then setting $X = -\log U_1$, $Y = -\log U_2$ shows that the above algorithm saves a logarithmic computation at the cost of two multiplications and the generation of a random number.

We can also generate k independent exponentials with mean 1 by first generating their sum, say by $-\log(U_1 \cdots U_k)$, and then generating $k - 1$ additional random numbers $U_1, \ldots, U_{k-1}$, which should then be ordered. If $U_{(1)} < U_{(2)} < \cdots < U_{(k-1)}$ are their ordered values, and if $-\log(U_1 \cdots U_k) = t$, then the k exponentials are

$$t[U_{(i)} - U_{(i-1)}], \quad i = 1, 2, \ldots, k, \quad \text{where } U_{(0)} \equiv 0, \quad U_{(k)} \equiv 1$$

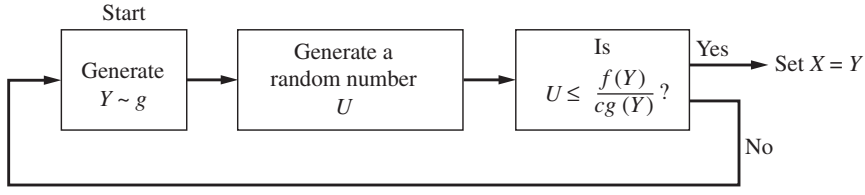

Figure 5.1. The rejection method for simulating a random variable X having density function f.

5.2 The Rejection Method

Suppose we have a method for generating a random variable having density function $g(x)$. We can use this as the basis for generating from the continuous distribution having density function of $f(x)$ by generating Y from g and then accepting this generated value with a probability proportional to $f(Y)/g(Y)$.

Specifically, let c be a constant such that

$$\frac{f(y)}{g(y)} \leqslant c \quad \text{for all } y$$

We then have the following technique (illustrated in Figure 5.1) for generating a random variable having density f.

The Rejection Method

STEP 1: Generate Y having density g.
STEP 2: Generate a random number U.
STEP 3: If $U \leqslant \frac{f(Y)}{cg(Y)}$, set $X = Y$. Otherwise, return to Step 1.

The reader should note that the rejection method is exactly the same as in the case of discrete random variables, with the only difference being that densities replace mass functions. In exactly the same way as we did in the discrete case we can prove the following result.

Theorem

(i) *The random variable generated by the rejection method has density f.*
(ii) *The number of iterations of the algorithm that are needed is a geometric random variable with mean c.*

As in the discrete case it should be noted that the way in which one accepts the value Y with probability $f(Y)/cg(Y)$ is by generating a random number U and then accepting Y if $U \leqslant f(Y)/cg(Y)$.

Example 5d Let us use the rejection method to generate a random variable having density function

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

Since this random variable (which is beta with parameters 2, 4) is concentrated in the interval (0, 1), let us consider the rejection method with

$$g(x) = 1, \quad 0 < x < 1$$

To determine the smallest constant c such that $f(x)/g(x) \leqslant c$, we use calculus to determine the maximum value of

$$\frac{f(x)}{g(x)} = 20x(1-x)^3$$

Differentiation of this quantity yields

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

Setting this equal to 0 shows that the maximal value is attained when $x = \frac{1}{4}$ and thus

$$\frac{f(x)}{g(x)} \leqslant 20\left(\frac{1}{4}\right)\left(\frac{3}{4}\right)^3 = \frac{135}{64} \equiv c$$

Hence,

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

and thus the rejection procedure is as follows:

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

The average number of times that Step 1 will be performed is $c = \frac{135}{64} \approx 2.11$. □

Example 5e Suppose we wanted to generate a random variable having the gamma $(\frac{3}{2}, 1)$ density

$$f(x) = Kx^{1/2}e^{-x}, \quad x > 0$$

where $K = 1/\Gamma(\frac{3}{2}) = 2/\sqrt{\pi}$. Because such a random variable is concentrated on the positive axis and has mean $\frac{3}{2}$, it is natural to try the rejection technique with an exponential random variable with the same mean. Hence, let

$$g(x) = \frac{2}{3}e^{-2x/3}, \quad x > 0$$

Now

$$\frac{f(x)}{g(x)} = \frac{3K}{2} x^{1/2} e^{-x/3}$$

By differentiating and setting the resultant derivative equal to 0, we find that the maximal value of this ratio is obtained when

$$\frac{1}{2} x^{-1/2} e^{-x/3} = \frac{1}{3} x^{1/2} e^{-x/3}$$

that is, when $x = \frac{3}{2}$. Hence

$$\begin{aligned} c = \text{Max} \frac{f(x)}{g(x)} &= \frac{3K}{2} \left(\frac{3}{2}\right)^{1/2} e^{-1/2} \\ &= \frac{3^{3/2}}{(2\pi e)^{1/2}} \quad \text{since } K = 2/\sqrt{\pi} \end{aligned}$$

Since

$$\frac{f(x)}{cg(x)} = (2e/3)^{1/2} x^{1/2} e^{-x/3}$$

we see that a gamma $(\frac{3}{2}, 1)$ random variable can be generated as follows:

STEP 1: Generate a random number U_1 and set $Y = -\frac{3}{2} \log U_1$.
STEP 2: Generate a random number U_2.
STEP 3: If $U_2 < (2eY/3)^{1/2} e^{-Y/3}$, set $X = Y$. Otherwise, return to Step 1.

The average number of iterations that will be needed is

$$c = 3 \left(\frac{3}{2\pi e}\right)^{1/2} \approx 1.257. \qquad \square$$

In the previous example, we generated a gamma random variable using the rejection approach with an exponential distribution having the same mean as the gamma. It turns out that this is always the most efficient exponential to use when generating a gamma random variable. To verify this, suppose we want to generate a random variable having density function

$$f(x) = Ke^{-\lambda x} x^{\alpha - 1}, \quad x > 0$$

where $\lambda > 0, \alpha > 0$, and $K = \lambda^\alpha / \Gamma(\alpha)$. The preceding is the density function of a gamma random variable with parameters α and λ and is known to have mean α/λ.

