Simulation

Third Edition

Sheldon M. Ross

Department of Industrial Engineering
and Operations Research
University of California
Berkeley, California

ACADEMIC PRESS

An Elsevier Science Imprint

San Diego London Boston New York
Sydney Tokyo Toronto

References

Knuth, D., *The Art of Computer Programming*, Vol. 2, 2nd ed., *Seminumerical Algorithms*. Addison-Wesley, Reading, MA, 2000.
L'Ecuyer, P., "Random Numbers for Simulation," *Commun. Assoc. Comput. Mach.* **33,** 1990.
Marsaglia, G., "Random Numbers Fall Mainly in the Planes," *Proc. Natl. Acad. Sci. USA* **61,** 25–28, 1962.
Marsaglia, G., "The Structure of Linear Congruential Sequences," in *Applications of Number Theory to Numerical Analysis*, S. K. Zaremba, ed., Academic Press, London, 1972, pp. 249–255.
Naylor, T., *Computer Simulation Techniques*. Wiley, New York, 1966.
Ripley, B., *Stochastic Simulation*. Wiley, New York, 1986.
von Neumann, J., "Various Techniques Used in Connection with Random Digits, 'Monte Carlo Method,'" *U.S. National Bureau of Standards Applied Mathematics Series*, No. 12, 36–38, 1951.

Generating Discrete Random Variables

4.1 The Inverse Transform Method

Suppose we want to generate the value of a discrete random variable X having probability mass function

$$P\{X = x_j\} = p_j, \quad j = 0, 1, \ldots, \sum_j p_j = 1$$

To accomplish this, we generate a random number U — that is, U is uniformly distributed over (0,1) — and set

$$X = \begin{cases} x_0 & \text{if } U < p_0 \\ x_1 & \text{if } p_0 \le U < p_0 + p_1 \\ \vdots & \\ x_j & \text{if } \sum_{i=1}^{j-1} p_i \le U < \sum_{i=1}^{j} p_i \\ \vdots & \end{cases}$$

Since, for $0 < a < b < 1$, $P\{a \le U < b\} = b - a$, we have that

$$P\{X = x_j\} = P\left\{ \sum_{i=1}^{j-1} p_i \le U < \sum_{i=1}^{j} p_i \right\} = p_j$$

and so X has the desired distribution.

narks

The preceding can be written algorithmically as

Generate a random number U
If $U < p_0$ set $X = x_0$ and stop
If $U < p_0 + p_1$ set $X = x_1$ and stop
If $U < p_0 + p_1 + p_2$ set $X = x_2$ and stop
$\vdots$

If the $x_i, i \geq 0$, are ordered so that $x_0 < x_1 < x_2 < \cdots$ and if we let F denote the distribution function of X, then $F(x_k) = \sum_{i=0}^{k} p_i$ and so

$$X \text{ will equal } x_j \quad \text{if} \quad F(x_{j-1}) \leq U < F(x_j)$$

her words, after generating a random number U we determine the value of finding the interval $[F(x_{j-1}), F(x_j))$ in which U lies [or, equivalently, by ng the inverse of $F(U)$]. It is for this reason that the above is called the discrete se transform method for generating X. □

ıe amount of time it takes to generate a discrete random variable by the above od is proportional to the number of intervals one must search. For this reason ;ometimes worthwhile to consider the possible values x_j of X in decreasing of the p_j.

mple 4a If we wanted to simulate a random variable X such that

$= 0.20, \quad p_2 = 0.15, \quad p_3 = 0.25, \quad p_4 = 0.40 \quad$ where $p_j = P\{X = j\}$

we could generate U and do the following:

If $U < 0.20$ set $X = 1$ and stop
If $U < 0.35$ set $X = 2$ and stop
If $U < 0.60$ set $X = 3$ and stop
Otherwise set $X = 4$

ver, a more efficient procedure is the following:

If $U < 0.40$ set $X = 4$ and stop
If $U < 0.65$ set $X = 3$ and stop
If $U < 0.85$ set $X = 1$ and stop
Otherwise set $X = 2$ □

One case where it is not necessary to search for the appropriate interval in which the random number lies is when the desired random variable is the discrete uniform random variable. That is, suppose we want to generate the value of X which is equally likely to take on any of the values $1, \ldots, n$. That is, $P\{X = j\} = 1/n, j = 1, \ldots, n$. Using the preceding results it follows that we can accomplish this by generating U and then setting

$$X = j \quad \text{if} \quad \frac{j-1}{n} \leq U < \frac{j}{n}$$

Therefore, X will equal j if $j - 1 \leq nU < j$; or, in other words,

$$X = \text{Int}(nU) + 1$$

where $\text{Int}(x)$ — sometimes written as $[x]$ — is the integer part of x (i.e., the largest integer less than or equal to x).

