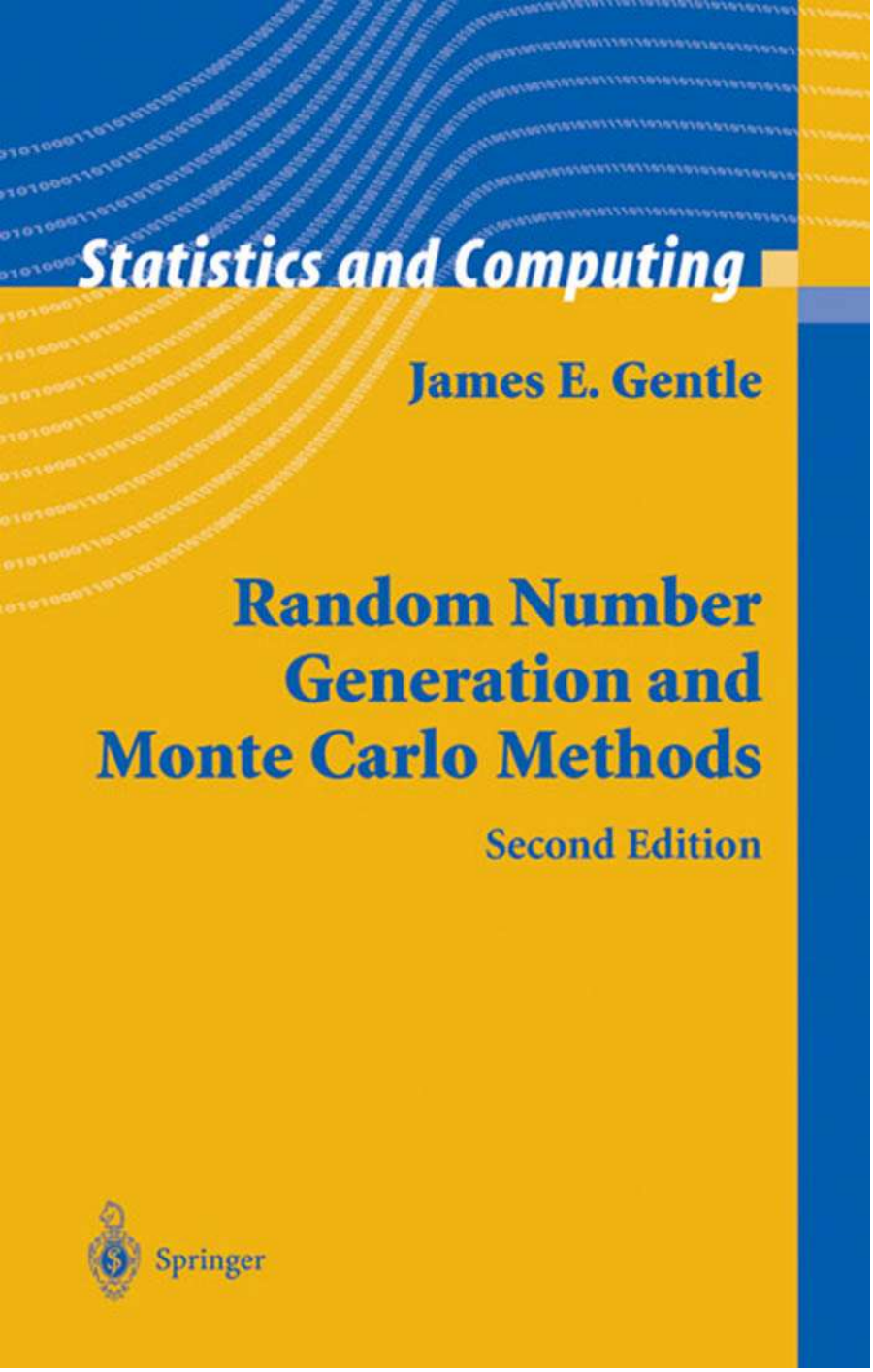
Statistics and Computing
James E. Gentle
Random Number Generation and Monte Carlo Methods
Second Edition
Springer

Chapter 4

Transformations of Uniform Deviates: General Methods

Sampling of random variates from a nonuniform distribution is usually done by applying a transformation to uniform variates. Each realization of the nonuniform random variable might be obtained from a single uniform variate or from a sequence of uniforms. Some methods that use a sequence of uniforms require that the sequence be independent; other methods use a random walk sequence, a Markov chain.