Suppose we plan to generate the preceding type random variable by the rejection method based on the exponential density with rate μ. Because

$$\frac{f(x)}{g(x)} = \frac{Ke^{-\lambda x} x^{\alpha - 1}}{\mu e^{-\mu x}} = \frac{K}{\mu} x^{\alpha - 1} e^{(\mu - \lambda)x}$$

we see that when $0 < \alpha < 1$

$$\lim_{x \to 0} \frac{f(x)}{g(x)} = \infty$$

thus showing that the rejection technique with an exponential can not be used in this case. As the gamma density reduces to the exponential when $\alpha = 1$, let us suppose that $\alpha > 1$. Now, when $\mu \geqslant \lambda$

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = \infty$$

and so we can restrict attention to values of μ that are strictly less than λ. With such a value of μ, the mean number of iterations of the algorithm that will be required is

$$c(\mu) = \operatorname*{Max}_x \frac{f(x)}{g(x)} = \operatorname*{Max}_x \frac{K}{\mu} x^{\alpha-1} e^{(\mu-\lambda)x}$$

To obtain the value of x at which the preceding maximum occurs, we differentiate and set equal to 0 to obtain

$$0 = (\alpha - 1)x^{\alpha-2} e^{(\mu-\lambda)x} - (\lambda - \mu)x^{\alpha-1} e^{(\mu-\lambda)x}$$

yielding that the maximum occurs at

$$x = \frac{\alpha - 1}{\lambda - \mu}$$

Substituting back yields that

$$\begin{aligned} c(\mu) &= \frac{K}{\mu} \left(\frac{\alpha - 1}{\lambda - \mu} \right)^{\alpha-1} e^{(\mu-\lambda)\left(\frac{\alpha-1}{\lambda-\mu}\right)} \\ &= \frac{K}{\mu} \left(\frac{\alpha - 1}{\lambda - \mu} \right)^{\alpha-1} e^{1-\alpha} \end{aligned}$$

Hence, the value of μ that minimizes $c(\mu)$ is that value that maximizes $\mu(\lambda - \mu)^{\alpha-1}$. Differentiation gives

$$\frac{d}{d\mu}\{\mu(\lambda - \mu)^{\alpha-1}\} = (\lambda - \mu)^{\alpha-1} - (\alpha - 1)\mu(\lambda - \mu)^{\alpha-2}$$

Setting the preceding equal to 0 yields that the best value of μ satisfies

$$\lambda - \mu = (\alpha - 1)\mu$$

or

$$\mu = \lambda/\alpha$$

That is, the exponential that minimizes the mean number of iterations needed by the rejection method to generate a gamma random variable with parameters α and λ has the same mean as the gamma; namely, α/λ.

Our next example shows how the rejection technique can be used to generate normal random variables.

Example 5f Generating a Normal Random Variable To generate a standard normal random variable Z (i.e., one with mean 0 and variance 1), note first that the absolute value of Z has probability density function

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \quad 0 < x < \infty \tag{5.2}$$

We start by generating from the preceding density function by using the rejection method with g being the exponential density function with mean 1—that is,

$$g(x) = e^{-x} \quad 0 < x < \infty$$

Now

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

and so the maximum value of $f(x)/g(x)$ occurs at the value of x that maximizes $x - x^2/2$. Calculus shows that this occurs when $x = 1$, and so we can take

$$c = \text{Max} \frac{f(x)}{g(x)} = \frac{f(1)}{g(1)} = \sqrt{2e/\pi}$$

Because

$$\begin{aligned} \frac{f(x)}{cg(x)} &= \exp\left\{x - \frac{x^2}{2} - \frac{1}{2}\right\} \\ &= \exp\left\{-\frac{(x-1)^2}{2}\right\} \end{aligned}$$

it follows that we can generate the absolute value of a standard normal random variable as follows:

STEP 1: Generate Y, an exponential random variable with rate 1.
STEP 2: Generate a random number U.
STEP 3: If $U \leqslant \exp\{-(Y-1)^2/2\}$, set $X = Y$. Otherwise, return to Step 1.

Once we have simulated a random variable X having density function as in Equation (5.1), —and such a random variable is thus distributed as the absolute value of a standard normal—we can then obtain a standard normal Z by letting Z be equally likely to be either X or $-X$.

In Step 3, the value Y is accepted if $U \leqslant \exp\{-(Y-1)^2/2\}$, which is equivalent to $-\log U \geqslant (Y-1)^2/2$. However, in Example 5b it was shown that $-\log U$ is exponential with rate 1, and so the above is equivalent to the following:

STEP 1: Generate independent exponentials with rate 1, Y_1 and Y_2.
STEP 2: If $Y_2 \geqslant (Y_1 - 1)^2/2$, set $X = Y_1$. Otherwise, return to Step 1.

Suppose now that the foregoing results in Y_1 being accepted—and so we know that Y_2 is larger than $(Y_1 - 1)^2/2$. By how much does the one exceed the other? To answer this, recall that Y_2 is exponential with rate 1, and so, given that it exceeds some value, the amount by which Y_2 exceeds $(Y_1 - 1)^2/2$ [i.e., its "additional life" beyond the time $(Y_1 - 1)^2/2$] is (by the memoryless property) also exponentially distributed with rate 1. That is, when we accept in Step 2 not only do we obtain X (the absolute value of a standard normal) but by computing $Y_2 - (Y_1 - 1)^2/2$ we can also generate an exponential random variable (independent of X) having rate 1.

Hence, summing up, we have the following algorithm that generates an exponential with rate 1 and an independent standard normal random variable.

STEP 1: Generate Y_1, an exponential random variable with rate 1.
STEP 2: Generate Y_2, an exponential random variable with rate 1.
STEP 3: If $Y_2 - (Y_1 - 1)^2/2 > 0$, set $Y = Y_2 - (Y_1 - 1)^2/2$ and go to Step 4. Otherwise, go to Step 1.
STEP 4: Generate a random number U and set

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

The random variables Z and Y generated by the foregoing are independent with Z being normal with mean 0 and variance 1 and Y being exponential with rate 1. (If you want the normal random variable to have mean μ and variable σ^2, just take $\mu + \sigma Z$.) □

Remarks

1. Since $c = \sqrt{2e/\pi} \approx 1.32$, the foregoing requires a geometric distributed number of iterations of Step 2 with mean 1.32.
2. If we want to generate a sequence of standard normal random variables, we can use the exponential random variable Y obtained in Step 3 as the initial exponential needed in Step 1 for the next normal to be generated. Hence, on the average, we can simulate a standard normal by generating $1.64 (= 2 \times 1.32 - 1)$ exponentials and computing 1.32 squares.
3. The sign of the standard normal can be determined without generating a new random number (as in Step 4). The first digit of an earlier random number can be used. That is, an earlier random number $r_1, r_2, \ldots, r_k$ should be used as $r_2, r_3, \ldots, r_k$ with r_1 being used to determine the sign. □

The rejection method is particularly useful when we need to simulate a random variable conditional on it being in some region. This is indicated by our next example.