Discrete uniform random variables are quite important in simulation, as is indicated in the following two examples.

Example 4b: Generating a Random Permutation Suppose we are interested in generating a permutation of the numbers $1, 2, \ldots, n$ which is such that all $n!$ possible orderings are equally likely. The following algorithm will accomplish this by first choosing one of the numbers $1, \ldots, n$ at random and then putting that number in position n; it then chooses at random one of the remaining $n - 1$ numbers and puts that number in position $n - 1$; it then chooses at random one of the remaining $n - 2$ numbers and puts it in position $n - 2$; and so on (where choosing a number at random means that each of the remaining numbers is equally likely to be chosen). However, so that we do not have to consider exactly which of the numbers remain to be positioned, it is convenient and efficient to keep the numbers in an ordered list and then randomly choose the position of the number rather than the number itself. That is, starting with any initial ordering $P_1, P_2, \ldots, P_n$ we pick one of the positions $1, \ldots, n$ at random and then interchange the number in that position with the one in position n. Now we randomly choose one of the positions $1, \ldots, n-1$ and interchange the number in this position with the one in position $n - 1$, and so on.

Recalling that $\text{Int}(kU) + 1$ will be equally likely to take on any of the values $1, 2, \ldots, k$, we see that the above algorithm for generating a random permutation can be written as follows:

STEP 1: Let $P_1, P_2, \ldots, P_n$ be any permutation of $1, 2, \ldots, n$ (e.g., we can choose $P_j = j, j = 1, \ldots, n$).
STEP 2: Set $k = n$.
STEP 3: Generate a random number U and let $I = \text{Int}(kU) + 1$.
STEP 4: Interchange the values of P_I and P_k.
STEP 5: Let $k = k - 1$ and if $k > 1$ go to Step 3.
STEP 6: $P_1, \ldots, P_n$ is the desired random permutation.

For instance, suppose $n = 4$ and the initial permutation is 1, 2, 3, 4. If the first value of I (which is equally likely to be either 1, 2, 3, or 4) is $I = 3$, then the elements in positions 3 and 4 are interchanged and so the new permutation is 1, 2, 4, 3. If the next value of I is $I = 2$, then the elements in positions 2 and 3 are interchanged and so the new permutation is 1, 4, 2, 3. If the final value of I is $I = 2$, then the final permutation is 1, 4, 2, 3, and this is the value of the random permutation. □

One very important property of the preceding algorithm is that it can also be used to generate a random subset, say of size r, of the integers $1, \ldots, n$. Namely, just follow the algorithm until the positions $n, n-1, \ldots, n-r+1$ are filled. The elements in these positions constitute the random subset. (In doing this we can always suppose that $r \leq n/2$; for if $r > n/2$ then we could choose a random subset of size $n - r$ and let the elements not in this subset be the random subset of size r.)

It should be noted that the ability to generate a random subset is particularly important in medical trials. For instance, suppose that a medical center is planning to test a new drug designed to reduce its user's blood cholesterol level. To test its effectiveness, the medical center has recruited 1000 volunteers to be subjects in the test. To take into account the possibility that the subjects' blood cholesterol levels may be affected by factors external to the test (such as changing weather conditions), it has been decided to split the volunteers into two groups of size 500—a *treatment* group that will be given the drug and a *control* that will be given a placebo. Both the volunteers and the administrators of the drug will not be told who is in each group (such a test is called *double-blind*). It remains to determine which of the volunteers should be chosen to constitute the treatment group. Clearly, one would want the treatment group and the control group to be as similar as possible in all respects with the exception that members in the first group are to receive the drug while those in the other group receive a placebo, for then it would be possible to conclude that any difference in response between the groups is indeed due to the drug. There is general agreement that the best way to accomplish this is to choose the 500 volunteers to be in the treatment group in a completely random fashion. That is, the choice should be made so that each of the $\binom{1000}{500}$ subsets of 500 volunteers is equally likely to constitute the set of volunteers.