For some distributions, there may be many choices of algorithms for generation of random numbers. The algorithms differ in speed, accuracy, storage requirements, and complexity of coding. Some of the faster methods are approximate, but given the current cost of computing, the speedup resulting from an additional approximation beyond the approximation resulting from the ordinary finite-precision representation is not worth the accuracy loss. All of the methods that we discuss in this chapter are exact, so any approximation is a result of the ordinary rounding and truncation necessary in using a computer to simulate real numbers or infinite sets. Occasionally, a research paper will contend that the quality of the random numbers generated by some particular method, such as Box–Muller or the ratio-of-uniforms, is bad, but the quality ultimately depends only on the quality of the underlying uniform generator. A particular method, however, may exacerbate some fault in the uniform generator, and it is always a good idea to conduct a goodness-of-fit test for the specific distribution of interest. This is an ad hoc test, as we advocate in Chapter 2.

After accuracy, the next most important criterion is speed. The speed of a random number generation algorithm has two aspects: the setup time and the generation time. In most cases, the generation time is the more important component to optimize. Whenever the setup time is significant, the computer program can preserve the variables initialized, so that if the function is called again with the same parameters, the setup step can be bypassed. In a case of relatively expensive setup overhead, a software system may provide a second

function for the same distribution with less setup time.

The two other criteria mentioned above, storage requirements and complexity of coding, are generally of very little concern in selecting algorithms for production random number generators.

Another important issue is whether the algorithm can be implemented in a portable manner, as discussed in Section 1.11.

The methods discussed in this chapter are "universal" in the sense that they apply to many different distributions. (Some authors call these "black box" methods.) Some of these methods are better than others for a particular distribution or for a particular range of the distribution. These techniques are used, either singly or in combination, for particular distributions. We discuss these methods in Chapter 5.

Some of these methods, especially those that involve inverting a function, apply directly only to univariate random variables, whereas other methods apply immediately to multivariate random variables.

The descriptions of the algorithms in this chapter are written with an emphasis on clarity, so they should not be incorporated directly into program code without considerations of the efficiency. These considerations generally involve avoiding unnecessary computations. This may mean defining a variable not mentioned in the algorithm description or reordering the steps slightly.

4.1 Inverse CDF Method

For the random variable X, the *cumulative distribution function*, or *CDF*, is the function P_X defined by

$$P_X(x) = \Pr(X \leq x),$$

where $\Pr(A)$ represents the probability of the event A. Two important properties are immediately obvious: the CDF is nondecreasing, and it is continuous from the right.

Continuous Distributions

If X is a scalar random variable with a continuous CDF P_X, then the random variable

$$U = P_X(X)$$

has a $\mathrm{U}(0,1)$ distribution. (This is easy to show; you are asked to do that in Exercise 4.1, page 159.) This fact provides a very simple relationship with a uniform random variable U and a random variable X with distribution function P:

$$X = P_X^{-1}(U). \tag{4.1}$$

Use of this straightforward transformation is called the *inverse CDF* technique. The reason it works can be seen in Figure 4.1; over a range for which the

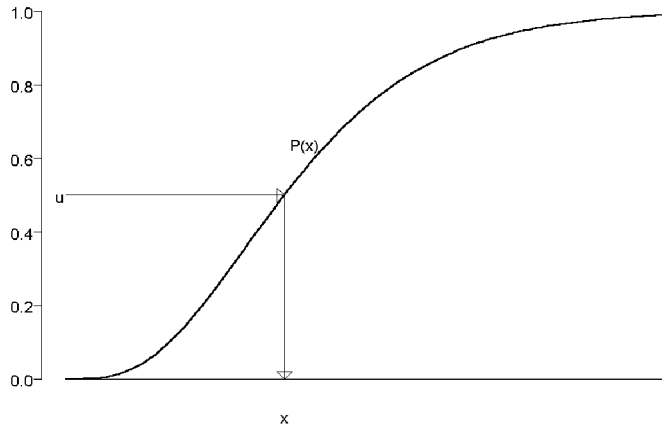

Figure 4.1: The Inverse CDF Method to Convert a Uniform Random Number to a Number from a Continuous Distribution

derivative of the CDF (the density) is large, there is more probability of realizing a uniform deviate.

The inverse CDF relationship exists between any two continuous (nonsingular) random variables. If X is a continuous random variable with CDF P_X and Y is a continuous random variable with CDF P_Y, then

$$X = P_X^{-1}(P_Y(Y))$$

over the ranges of positive support. Use of this kind of relationship is a matching of "scores" (that is, of percentile points) of one distribution with those of another distribution. In addition to the uniform distribution, as above, this kind of transformation is sometimes used with the normal distribution.