Example 5g Suppose we want to generate a gamma (2, 1) random variable conditional on its value exceeding 5. That is, we want to generate a random variable having density function

$$f(x) = \frac{xe^{-x}}{\int_5^\infty xe^{-x}dx} = \frac{xe^{-x}}{6e^{-5}}, \quad x \geqslant 5$$

where the preceding integral was evaluated by using integration by parts. Because a gamma (2, 1) random variable has expected value 2, we will use the rejection method based on an exponential with mean 2 that is conditioned to be at least 5. That is, we will use

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

Now,

$$\frac{f(x)}{g(x)} = \frac{e^{5/2}}{3}xe^{-x/2}, \quad x \geqslant 5$$

Because $xe^{-x/2}$ is a decreasing function of x when $x \geqslant 5$, it follows that the number of iterations needed in the algorithm will be geometric with mean

$$c = \underset{x \geqslant 5}{\text{Max}} \left\{ \frac{f(x)}{g(x)} \right\} = \frac{f(5)}{g(5)} = 5/3$$

To generate an exponential with rate 1/2 that is conditioned to exceed 5, we use the fact that the amount by which it exceeds 5 is (by the lack of memory property of exponential random variables) also exponential with rate 1/2. Therefore, if X is exponential with rate 1/2, it follows that $5 + X$ has the same distribution as does X conditioned to exceed 5. Therefore, we have the following algorithm to simulate a random variable X having density function f.

STEP 1: Generate a random number U.
STEP 2: Set $Y = 5 - 2\log(U)$.
STEP 3: Generate a random number U.
STEP 4: If $U \leqslant \frac{e^{5/2}}{5} Y e^{-Y/2}$, set $X = Y$ and stop; otherwise return to step 1. □

Just as we simulated a normal random variable in Example 5f by using the rejection method based on an exponential random variable, we can also effectively simulate a normal random variable that is conditioned to lie in some interval by using the rejection method based on an exponential random variable. The details (including the determination of the best exponential mean) are illustrated in Section 8.8.

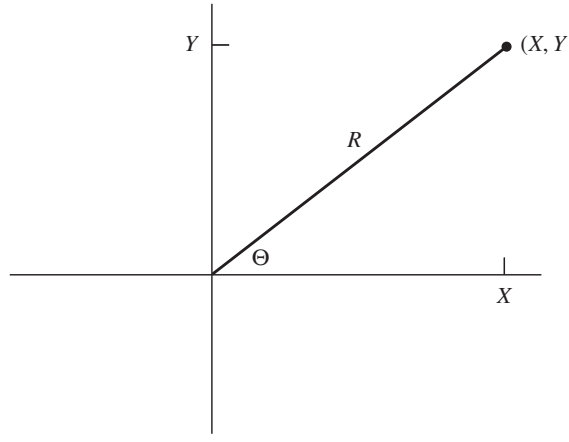

Figure 5.2. Polar Coordinates.

5.3 The Polar Method for Generating Normal Random Variables

Let X and Y be independent standard normal random variables and let R and Θ denote the polar coordinates of the vector (X, Y). That is (see Figure 5.2),

$$R^2 = X^2 + Y^2$$
$$\tan \Theta = \frac{Y}{X}$$

Since X and Y are independent, their joint density is the product of their individual densities and is thus given by

$$\begin{aligned} f(x, y) &= \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \\ &= \frac{1}{2\pi} e^{-(x^2+y^2)/2} \end{aligned} \tag{5.3}$$

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

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

As the Jacobian of this transformation—that is, the determinant of partial derivatives of d and θ with respect to x and y—is easily shown to equal 2, it follows from Equation (5.3) that the joint density function of R^2 and Θ is given by

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

However, as this is equal to the product of an exponential density having mean 2 (namely, $\frac{1}{2}e^{-d/2}$) and the uniform density on $(0, 2\pi)$ [namely, $(2\pi)^{-1}$], it follows that

$$R^2 \text{ and } \Theta \text{ are independent, with } R^2 \text{ being exponential with mean 2 and } \Theta \text{ being uniformly distributed over } (0, 2\pi) \tag{5.4}$$

We can now generate a pair of independent standard normal random variables X and Y by using (5.4) to first generate their polar coordinates and then transform back to rectangular coordinates. This is accomplished as follows:

STEP 1: Generate random numbers U_1 and U_2.
STEP 2: $R^2 = -2\log U_1$ (and thus R^2 is exponential with mean 2). $\Theta = 2\pi\ U_2$ (and thus Θ is uniform between 0 and 2π).
STEP 3: Now let

$$\begin{aligned} X &= R\cos\Theta = \sqrt{-2\log U_1}\cos(2\pi U_2) \\ Y &= R\sin\Theta = \sqrt{-2\log U_1}\sin(2\pi U_2) \end{aligned} \tag{5.5}$$

The transformations given by Equations (5.5) are known as the Box–Muller transformations.

Unfortunately, the use of the Box–Muller transformations (5.5) to generate a pair of independent standard normals is computationally not very efficient: The reason for this is the need to compute the sine and cosine trigonometric functions. There is, however, a way to get around this time-consuming difficulty by a indirect computation of the sine and cosine of a random angle (as opposed to a direct computation which generates U and then computes the sine and cosine of $2\pi U$). To begin, note that if U is uniform on (0, 1) then $2U$ is uniform on (0, 2) and so $2U - 1$ is uniform on $(-1, 1)$. Thus, if we generate random numbers U_1 and U_2 and set

$$\begin{aligned} V_1 &= 2U_1 - 1 \\ V_2 &= 2U_2 - 1 \end{aligned}$$

then (V_1, V_2) is uniformly distributed in the square of area 4 centered at (0, 0)—see Figures 5.3.

Suppose now that we continually generate such pairs (V_1, V_2) until we obtain one that is contained in the circle of radius 1 centered at (0, 0)—that is, until (V_1, V_2) is such that $V_1^2 + V_2^2 \leqslant 1$. It now follows that such a pair (V_1, V_2) is uniformly distributed in the circle. If we let R and Θ denote the polar coordinates of this pair, then it is not difficult to verify that R and Θ are independent, with R^2 being uniformly distributed on (0, 1) (see Exercise 21) and with Θ being uniformly distributed over $(0, 2\pi)$. Since Θ is thus a random angle, it follows that we can generate the sine and cosine of a random angle Θ by generating a random point

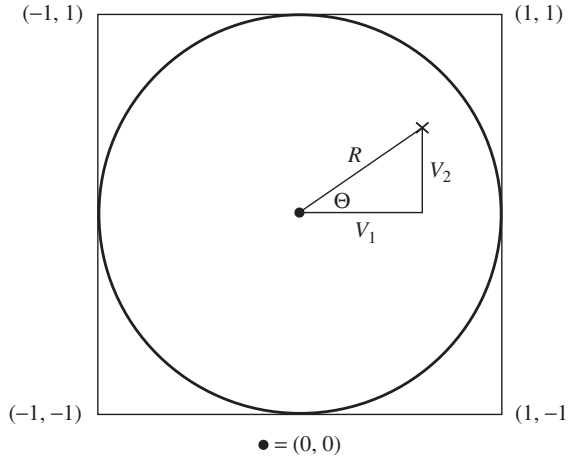

Figure 5.3. (V_1, V_2) Uniformly Distributed in the Square.