Remark Another way to generate a random permutation is to generate n random numbers $U_1, \ldots, U_n$, order them, and then use the indices of the successive values as the random permutation. For instance, if $n = 4$, and $U_1 = 0.4, U_2 = 0.1, U_3 = 0.8, U_4 = 0.7$, then, because $U_2 < U_1 < U_4 < U_3$, the random permutation is 2, 1, 4, 3. The difficulty with this approach, however, is that ordering the random numbers typically requires on the order of $n \log(n)$ comparisons. □

Example 4c: Calculating Averages Suppose we want to approximate $\bar{a} = \sum_{i=1}^{n} a(i)/n$, where n is large and the values $a(i), i = 1, \ldots, n$, are complicated and not easily calculated. One way to accomplish this is to note that if X

is a discrete uniform random variable over the integers $1, \ldots, n$, then the r variable $a(X)$ has a mean given by

$$E[a(X)] = \sum_{i=1}^{n} a(i)P\{X = i\} = \sum_{i=1}^{n} \frac{a(i)}{n} = \bar{a}$$

Hence, if we generate k discrete uniform random variables $X_i, i = 1, \ldots,$ generating k random numbers U_i and setting $X_i = \text{Int}(nU_i) + 1$—then the k random variables $a(X_i)$ will have mean $\bar{a}$, and so by the strong law numbers it follows that when k is large (though much smaller than n) the of these values should approximately equal $\bar{a}$. Hence, we can approxima using

$$\bar{a} \approx \sum_{i=1}^{k} \frac{a(X_i)}{k}$$

Another random variable that can be generated without needing to sea the relevant interval in which the random number falls is the geometric.

Example 4d Recall that X is said to be a geometric random variabl parameter p if

$$P\{X = i\} = pq^{i-1}, \quad i \geq 1, \quad \text{where } q = 1 - p$$

X can be thought of as representing the time of the first success when indep trials, each of which is a success with probability p, are performed. Since

$$\begin{aligned} \sum_{i=1}^{j-1} P\{X = i\} &= 1 - P\{X > j-1\} \\ &= 1 - P\{\text{first } j-1 \text{ trials are all failures}\} \\ &= 1 - q^{j-1}, \quad j \geq 1 \end{aligned}$$

we can generate the value of X by generating a random number U and set equal to that value j for which

$$1 - q^{j-1} \leq U < 1 - q^j$$

or, equivalently, for which

$$q^j < 1 - U \leq q^{j-1}$$

Markov Chain Monte Carlo Methods

Introduction

It is, in general, very difficult to simulate the value of a random vector $\mathbf{X}$ whose component random variables are dependent. In this chapter we present a powerful approach for generating a vector whose distribution is approximately that of $\mathbf{X}$. This approach, called the Markov chain Monte Carlo method, has the added significance of only requiring that the mass (or density) function of $\mathbf{X}$ be specified up to a multiplicative constant, and this, we will see, is of great importance in applications.

In Section 10.1 we introduce and give the needed results about Markov chains. In Section 10.2 we present the Hastings–Metropolis algorithm for constructing a Markov chain having a specified probability mass function as its limiting distribution. A special case of this algorithm, referred to as the Gibbs sampler, is studied in Section 10.3. The Gibbs sampler is probably the most widely used Markov chain Monte Carlo method. An application of the preceding methods to deterministic optimization problems, known as simulated annealing, is presented in Section 10.4. In Section 10.5 we present the sampling importance resampling (SIR) technique. While not strictly a Markov chain Monte Carlo algorithm, it also results in approximately simulating a random vector whose mass function is specified up to a multiplicative constant.

10.1 Markov Chains

Consider a collection of random variables $X_0, X_1, \ldots$. Interpret X_n as the "state of the system at time n," and suppose that the set of possible values of the X_n — that is, the possible states of the system — is the set $1, \ldots, N$. If there exists a set of numbers $P_{ij}, i, j = 1, \ldots, N$, such that whenever the process is in state i then, independent of the past states, the probability that the next state is j is P_{ij}, then we say that the collection $\{X_n, n \geq 0\}$ constitutes a *Markov chain* having transition

robabilities P_{ij}, $i, j = 1, \ldots, N$. Since the process must be in some state after it aves states i, these transition probabilities satisfy

$$\sum_{j=1}^{N} P_{ij} = 1, \quad i = 1, \ldots, N$$

A Markov chain is said to be irreducible if for each pair of states i and j there a positive probability, starting in state i, that the process will ever enter state . For an irreducible Markov chain, let π_j denote the long-run proportion of time at the process is in state j. (It can be shown that π_j exists and is constant, with robability 1, independent of the initial state.) The quantities π_j, $j = 1, \ldots, N$, an be shown to be the unique solution of the following set of linear equations:

$$\begin{aligned} \pi_j &= \sum_{i=1}^{N} \pi_i P_{ij}, \quad j = 1, \ldots, N \\ \sum_{j=1}^{N} \pi_j &= 1 \end{aligned} \tag{10.1}$$

emark The set of equations (10.1) have a heuristic interpretation. Since π_i is he proportion of time that the Markov chain is in state i and since each transition ut of state i is into state j with probability P_{ij}, it follows that $\pi_i P_{ij}$ is the proportion f time in which the Markov chain has just entered state j from state i. Hence, the op part of Equation (10.1) states the intuitively clear fact that the proportion of ime in which the Markov chain has just entered state j is equal to the sum, over all tates i, of the proportion of time in which it has just entered state j from state i. he bottom part of Equation (10.1) says, of course, that summing the proportion f time in which the chain is in state j, over all j, must equal 1. □

he $\{\pi_j\}$ are often called the *stationary probabilities* of the Markov chain. For f the initial state of the Markov chain is distributed according to the $\{\pi_j\}$ then $^{D}\{X_n = j\} = \pi_j$, for all n and j (see Exercise 1).

An important property of Markov chains is that for any function h on the state pace, with probability 1,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} h(X_i) = \sum_{j=1}^{N} \pi_j h(j) \tag{10.2}$$

he preceding follows since if $p_j(n)$ is the proportion of time that the chain is in state j between times $1, \ldots, n$ then

$$\frac{1}{n} \sum_{i=1}^{n} h(X_i) = \sum_{j=1}^{N} h(j) P_j(n) \to \sum_{j=1}^{N} h(j)\pi_j$$

The quantity π_j can often be interpreted as the limiting probability that the chain is in state j. To make precise the conditions under which it has this interpretation, we first need the definition of an aperiodic Markov chain.

Definition *An irreducible Markov chain is said to be aperiodic if for some $n \geq 0$ and some state j,*

$$P\{X_n = j | X_0 = j\} > 0 \qquad \text{and} \qquad P\{X_{n+1} = j | X_0 = j\} > 0$$

It can be shown that if the Markov chain is irreducible and aperiodic then

$$\pi_j = \lim_{n\to\infty} P\{X_n = j\}, \quad j = 1, \ldots, N$$

There is sometimes an easier way than solving the set of equations (10.1) of finding the stationary probabilities. Suppose one can find positive numbers x_j, $j = 1, \ldots, N$ such that

$$x_i P_{ij} = x_j P_{ji}, \quad \text{for } i \neq j, \qquad \sum_{j=1}^{N} x_j = 1$$

Then summing the preceding equations over all states i yields

$$\sum_{i=1}^{N} x_i P_{ij} = x_j \sum_{i=1}^{N} P_{ji} = x_j$$

which, since $\{\pi_j, j = 1, \ldots, N\}$ is the unique solution of (10.1), implies that

$$\pi_j = x_j$$

When $\pi_i P_{ij} = \pi_j P_{ji}$, for all $i \neq j$, the Markov chain is said to be *time reversible*, because it can be shown, under this condition, that if the initial state is chosen according to the probabilities $\{\pi_j\}$, then starting at any time the sequence of states going backwards in time will also be a Markov chain with transition probabilities P_{ij}.

Suppose now that we want to generate the value of a random variable X having probability mass function $P\{X = j\} = p_j$, $j = 1, \ldots, N$. If we could generate an irreducible aperiodic Markov chain with limiting probabilities P_j, $J = 1, \ldots, N$, then we would be able to approximately generate such a random variable by running the chain for n steps to obtain the value of X_n, where n is large. In addition, if our objective was to generate many random variables distributed according to p_j, $j = 1, \ldots, N$, so as to be able to estimate $E[h(X)] = \sum_{j=1}^{N} h(j) p_j$, then we could also estimate this quantity by using the estimator $\frac{1}{n} \sum_{i=1}^{n} h(X_i)$. However, since the early states of the Markov chain can be strongly influenced by the initial

which shows that the conditional distribution is that of an exponential with rate $\lambda - \mu$ that is conditioned to be less than a.