Whenever the inverse of the distribution function is easy to compute, the inverse CDF method is a good one. It also has the advantage that basic relationships among a set of uniform deviates (such as order relationships) may result in similar relationships among the set of deviates from the other distribution.

Because it is relatively difficult to compute the inverse of some distribution functions of interest, however, the inverse CDF method is not as commonly used as its simplicity might suggest. Even when the inverse P^{-1} exists in closed form, evaluating it directly may be much slower than use of some alternative method for sampling random numbers. On the other hand, in some cases when P^{-1} does not exist in closed form, use of the inverse CDF method by solving the equation

$$P(x) - u = 0$$

may be better than the use of any other method.

Discrete Distributions

The inverse CDF method also applies to discrete distributions, but of course we cannot take the inverse of the distribution function. Suppose that the discrete random variable X has mass points

$$m_1 < m_2 < m_3 < \ldots$$

with probabilities

$$p_1,\ p_2,\ p_3,\ \ldots$$

and with the distribution function

$$P(x) = \sum_{i \ni m_i \leq x} p_i.$$

To use the inverse CDF method for this distribution, we first generate a realization u of the uniform random variable U. We then deliver the realization of the target distribution as x, where x satisfies the relationship

$$P(x_{(-)}) < u \leq P(x), \tag{4.2}$$

where $x_{(-)}$ is a value arbitrarily close to, but less than, x. Alternatively, we have

$$x = \min\{t,\ \text{s.t.}\ u \leq P(t)\}. \tag{4.3}$$

This is illustrated in Figure 4.2.

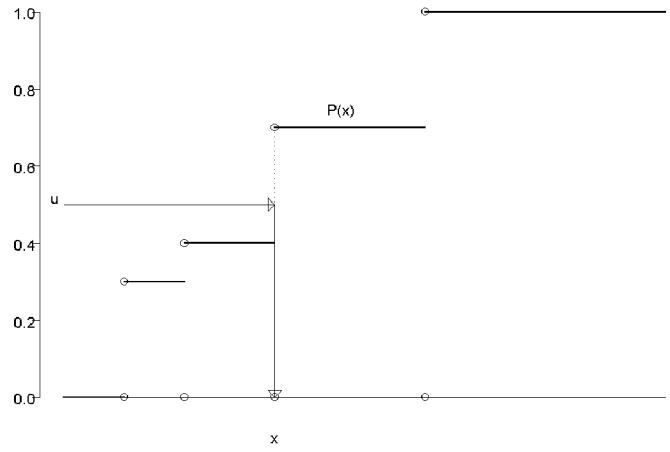

Figure 4.2: The Inverse CDF Method to Convert a Uniform Random Number to a Number from a Discrete Distribution

An example of a common and very simple application of the inverse CDF technique is for generating a random deviate from a Bernoulli distribution with parameter π, as in Algorithm 4.1. The probability function for the Bernoulli distribution with parameter π is

$$p(x) = \pi^x(1-\pi)^{1-x}, \quad \text{for } x = 0, 1, \tag{4.4}$$

where $0 < \pi < 1$.

Algorithm 4.1 Generating a Bernoulli Deviate by the Inverse CDF

1. Generate u from a $\mathrm{U}(0,1)$ distribution.
2. If $u < \pi$, then
   2.a. deliver 0;
   otherwise,
   2.b. deliver 1. ∎

Without loss of generality, we often assume that the mass points of a discrete distribution are the integers $1, 2, 3, \ldots$. The special case in which there are k mass points and they all have equal probability is called the discrete uniform distribution, and the use of the inverse CDF method is particularly simple: the value is $\lceil uk \rceil$.

The evaluation of the inverse CDF involves a search. Rather than starting the search at some arbitrary point, it is usually more efficient to start it at some point with a higher probability of being near the solution. Either the mean or the mode is usually a good place to begin the search, which then proceeds up or down as necessary.

Table Lookup

Using the inverse CDF method for a general discrete distribution is essentially a table lookup; it usually requires a search for the x in equation (4.2). The search may be performed sequentially or by some tree traversal. Marsaglia (1963), Norman and Cannon (1972), and Chen and Asau (1974) describe various ways of speeding up the table lookup. Improving the efficiency of the table-lookup method is often done by incorporating some aspects of the *urn method*, in which the distribution is simulated by a table (an "urn") that contains the mass points in proportion to their population frequency. In the urn, each p_i is represented as a rational fraction n_i/N, and a table of length N is constructed with n_i pointers to the mass point i. A discrete uniform deviate, $\lceil uN \rceil$, is then used as an index to the table to yield the target distribution.