(V_1, V_2) in the circle and then setting

$$\sin\Theta = \frac{V_2}{R} = \frac{V_2}{\left(V_1^2 + V_2^2\right)^{1/2}}$$
$$\cos\Theta = \frac{V_1}{R} = \frac{V_1}{\left(V_1^2 + V_2^2\right)^{1/2}}$$

It now follows from the Box–Muller transformation (5.5) that we can generate independent standard normals by generating a random number U and setting

$$X = (-2\log U)^{1/2}\frac{V_1}{\left(V_1^2 + V_2^2\right)^{1/2}}$$
$$Y = (-2\log U)^{1/2}\frac{V_2}{\left(V_1^2 + V_2^2\right)^{1/2}} \tag{5.6}$$

In fact, since $R^2 = V_1^2 + V_2^2$ is itself uniformly distributed over (0, 1) and is independent of the random angle Θ, we can use it as the random number U needed in Equations (5.6). Therefore, letting $S = R^2$, we obtain that

$$X = (-2\log S)^{1/2}\frac{V_1}{S^{1/2}} = V_1\left(\frac{-2\log S}{S}\right)^{1/2}$$
$$Y = (-2\log S)^{1/2}\frac{V_2}{S^{1/2}} = V_2\left(\frac{-2\log S}{S}\right)^{1/2}$$

are independent standard normals when (V_1, V_2) is a randomly chosen point in the circle of radius 1 centered at the origin, and $S = V_1^2 + V_2^2$.

Summing up, we thus have the following approach to generating a pair of independent standard normals:

STEP 1: Generate random numbers, U_1 and U_2.
STEP 2: Set $V_1 = 2U_1 - 1$, $V_2 = 2U_2 - 1$, $S = V_1^2 + V_2^2$.
STEP 3: If $S > 1$ return to Step 1.
STEP 4: Return the independent standard normals.

$$X = \sqrt{\frac{-2\log S}{S}}V_1, \qquad Y = \sqrt{\frac{-2\log S}{S}}V_2$$

The above is called the polar method. Since the probability that a random point in the square will fall within the circle is equal to $\pi/4$ (the area of the circle divided by the area of the square), it follows that, on average, the polar method will require $4/\pi = 1.273$ iterations of Step 1. Hence it will, on average, require 2.546 random numbers, 1 logarithm, 1 square root, 1 division, and 4.546 multiplications to generate two independent unit normals.

5.4 Generating a Poisson Process

Suppose we wanted to generate the first n event times of a Poisson process with rate λ. To do so we make use of the result that the times between successive events for such a process are independent exponential random variables each with rate λ. Thus, one way to generate the process is to generate these interarrival times. So if we generate n random numbers $U_1, U_2, \ldots, U_n$ and set $X_i = -\frac{1}{\lambda}\log U_i$, then X_i can be regarded as the time between the $(i-1)$st and the ith event of the Poisson process. Since the actual time of the jth event will equal the sum of the first j interarrival times, it thus follows that the generated values of the first n event times are $\sum_{i=1}^{j} X_i$, $j = 1, \ldots, n$.

If we wanted to generate the first T time units of the Poisson process, we can follow the preceding procedure of successively generating the interarrival times, stopping when their sum exceeds T. That is, the following algorithm can be used to generate all the event times occurring in $(0, T)$ of a Poisson process having rate λ. In the algorithm t refers to time, I is the number of events that have occurred by time t, and $S(I)$ is the most recent event time.

Generating the First T Time Units of a Poisson Process with Rate λ

STEP 1: $t = 0$, $I = 0$.
STEP 2: Generate a random number U.
STEP 3: $t = t - \frac{1}{\lambda}\log U$. If $t > T$, stop.

STEP 4: $I = I + 1, S(I) = t$.
STEP 5: Go to Step 2.

The final value of I in the preceding algorithm will represent the number of events that occur by time T, and the values $S(1), \ldots, S(I)$ will be the I event times in increasing order.

Another way to simulate the first T time units of a Poisson process with rate λ starts by simulating $N(T)$, the total number of events that occur by time T. Because $N(T)$ is Poisson with mean λT, this is easily accomplished by one of the approaches given in Chapter 4. If the simulated value of $N(T)$ is n, then n random numbers $U_1, \ldots, U_n$ are generated, and $\{TU_1, \ldots, TU_n\}$ are taken as the set of event times by time T of the Poisson process. This works because conditional on $N(T) = n$, the unordered set of event times are distributed as a set of n independent uniform $(0, t)$ random variables.

To verify that the preceding method works, let $N(t)$ equal the number of values in the set $\{TU_1, \ldots, TU_{N(T)}\}$ that are less than t. We must now argue that $N(t), 0 \leqslant t \leqslant T$, is a Poisson process. To show that it has independent and stationary increments, let $I_1, \ldots, I_r$ be r disjoint time intervals in the interval $[0, T]$. Say that the i^{th} Poisson event is a type i event if TU_i lies in the i^{th} of these r disjoint time intervals, $i = 1, \ldots, r$, and say it is type $r + 1$ if it does not lie in any of the r intervals. Because the $U_i, i \geqslant 1$, are independent, it follows that each of the Poisson number of events $N(T)$ is independently classified as being of one of the types $1, \ldots, r + 1$, with respective probabilities $p_1, \ldots, p_{r+1}$, where p_i is the length of the interval I_i divided by T when $i \leqslant r$, and $p_{r+1} = 1 - \sum_{i=1}^{r} p_i$. It now follows, from the results of Section 2.8, that $N_1, \ldots, N_r$, the numbers of events in the disjoint intervals, are independent Poisson random variables, with $E[N_i]$ equal to λ multiplied by the length of the interval I_i; which establishes that $N(t), 0 \leqslant t \leqslant T$, has stationary as well as independent increments. Because the number of events in any interval of length h is Poisson distributed with mean λh, we have

$$\lim_{h \to 0} \frac{P\{N(h) = 1\}}{h} = \lim_{h \to 0} \frac{\lambda h e^{-\lambda h}}{h} = \lambda$$

and

$$\lim_{h \to 0} \frac{P\{N(h) \geqslant 2\}}{h} = \lim_{h \to 0} \frac{1 - e^{-\lambda h} - \lambda h e^{-\lambda h}}{h} = 0$$

which completes the verification.