Using this result, we can estimate β by letting the initial state $(x_1, x_2, x_3, x_4, x_5)$ be any five positive numbers that sum to 15. Now randomly choose two elements from the set 1, 2, 3, 4, 5; say $I = 2$ and $J = 5$ are chosen. Then the conditional distribution of X_2, X_5 given the other values is the conditional distribution of two independent exponentials with means 2 and 5, given that their sum is $15 - x_1 - x_3 - x_4$. But, by the preceding, the values of X_2 and X_5 can be obtained by generating the value of an exponential with rate $\frac{1}{2} - \frac{1}{5} = \frac{3}{10}$ that is conditioned to be less than $15 - x_1 - x_3 - x_4$, then setting x_2 equal to that value and resetting x_5 to make $\sum_{i=1}^{5} x_i = 15$. This process should be continually repeated, and the proportion of state vectors $\mathbf{x}$ having $\prod_{i=1}^{5} x_i > 120$ is the estimate of β. □

Example 10j Suppose that n independent trials are performed; each of which results in one of the outcomes $1, 2, \ldots, r$, with respective probabilities $p_1, p_2, \ldots, p_r$, $\sum_{i=1}^{r} p_i = 1$, and let X_i denote the number of trials that result in outcome i. The random variables $X_1, \ldots, X_r$, whose joint distribution is called the multinomial distribution, were introduced in Example 4g where it was shown how they can be simulated. Now suppose $n > r$, and that we want to simulate $X_1, \ldots, X_r$ conditional on the event that they are all positive. That is, we want to simulate the result of the trials conditional on the event that each outcome occurs at least once. How can this be efficiently accomplished when this conditioning event has a very small probability?

Solution To begin, it should be noted that it would be wrong to suppose that we could just generate the result of $n - r$ of these trials, and then let X_i equal 1 plus the number of these $n - r$ trials that result in outcome i. (That is, attempting to put aside the r trials in which all outcomes occur once, and then simulating the remaining $n - r$ trials does not work.) To see why, let $n = 4$ and $r = 2$. Then, under the putting aside method, the probability that exactly 2 of the trials would result in outcome 1 is $2p(1 - p)$, where $p = p_1$. However, for the multinomial random variables X_1, X_2

$$\begin{aligned} P\{X_1 = 2 | X_1 > 0, X_2 > 0\} &= \frac{P\{X_1 = 2\}}{P\{X_1 > 0, X_2 > 0\}} \\ &= \frac{P\{X_1 = 2\}}{1 - P\{X_1 = 4\} - P\{X_2 = 4\}} \\ &= \frac{\binom{4}{2} p^2 (1-p)^2}{1 - p^4 - (1-p)^4} \end{aligned}$$

As the preceding is not equal to $2p(1 - p)$ (try $p = 1/2$), the method does not work.

We can use the Gibbs sampler to generate a Markov chain having the appropriate limiting probabilities. Let the initial state be any arbitrary vector of r positive integers whose sum is n, and let the states change in the following manner. Whenever the state is $x_1, \ldots, x_r$, generate the next state by first randomly choosing two of the indices from $1, \ldots, r$. If i and j are chosen, let $s = x_i + x_j$, and simulate X_i and X_j from their conditional distribution given that $X_k = x_k$, $k \neq i, j$. Because conditional on $X_k = x_k$, $k \neq i, j$ there are a total of s trials that result in either outcome i or j, it follows that the number of these trials that result in outcome i is distributed as a binomial random variable with parameters $(s, \frac{p_i}{p_i + p_j})$ that is conditioned to be one of the values $1, \ldots, s - 1$. Consequently, the discrete inverse transform method can be used to simulate such a random variable; if its value is v, then the next state is the same as the previous one with the exception that the new values of x_i and x_j are v and $s - v$. Continuing on in this manner results in a sequence of states whose limiting distribution is that of the multinomial conditional on the event that all outcomes occur at least once. □