The method of Marsaglia implemented by Norman and Cannon (1972) involves forming a table partitioned in such a way that the individual partitions can be sampled with equal probabilities. In this scheme, the probability p_i associated with the i^{th} mass point is expressed to a precision t in the base b as

$$p_i \approx \sum_{j=1}^{t} d_{ij} b^{-j}, \tag{4.5}$$

with $0 \le d_{ij} < b$. We assume that there are n mass points. (We frequently assume a finite set of mass points when working with discrete distributions. Because we can work with only a finite number of different values on the computer, it does not directly limit our methods, but we should be aware of any such exclusion of rare events and in some cases must modify our methods to be able to model rare events.) Let

$$\begin{aligned} P_0 &= 0, \\ P_j &= b^{-j} \sum_{i=1}^{n} d_{ij} \quad \text{for } j = 1, 2, \ldots, t, \\ N_0 &= 0, \\ N_k &= b^{-j} \sum_{j=1}^{k} \sum_{i=1}^{n} d_{ij} \quad \text{for } k = 1, 2, \ldots, t. \end{aligned}$$

Now, form a partitioned array and, in the j^{th} partition, store d_{ij} copies of i (remember that the mass points are taken to be the integers; they could just as well be indexes). The j^{th} partition is in locations $N_{j-1}+1$ to N_j. There are N_t storage locations in all. Norman and Cannon (1972) show how to reduce the size of the table with a slight modification to the method. After the partitions are set up, Algorithm 4.2 generates a random number from the given distribution.

Algorithm 4.2 Marsaglia/Norman/Cannon Table Lookup for Sampling a Discrete Random Variate

1. Generate u from a $\mathrm{U}(0,1)$ distribution, and represent it in base b to t places:
$$u = \sum_{j=1}^{t} d_j b^{-j}.$$

2. Find m such that
$$\sum_{j=0}^{m-1} P_j \le u < \sum_{j=0}^{m} P_j.$$

3. Take as the generated value the contents of location
$$\sum_{j=1}^{m} d_j b^{m-j} - \left(b^m \sum_{j=0}^{m-1} P_j - N_{m-1} \right) + 1.$$

∎

In this algorithm, the j^{th} partition is chosen with probability P_j. The probability of the i^{th} mass point is

$$\begin{aligned}
\Pr(X=i) &= \sum_{j=1}^{t} \Pr(j^{\text{th}} \text{ partition is chosen}) \times \\
&\qquad \Pr(i \text{ is chosen from } j^{\text{th}} \text{ partition}) \\
&= \sum_{j=1}^{t} P_j \frac{d_{ij}}{\sum_{k=1}^{n} d_{kj}} \\
&= \sum_{j=1}^{t} b^{-j} \sum_{i=1}^{n} d_{ij} \frac{d_{ij}}{\sum_{k=1}^{n} d_{kj}} \\
&= \sum_{j=1}^{t} d_{ij} b^{-j} \\
&\approx p_i.
\end{aligned}$$

Norman and Cannon (1972) give a program to implement this algorithm that forms an equivalent but more compact partitioned array.

Chen and Asau (1974) give a hashing method using a "guide table". The guide table contains n values g_i that serve as indexes to the n mass points in the CDF table. The i^{th} guide value is the index of the largest mass point whose CDF value is less than i/n:

$$g_i = \max_{\sum_{k=1}^{j} p_k < i/n} j.$$

After the guide table is set up, Algorithm 4.3 generates a random number from the given distribution.

Algorithm 4.3 Sampling a Discrete Random Variate Using the Chen and Asau Guide Table Method

1. Generate u from a $\mathrm{U}(0,1)$ distribution, and set $i = \lceil un \rceil$.
2. Set $x = g_i + 1$.
3. While $\sum_{k=1}^{x} p_k > u$, set $x = x - 1$. ∎

Efficiency of the Inverse CDF for Discrete Distributions

Rather than using a stored table of the mass points of the distribution, we may seek other efficient methods of searching for the x in equation (4.2). The search can often be improved by knowledge of the relative magnitude of the probabilities of the points. The basic idea is to begin at a point with a high probability of satisfying the relation (4.2). Obviously, the mode is a good place to begin the search, especially if the probability at the mode is quite high.