If all we wanted was to simulate the set of event times of the Poisson process, then the preceding approach would be more efficient than simulating the exponentially distributed interarrival times. However, we usually desire the event times in increasing order; thus, we would also need to order the values $TU_i, i = 1, \ldots, n$.

5.5 Generating a Nonhomogeneous Poisson Process

An extremely important counting process for modeling purposes is the nonhomogeneous Poisson process, which relaxes the Poisson process assumption of stationary increments. Thus, it allows for the possibility that the arrival rate need not be constant but can vary with time. It is usually very difficult to obtain analytical results for a mathematical model that assumes a nonhomogeneous Poisson arrival process, and as a result such processes are not applied as often as they should be. However, because simulation can be used to analyze such models, we expect that such mathematical models will become more common.

Suppose that we wanted to simulate the first T time units of a nonhomogeneous Poisson process with intensity function $\lambda(t)$. The first method we present, called the *thinning* or *random sampling* approach, starts by choosing a value λ which is such that

$$\lambda(t) \leqslant \lambda \quad \text{for all } t \leqslant T$$

Now, as shown in Chapter 2, such a nonhomogeneous Poisson process can be generated by a random selection of the event times of a Poisson process having rate λ. That is, if an event of a Poisson process with rate λ that occurs at time t is counted (independently of what has transpired previously) with probability $\lambda(t)/\lambda$, then the process of counted events is a nonhomogeneous Poisson process with intensity function $\lambda(t), 0 \leqslant t \leqslant T$. Hence, by simulating a Poisson process and then randomly counting its events, we can generate the desired nonhomogeneous Poisson process. This can be written algorithmically as follows.

Generating the First T Time Units of a Nonhomogeneous Poisson Process

STEP 1: $t = 0, I = 0$.
STEP 2: Generate a random number U.
STEP 3: $t = t - \frac{1}{\lambda} \log U$. If $t > T$, stop.
STEP 4: Generate a random number U.
STEP 5: If $U \leqslant \lambda(t)/\lambda$, set $I = I + 1, S(I) = t$.
STEP 6: Go to Step 2.

In the above $\lambda(t)$ is the intensity function and λ is such that $\lambda(t) \leqslant \lambda$. The final value of I represents the number of events time T, and $S(1), \ldots, S(I)$ are the event times.

The above procedure, referred to as the thinning algorithm (because it "thins" the homogeneous Poisson points), is clearly most efficient, in the sense of having the fewest number of rejected events times, when $\lambda(t)$ is near λ throughout the interval. Thus, an obvious improvement is to break up the interval into subintervals and then use the procedure over each subinterval. That is, determine appropriate values $k, 0 = t_0 < t_1 < t_2 < \cdots < t_k < t_{k+1} = T, \lambda_1, \ldots, \lambda_{k+1}$ such that

$$\lambda(s) \leqslant \lambda_i \quad \text{if } t_{i-1} \leqslant s < t_i, \quad i = 1, \ldots, k+1 \tag{5.7}$$

Now generate the nonhomogeneous Poisson process over the interval (t_{i-1}, t_i) by generating exponential random variables with rate λ_i, and accepting the generated event occurring at time s, $s \in (t_{i-1}, t_i)$, with probability $\lambda(s)/\lambda_i$. Because of the memoryless property of the exponential and the fact that the rate of an exponential can be changed upon multiplication by a constant, it follows that there is no loss of efficiency in going from one subinterval to the next. That is, if we are at $t \in (t_{i-1}, t_i)$ and generate X, an exponential with rate λ_i, which is such that $t + X > t_i$, then we can use $\lambda_i[X - (t_i - t)]/\lambda_{i+1}$ as the next exponential with rate λ_{i+1}.

We thus have the following algorithm for generating the first T time units of a nonhomogeneous Poisson process with intensity function $\lambda(s)$ when the relations (5.7) are satisfied. In the algorithm t represents the present time, J the present interval (i.e., $J = j$ when $t_{j-1} \leqslant t < t_j$), I the number of events so far, and $S(1), \ldots, S(I)$ the event times.

Generating the First T Time Units of a Nonhomogeneous Poisson Process

STEP 1: $t = 0$, $J = 1$, $I = 0$.
STEP 2: Generate a random number U and set $X = \frac{-1}{\lambda_J} \log U$.
STEP 3: If $t + X > t_J$, go to Step 8.
STEP 4: $t = t + X$.
STEP 5: Generate a random number U.
STEP 6: If $U \leqslant \lambda(t)/\lambda_J$, set $I = I + 1$, $S(I) = t$.
STEP 7: Go to Step 2.
STEP 8: If $J = k + 1$, stop.
STEP 9: $X = (X - t_J + t)\lambda_J/\lambda_{J+1}$, $t = t_J$, $J = J + 1$.
STEP 10: Go to Step 3.

Suppose now that over some subinterval (t_{i-1}, t_i) we have that $\lambda_i > 0$, where

$$\lambda_i \equiv \text{Infimum}\{\lambda(s): t_{i-1} \leqslant s < t_i\}$$

In such a situation we should not use the thinning algorithm directly but rather should first simulate a Poisson process with rate λ_i over the desired interval and then simulate a nonhomogeneous Poisson process with the intensity function $\lambda(s) = \lambda(s) - \lambda_i$ when $s \in (t_{i-1}, t_i)$. (The final exponential generated for the Poisson process, which carries one beyond the desired boundary, need not be wasted but can be suitably transformed so as to be reusable.) The superposition (or merging) of the two processes yields the desired process over the interval. The reason for doing it this way is that it saves the need to generate uniform random variables for a Poisson distributed number, with mean $\lambda_i(t_i - t_{i-1})$, of the event times. For example, consider the case where

$$\lambda(s) = 10 + s, \quad 0 < s < 1$$

Using the thinning method with $\lambda = 11$ would generate an expected number of 11 events, each of which would require a random number to determine whether or not it should be accepted. On the other hand, to generate a Poisson process with rate 10 and then merge it with a nonhomogeneous Poisson process with rate $\lambda(s) = s, 0 < s < 1$ (generated by the thinning algorithm with $\lambda = 1$), would yield an equally distributed number of event times but with the expected number needing to be checked to determine acceptance being equal to 1.