Remarks

1. The same argument can be used to verify that we obtain the appropriate limiting mass function when we consider the coordinates in sequence and apply the Gibbs sampler (as in Example 10i), or when we use it via conditioning on less than all but one of the values (as in Example 10j). These results are proven by noticing that if one chooses the initial state according to the mass function f, then, in either case, the next state also has mass function f. But this shows that f satisfies the equations (10.1), implying by uniqueness that f is the limiting mass function.
2. Suppose you are using the Gibbs sampler to estimate $E[X_i]$ in a situation where the conditional means $E[X_i | X_j, j \neq i]$ are easily computed. Then, rather than using the average of the successive values of X_i as the estimator, it is usually better to use the average of the conditional expectations. That is, if the present state is $\mathbf{x}$, then take $E[X_i | X_j = x_j, j \neq i]$ rather than x_i as the estimate from that iteration. Similarly, if you are trying to estimate $P\{X_i = x\}$, and $P\{X_i = x | X_j, j \neq i\}$ is easily computed, then the average of these quantities is usually a better estimator than is the proportion of time in which the ith component of the state vector equals x.
3. The Gibbs sampler shows that knowledge of all the conditional distributions of X_i given the values of the other X_j, $j \neq i$, determines the joint distribution of $\boldsymbol{X}$. □

10.4 Simulated Annealing

Let $\mathcal{A}$ be a finite set of vectors and let $V(\mathbf{x})$ be a nonnegative function defined on $\mathbf{x} \in \mathcal{A}$, and suppose that we are interested in finding its maximal value and at least one argument at which the maximal value is attained. That is, letting

$$V^* = \max_{\mathbf{x} \in \mathcal{A}} V(\mathbf{x})$$

and

$$\mathcal{M} = \{\mathbf{x} \in \mathcal{A}: V(\mathbf{x}) = V^*\}$$

we are interested in finding V^* as well as an element in $\mathcal{M}$. We will now show how this can be accomplished by using the methods of this chapter.

To begin, let $\lambda > 0$ and consider the following probability mass function on the set of values in $\mathcal{A}$:

$$p_\lambda(\mathbf{x}) = \frac{e^{\lambda V(\mathbf{x})}}{\sum_{\mathbf{x}\in\mathcal{A}} e^{\lambda V(\mathbf{x})}}$$

By multiplying the numerator and denominator of the preceding by $e^{-\lambda V^*}$, and letting $|\mathcal{M}|$ denote the number of elements in $\mathcal{M}$, we see that

$$p_\lambda(\mathbf{x}) = \frac{e^{\lambda(V(\mathbf{x})-V^*)}}{|\mathcal{M}| + \sum_{\mathbf{x}\notin\mathcal{M}} e^{\lambda(V(\mathbf{x})-V^*)}}$$

However, since $V(\mathbf{x}) - V^* < 0$ for $\mathbf{x} \notin \mathcal{M}$, we obtain that as $\lambda \to \infty$,

$$p_\lambda(\mathbf{x}) \to \frac{\delta(\mathbf{x}, \mathcal{M})}{|\mathcal{M}|}$$

where $\delta(\mathbf{x}, \mathcal{M}) = 1$ if $\mathbf{x} \in \mathcal{M}$ and is 0 otherwise.

Hence, if we let λ be large and generate a Markov chain whose limiting distribution is $p_\lambda(\mathbf{x})$, then most of the mass of this limiting distribution will be concentrated on points in $\mathcal{M}$. An approach that is often useful in defining such a chain is to introduce the concept of neighboring vectors and then use a Hastings–Metropolis algorithm. For instance, we could say that the two vectors $\mathbf{x} \in \mathcal{A}$ and $\mathbf{y} \in \mathcal{A}$ are neighbors if they differ in only a single coordinate or if one can be obtained from the other by interchanging two of its components. We could then let the target next state from $\mathbf{x}$ be equally likely to be any of its neighbors, and if the neighbor $\mathbf{y}$ is chosen, then the next state becomes $\mathbf{y}$ with probability

$$\min\left\{1, \frac{e^{\lambda V(\mathbf{y})}/|N(\mathbf{y})|}{e^{\lambda V(\mathbf{x})}/|N(\mathbf{x})|}\right\}$$

or remains $\mathbf{x}$ otherwise, where $|N(\mathbf{z})|$ is the number of neighbors of $\mathbf{z}$. If each vector has the same number of neighbors (and if not already so, this can almost always be arranged by increasing the state space and letting the V value of any new state equal 0), then when the state is $\mathbf{x}$, one of its neighbors, say $\mathbf{y}$, is randomly chosen; if $V(\mathbf{y}) \geq V(\mathbf{x})$, then the chain moves to state $\mathbf{y}$, and if $V(\mathbf{y}) < V(\mathbf{x})$, then the chain moves to state $\mathbf{y}$ with probability $\exp\{\lambda(V(\mathbf{y}) - V(\mathbf{x}))\}$ or remains in state $\mathbf{x}$ otherwise.

One weakness with the preceding algorithm is that because λ was chosen to be large, if the chain enters a state $\mathbf{x}$ whose V value is greater than that of each of its neighbors, then it might take a long time for the chain to move to a different state. That is, whereas a large value of λ is needed for the limiting distribution to put most of its weight on points in $\mathcal{M}$, such a value typically requires a very large number of transitions before the limiting distribution is approached. A second weakness is that since there are only a finite number of possible values of $\mathbf{x}$, the whole concept of convergence seems meaningless since we could always, in theory, just try each of the possible values and so obtain convergence in a finite number of steps. Thus, rather than considering the preceding from a strictly mathematical point of view, it makes more sense to regard it as a heuristic approach, and in doing so it has been found to be useful to allow the value of λ to change with time.

A popular variation of the preceding, known as *simulated annealing*, operates as follows. If the nth state of the Markov chain is $\mathbf{x}$, then a neighboring value is randomly selected. If it is $\mathbf{y}$, then the next state is either $\mathbf{y}$ with probability

$$\min\left\{1, \frac{\exp\{\lambda_n V(\mathbf{y})\}/|N(\mathbf{y})|}{\exp\{\lambda_n V(\mathbf{x})\}/|N(\mathbf{x})|}\right\}$$

or it remains $\mathbf{x}$, where λ_n, $n \geq 1$, is a prescribed set of values that start out small (thus resulting in a large number of changes in state) and then grow.

A computationally useful choice of λ_n (and a choice that mathematically results in convergence) is to let $\lambda_n = C\log(1 + n)$, where $C > 0$ is any fixed positive constant (see Besag *et al.*, 1995; Diaconis and Holmes, 1995). If we then generate m successive states $X_1, \ldots, X_m$, we can then estimate V^* by $\max_{i=1,\ldots,m} V(X_i)$, and if the maximum occurs at X_{i^*} then this is taken as an estimated point in $\mathcal{M}$.

Example 10k: The Traveling Salesman Problem One version of the traveling salesman problem is for the salesman to start at city 0 and then sequentially visit all of the cities $1, \ldots, r$. A possible choice is then a permutation $x_1, \ldots, x_r$ of $1, \ldots, r$ with the interpretation that from 0 the salesman goes to city x_1, then to x_2, and so on. If we suppose that a nonnegative reward $v(i, j)$ is earned whenever the salesman goes directly from city i to city j, then the return of the choice $\mathbf{x} = (x_1, \ldots, x_r)$ is

$$V(\mathbf{x}) = \sum_{i=1}^{r} v(x_{i-1}, x_i) \quad \text{where } x_0 = 0$$

By letting two permutations be neighbors if one results from an interchange of two of the coordinates of the other, we can use simulated annealing to approximate the best path. Namely, start with any permutation $\mathbf{x}$ and let $X_0 = \mathbf{x}$. Now, once the nth state (that is, permutation) has been determined, $n \geq 0$, then generate one of its neighbors at random [by choosing I, J equally likely to be any of the $\binom{r}{2}$ values $i \neq j, i, j = 1, \ldots, r$ and then interchanging the values of the Ith and

Jth elements of X_n]. Let the generated neighbor be $\mathbf{y}$. Then if $V(\mathbf{y}) \geq V(X_n)$, set $X_{n+1} = \mathbf{y}$. Otherwise, set $X_{n+1} = \mathbf{y}$ with probability $(1+n)^{(V(\mathbf{y})-V(\mathbf{X}_n))}$, or set it equal to X_n otherwise. [Note that we are using $\lambda_n = \log(1+n)$.] □

10.5 The Sampling Importance Resampling Algorithm

The sampling importance resampling, or SIR, algorithm is a method for generating a random vector $\boldsymbol{X}$ whose mass function

$$f(\boldsymbol{x}) = C_1 f_o(\boldsymbol{x})$$

is specified up to a multiplicative constant by simulating a Markov chain whose limiting probabilities are given by a mass function

$$g(\boldsymbol{x}) = C_2 g_o(\boldsymbol{x})$$

that is also specified up to a multiplicative constant. It is similar to the acceptance–rejection technique, where one starts by generating the value of a random vector $\boldsymbol{Y}$ with density g and then, if $\boldsymbol{Y} = \boldsymbol{y}$, accepting this value with probability $f(\mathbf{y})/cg(\mathbf{y})$, where c is a constant chosen so that $f(\mathbf{x})/cg(\mathbf{x}) \leq 1$, for all $\boldsymbol{x}$. If the value is not accepted, then the process begins anew, and the eventually accepted value $\boldsymbol{X}$ has density f. However, as f and g are no longer totally specified, this approach is not available.