A second method for simulating a nonhomogeneous Poisson process having intensity function $\lambda(t), t > 0$, is to directly generate the successive event times. So let $S_1, S_2, \ldots$ denote the successive event times of such a process. As these random variables are clearly dependent, we generate them in sequence—starting with S_1, and then using the generated value of S_1 to generate S_2, and so on.

To start, note that if an event occurs at time s, then, independent of what has occurred prior to s, the additional time until the next event has the distribution F_s, given by

$$\begin{aligned} F_s(x) &= P\{\text{time from } s \text{ until next event is less than } x | \text{event at } s\} \\ &= P\{\text{next event is before } x + s | \text{event at } s\} \\ &= P\{\text{event between } s \text{ and } s + x | \text{event at } s\} \\ &= P\{\text{event between } s \text{ and } s + x\} \text{ by independent increments} \\ &= 1 - P\{0 \text{ events in}(s, s + x)\} \\ &= 1 - \exp\left(-\int_s^{s+x} \lambda(y)\, dy\right) \\ &= 1 - \exp\left(-\int_0^x \lambda(s + y) dy\right) \end{aligned} \tag{5.8}$$

We can now simulate the event times $S_1, S_2, \ldots$ by generating S_1 from the distribution F_0; if the simulated value of S_1 is s_1, we generate S_2 by adding s_1 to a generated value from the distribution F_{s_1}; if this sum is s_2 we generate S_3 by adding s_2 to a generated value from the distribution F_{s_2}; and so on. The method used to simulate from these distributions should of course depend on their form. In the following example the distributions F_s are easily inverted and so the inverse transform method can be applied.

Example 5h Suppose that $\lambda(t) = 1/(t+a), t \geqslant 0$, for some positive constant a. Then

$$\int_0^x \lambda(s + y) dy = \int_0^x \frac{1}{s + y + a}\, dy = \log\left(\frac{x + s + a}{s + a}\right)$$

Hence, from Equation (5.8),

$$F_s(x) = 1 - \frac{s + a}{x + s + a} = \frac{x}{x + s + a}$$

To invert this, suppose that $x = F_s^{-1}(u)$, and so

$$u = F_s(x) = \frac{x}{x + s + a}$$

or, equivalently,

$$x = \frac{u(s + a)}{1 - u}$$

That is,

$$F_s^{-1}(u) = (s + a)\,\frac{u}{1 - u}$$

We can therefore generate the successive event times $S_1, S_2, \ldots$ by generating random numbers $U_1, U_2, \ldots$ and then recursively setting

$$S_1 = \frac{aU_1}{1 - U_1}$$

$$S_2 = S_1 + (S_1 + a)\frac{U_2}{1 - U_2} = \frac{S_1 + aU_2}{1 - U_2}$$

and, in general,

$$S_j = S_{j-1} + (S_{j-1} + a)\,\frac{U_j}{1 - U_j} = \frac{S_{j-1} + aU_j}{1 - U_j}, \quad j \geqslant 2$$ □

5.6 Simulating a Two-Dimensional Poisson Process

A process consisting of randomly occurring points in the plane is said to constitute a two-dimensional Poisson process having rate λ, $\lambda > 0$, if

1. The number of points occurring in any given region of area A is Poisson distributed with mean λA.
2. The numbers of points occurring in disjoint regions are independent.

For a given fixed point $\mathbf{0}$ in the plane, we now show how to simulate points, according to a two-dimensional Poisson process with rate λ, that occur in a circular region of radius r centered at $\mathbf{0}$.

Let $C(a)$ denote the circle of radius a centered at $\mathbf{0}$, and note that, from Condition 1, the number of points in $C(a)$ is Poisson distributed with mean $\lambda\pi a^2$. Let R_i, $i \geq 1$, denote the distance from the origin $\mathbf{0}$ to its ith nearest point (Figure 5.4). Then

$$\begin{aligned} P\left\{\pi R_1^2 > x\right\} &= P\{R_1 > \sqrt{x/\pi}\} \\ &= P\left\{\text{no points in C}\left(\sqrt{\frac{x}{\pi}}\right)\right\} \\ &= e^{-\lambda x} \end{aligned}$$

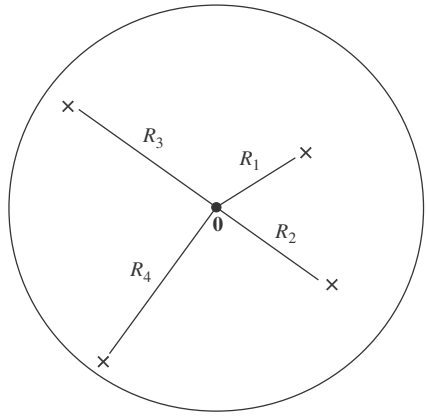

Figure 5.4. Two Dimensional Poisson Process.

where the last equality uses the fact that the area of $C(\sqrt{x/\pi})$ is x. Also, with $C(b) - C(a)$ denoting the region between $C(b)$ and $C(a)$, $a < b$, we have

$$\begin{aligned}
&P\left\{\pi R_2^2 - \pi R_1^2 > x | R_1 = a\right\} \\
&= P\left\{R_2 > \sqrt{\frac{x + \pi R_1^2}{\pi}} | R_1 = a\right\} \\
&= P\left\{\text{no points in } C\left(\sqrt{\frac{x + \pi a^2}{\pi}}\right) - C(a) | R_1 = a\right\} \\
&= P\left\{\text{no points in } C\left(\sqrt{\frac{x + \pi a^2}{\pi}}\right) - C(a)\right\} \quad \text{by Condition 2} \\
&= e^{-\lambda x}
\end{aligned}$$

In fact, the same argument can be repeated continually to obtain the following proposition.

Proposition *With $R_0 = 0, \pi R_i^2 - \pi R_{i-1}^2, i \geq 1$, are independent exponential random variables each having rate λ.*

In other words, the amount of area that need be traversed to encounter a Poisson point is exponential with rate λ. Since, by symmetry, the respective angles of the Poisson points are independent and uniformly distributed over $(0, 2\pi)$, we thus have the following algorithm for simulating the Poisson process over a circular region of radius r about $\mathbf{0}$.

STEP 1: Generate independent exponentials with rate λ, $X_1, X_2, \ldots$, stopping at

$$N = \text{Min}\{n: X_1 + \cdots + X_n > \pi r^2\}$$

STEP 2: If $N = 1$ stop; there are no points in $C(r)$. Otherwise, for $i = 1, \ldots, N-1$, set

$$R_i = \sqrt{\frac{X_1 + \cdots + X_i}{\pi}}$$

(that is, $\pi R_i^2 = X_1 + \cdots + X_i$).