The SIR approach starts by generating m successive states of a Markov chain whose limiting probability mass function is g. Let these state values be denoted as $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m$. Now, define the "weights" $w_i, i = 1, \ldots, m$, by

$$w_i = \frac{f_o(\boldsymbol{y}_i)}{g_o(\boldsymbol{y}_i)}$$

and generate a random vector $\boldsymbol{X}$ such that

$$P\{\boldsymbol{X} = \boldsymbol{y}_j\} = \frac{w_j}{\sum_{i=1}^m w_i}, \quad j = 1, \ldots, m$$

We will show that when m is large, the random vector $\boldsymbol{X}$ has a mass function approximately equal to f.

Proposition *The distribution of the vector $\boldsymbol{X}$ obtained by the SIR method converges as $m \to \infty$ to f.*

Proof Let $\boldsymbol{Y}_i, i = 1, \ldots, m$, denote the m random vectors generated by the Markov chain whose limiting mass function is g, and let $W_i = f_o(\boldsymbol{Y}_i)/g_o(\boldsymbol{Y}_i)$ denote their weights. For a fixed set of vectors $\mathcal{A}$, let $I_i = 1$ if $\boldsymbol{Y}_i \in \mathcal{A}$ and let it equal 0 otherwise. Then

$$P\{\boldsymbol{X} \in \mathcal{A} | \boldsymbol{Y}_i, i = 1, \ldots, m\} = \frac{\sum_{i=1}^m I_i W_i}{\sum_{i=1}^m W_i} \tag{10.5}$$

Now, by the Markov chain result of Equation (10.2), we see that as $m \to \infty$,

$$\sum_{i=1}^m I_i W_i/m \to E_g[IW] = E_g[IW|I=1]P_g\{I=1\} = E_g[W|\boldsymbol{Y} \in \mathcal{A}]P_g\{\boldsymbol{Y} \in \mathcal{A}\}$$

and

$$\sum_{i=1}^m W_i/m \to E_g[W] = E_g[f_o(\boldsymbol{Y})/g_o(\boldsymbol{Y})] = \int \frac{f_o(\boldsymbol{y})}{g_o(\boldsymbol{y})} g(\boldsymbol{y})d\boldsymbol{y} = C_2/C_1$$

Hence, dividing numerator and denominator of (10.5) by m shows that

$$P\{\boldsymbol{X} \in \mathcal{A} | \boldsymbol{Y}_i, i = 1, \ldots, m\} \to \frac{C_1}{C_2} E_g[W|\boldsymbol{Y} \in \mathcal{A}]P_g\{\boldsymbol{Y} \in \mathcal{A}\}$$

But,

$$\begin{aligned}\frac{C_1}{C_2} E_g[W|\boldsymbol{Y} \in \mathcal{A}]P_g\{\boldsymbol{Y} \in \mathcal{A}\} &= \frac{C_1}{C_2} E_g\left[\frac{f_o(\boldsymbol{Y})}{g_o(\boldsymbol{Y})} | \boldsymbol{Y} \in \mathcal{A}\right] P_g\{\boldsymbol{Y} \in \mathcal{A}\} \\ &= \int_{\boldsymbol{y} \in \mathcal{A}} \frac{f(\boldsymbol{y})}{g(\boldsymbol{y})} g(\boldsymbol{y})d\boldsymbol{y} \\ &= \int_{\boldsymbol{y} \in \mathcal{A}} f(\boldsymbol{y})d\boldsymbol{y}\end{aligned}$$

Hence, as $m \to \infty$,

$$P\{\boldsymbol{X} \in \mathcal{A} | \boldsymbol{Y}_i, i = 1, \ldots, m\} \to \int_{\boldsymbol{y} \in \mathcal{A}} f(\boldsymbol{y})d\boldsymbol{y}$$

which implies, by a mathematical result known as Lebesgue's dominated convergence theorem, that

$$P\{\boldsymbol{X} \in \mathcal{A}\} = E[P\{\boldsymbol{X} \in \mathcal{A} | \boldsymbol{Y}_i, i = 1, \ldots, m\}] \to \int_{\boldsymbol{y} \in \mathcal{A}} f(\boldsymbol{y})d\boldsymbol{y}$$

and the result is proved. □