STEP 3: Generate random numbers $U_1, \ldots, U_{N-1}$.

STEP 4: The polar coordinates of the $N - 1$ Poisson points are

$$(R_i, 2\pi U_i), \quad i = 1, \ldots, N - 1$$

The above algorithm can be considered as the fanning out from a circle centered at $\mathbf{0}$ with a radius that expands continuously from 0 to r. The successive radii at which points are encountered are simulated by using the result that the additional area necessary to explore until one encounters another point is always exponentially distributed with rate λ. This fanning-out technique can also be used to simulate the process over noncircular regions. For example, consider a nonnegative function $f(x)$ and suppose that we are interested in simulating the Poisson process in the region between the x-axis and the function f (Figure 5.5) with x going from 0 to T. To do so, we can start at the left-hand edge and fan vertically to the right by considering the successive areas encountered. Specifically, if $X_1 < X_2 < \cdots$ denote the successive projections of the Poisson process points on the x-axis, it follows in exactly the same manner as before that (with $X_0 = 0$)

$$\int_{X_{i-1}}^{X_i} f(x)\,dx, \quad i = 1, \ldots, \text{ are independent with rate } \lambda$$

Hence, we can simulate the Poisson points by generating independent exponential random variables with rate λ, $W_1, W_2, \ldots$, stopping at

$$N = \text{Min}\left\{n: W_1 + \cdots + W_n > \int_0^T f(x)dx\right\}$$

We now determine $X_1, \ldots, X_{N-1}$ by using the equations

$$\begin{aligned}
\int_0^{X_1} f(x)dx &= W_1 \\
\int_{X_1}^{X_2} f(x)dx &= W_2 \\
&\vdots \\
\int_{X_{N-2}}^{X_{N-1}} f(x)dx &= W_{N-1}
\end{aligned}$$

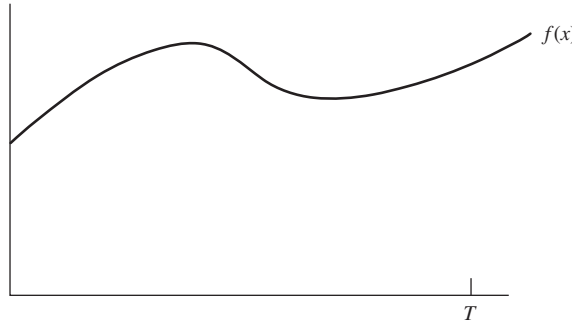

Figure 5.5. Graph of f.

Because the projection on the y-axis of the point whose x-coordinate is X_i is clearly uniformly distributed over $(0, f(X_i))$, it thus follows that if we now generate random numbers $U_1, \ldots, U_{N-1}$, then the simulated Poisson points are, in rectangular coordinates, $(X_i, U_i f(X_i)), i = 1, \ldots, N-1$.

The above procedure is most useful when f is regular enough so that the above equations can be efficiently solved for the values of X_i. For example, if $f(x) = c$ (and so the region is a rectangle), we can express X_i as

$$X_i = \frac{W_1 + \cdots + W_i}{c}$$

and the Poisson points are

$$(X_i, cU_i), \quad i = 1, \ldots, N-1$$

Exercises

1. Give a method for generating a random variable having density function

$$f(x) = e^x/(e-1), \quad 0 \leqslant x \leqslant 1$$

2. Give a method to generate a random variable having density function

$$f(x) = \begin{cases} \frac{x-2}{2} & \text{if } 2 \leqslant x \leqslant 3 \\ \frac{2-x/3}{2} & \text{if } 3 \leqslant x \leqslant 6 \end{cases}$$

3. Use the inverse transform method to generate a random variable having distribution function

$$F(x) = \frac{x^2 + x}{2}, \quad 0 \leqslant x \leqslant 1$$

4. Give a method for generating a random variable having distribution function

$$F(x) = 1 - \exp(-\alpha x^{\beta}), \quad 0 < x < \infty$$

A random variable having such a distribution is said to be a Weibull random variable.

5. Give a method for generating a random variable having density function

$$f(x) = \begin{cases} e^{2x}, & -\infty < x < 0 \\ e^{-2x}, & 0 < x < \infty \end{cases}$$

6. Let X be an exponential random variable with mean 1. Give an efficient algorithm for simulating a random variable whose distribution is the conditional distribution of X given that $X < 0.05$. That is, its density function is

$$f(x) = \frac{e^{-x}}{1 - e^{-0.05}}, \quad 0 < x < 0.05$$

Generate 1000 such variables and use them to estimate $E[X|X < 0.05]$. Then determine the exact value of $E[X|X < 0.05]$.

7. (The Composition Method) Suppose it is relatively easy to generate random variables from any of the distributions $F_i, i = 1, \ldots, n$. How could we generate a random variable having the distribution function

$$F(x) = \sum_{i=1}^{n} p_i F_i(x)$$

where $p_i, i = 1, \ldots, n$, are nonnegative numbers whose sum is 1?

8. Using the result of Exercise 7, give algorithms for generating random variables from the following distributions.

(a) $F(x) = \frac{x+x^3+x^5}{3}, 0 \leqslant x \leqslant 1$

(b) $F(x) = \begin{cases} \frac{1-e^{-2x}+2x}{3} & \text{if} 0 < x < 1 \\ \frac{3-e^{-2x}}{3} & \text{if} 1 < x < \infty \end{cases}$

(c) $F(x) = \sum_{i=1}^{n} \alpha_i x^i, 0 \leqslant x \leqslant 1, \quad \text{where } \alpha_i \geqslant 0, \quad \sum_{i=1}^{n} \alpha_i = 1$

9. Give a method to generate a random variable having distribution function

$$F(x) = \int_0^{\infty} x^y e^{-y}\, dy, \quad 0 \leqslant x \leqslant 1$$

[Hint: Think in terms of the composition method of Exercise 7. In particular, let F denote the distribution function of X, and suppose that the conditional distribution of X given that $Y = y$ is

$$P\{X \leqslant x|Y = y\} = x^y, \quad 0 \leqslant x \leqslant 1$$

10. A casualty insurance company has 1000 policyholders, each of whom will independently present a claim in the next month with probability .05. Assuming that the amounts of the claims made are independent exponential random variables with mean \$800, use simulation to estimate the probability that the sum of these claims exceeds \$50,000.

11. Write an algorithm that can be used to generate exponential random variables in sets of 3. Compare the computational requirements of this method with the one presented after Example 5c which generates them in pairs.

12. Suppose it is easy to generate random variable from any of the distribution $F_i, i = 1, \ldots, n$. How can we generate from the following distributions?

(a) $F(x) = \prod_{i=1}^{n} F_i(x)$
(b) $F(x) = 1 - \prod_{i=1}^{n}[1 - F_i(x)]$

[Hint: If $X_i, i = 1, \ldots, n$, are independent random variables, with X_i having distribution F_i, what random variable has distribution function F?]

13. Using the rejection method and the results of Exercise 12, give two other methods, aside from the inverse transform method, that can be used to generate a random variable having distribution function

$$F(x) = x^n, \quad 0 \leqslant x \leqslant 1$$

Discuss the efficiency of the three approaches to generating from F.

14. Let G be a distribution function with density g and suppose, for constants $a < b$, we want to generate a random variable from the distribution function

$$F(x) = \frac{G(x) - G(a)}{G(b) - G(a)}, \quad a \leqslant x \leqslant b$$

(a) If X has distribution G, then F is the conditional distribution of X given what information?
(b) Show that the rejection method reduces in this case to generating a random variable X having distribution G and then accepting it if it lies between a and b.

15. Give two methods for generating a random variable having density function

$$f(x) = xe^{-x}, \quad 0 \leqslant x < \infty$$

and compare their efficiency.

16. Give two algorithms for generating a random variable having distribution function

$$F(x) = 1 - e^{-x} - e^{-2x} + e^{-3x}, \quad x > 0$$

17. Give two algorithms for generating a random variable having density function

$$f(x) = \frac{1}{4} + 2x^3 + \frac{5}{4}x^4, \quad 0 < x < 1$$

18. Give an algorithm for generating a random variable having density function

$$f(x) = 2xe^{-x^2}, \quad x > 0$$

19. Show how to generate a random variable whose distribution function is

$$F(x) = \frac{1}{2}(x + x^2), \quad 0 \le x \le 1$$

using

(a) the inverse transform method;
(b) the rejection method;
(c) the composition method.

Which method do you think is best for this example? Briefly explain your answer.

20. Use the rejection method to find an efficient way to generate a random variable having density function

$$f(x) = \frac{1}{2}(1 + x)e^{-x}, \quad 0 < x < \infty$$

21. When generating a gamma random variable with parameters $(\alpha, 1), \alpha < 1$, that is conditioned to exceed c by using the rejection technique with an exponential conditioned to exceed c, what is the best exponential to use? Is it necessarily the one with mean α, the mean of the gamma $(\alpha, 1)$ random variable?

22. Give an algorithm that generates a random variable having density

$$f(x) = 30(x^2 - 2x^3 + x^4), \quad 0 \leqslant x \leqslant 1$$

Discuss the efficiency of this approach.

23. Give an efficient method to generate a random variable X having density

$$f(x) = \frac{1}{.000336}x(1 - x)^3, \ .8 < x < 1$$

24. In Example 5f we simulated a normal random variable by using the rejection technique with an exponential distribution with rate 1. Show that among all exponential density functions $g(x) = \lambda e^{-\lambda x}$ the number of iterations needed is minimized when $\lambda = 1$.

25. Write a program that generates normal random variables by the method of Example 5f.

26. Let (X, Y) be uniformly distributed in a circle of radius 1. Show that if R is the distance from the center of the circle to (X, Y) then R^2 is uniform on (0, 1).

27. Write a program that generates the first T time units of a Poisson process having rate λ.

28. To complete a job a worker must go through k stages in sequence. The time to complete stage i is an exponential random variable with rate λ_i, $i = 1, \ldots, k$. However, after completing stage i the worker will only go to the next stage with probability α_i, $i = 1, \ldots, k-1$. That is, after completing stage i the worker will stop working with probability $1-\alpha_i$. If we let X denote the amount of time that the worker spends on the job, then X is called a *Coxian* random variable. Write an algorithm for generating such a random variable.

29. Buses arrive at a sporting event according to a Poisson process with rate 5 per hour. Each bus is equally likely to contain either 20, 21,..., 40 fans, with the numbers in the different buses being independent. Write an algorithm to simulate the arrival of fans to the event by time $t = 1$.

30.

(a) Write a program that uses the thinning algorithm to generate the first 10 time units of a nonhomogeneous Poisson process with intensity function

$$\lambda(t) = 3 + \frac{4}{t+1}$$

(b) Give a way to improve upon the thinning algorithm for this example.

31. Give an efficient algorithm to generate the first 10 times units of a nonhomogeneous Poisson process having intensity function

$$\lambda(t) = \begin{cases} \frac{t}{5}, & 0 < t < 5 \\ 1 + 5(t-5), & 5 < t < 10 \end{cases}$$

32. Write a program to generate the points of a two-dimensional Poisson process within a circle of radius R, and run the program for $\lambda = 1$ and $R = 5$. Plot the points obtained.

Bibliography

Dagpunar, T., *Principles of Random Variate Generation*. Clarendon Press, Oxford, 1988.

Devroye, L., *Nonuniform Random Variate Generation*. Springer-Verlag, New York, 1986.

Fishman, G. S., *Principles of Discrete Event Simulation*. Wiley, New York, 1978.

Knuth, D., *The Art of Computer Programming*, Vol. 2, 2nd ed., *Seminumerical Algorithms*. Addison-Wesley, Reading, MA, 2000.

Law, A. M., and W. D. Kelton, *Simulation Modelling and Analysis*, 3rd ed. McGraw-Hill, New York, 1997.

Lewis, P. A. W., and G. S. Shedler, "Simulation of Nonhomogeneous Poisson Processes by Thinning," *Nav. Res. Log. Quart.*, **26**, 403–413, 1979.

Marsaglia, G., "Generating Discrete Random Variables in a Computer," *Commun. Assoc. Comput. Mach.*, **6**, 37–38, 1963.

Morgan, B. J. T., *Elements of Simulation*. Chapman and Hall, London, 1983.

Ripley, B. D., "Computer Generation of Random Variables: A Tutorial," *Inst. Statist. Rev.*, **51**, 301–319, 1983.

Ripley, B. D., *Stochastic Simulation*. Wiley, New York, 1986.

Rubenstein, R. Y., *Simulation and the Monte Carlo Method*. Wiley, New York, 1981.

Schmeiser, B. W., "Random Variate Generation, a Survey," *Proc. 1980 Winter Simulation Conf.,* Orlando, FL; pp. 79–104, 1980.