Atomic Scale Simulations

Sampling Methods

Today we discuss how to sample probability distributions.

Discrete Distributions

Any discrete distribution can be sampled by constructing the cumulant. Suppose we want to sample the integer k where $1 \leq k \leq N$ with probability p_k. An example might be to sample the roll of several dice. A possible sampling algorithm for any discrete distribution is as follows:

1. Before the MC sampling begins make the table: $F_k = F_{k-1} + p_k$ with $F_0 = 0$. Since p_k is a probability we must have $0 \leq p_k \leq 1$ and F_k is monotonically increasing with $F_N = 1$.

2. Sample u, a uniform rn in $(0, 1)$. Find the unique k that satisfies $F_{k-1} < u < F_k$. Unless p_k has a simple form, one must solve this inequality using the bisection method.

The operation involved in step 2 will take in general $log_2(N)$ steps. If this is time-consuming for large N one can use an improved algorithm (see Knuth) which involves a more elaborate table set up (of order $N \log_2(N)$ steps but which takes only a few operations to sample, independent of the value of N. Essentially we sample uniformly in the range of 1 to N but correct for the fact that P_k is non-uniform by correcting the sampling.

The "Walker" aliases method is as follows:

1. Set up tables Y_k and J_k as follows. Form the list of pairs $(p_1, 1), (p_2, 2) \ldots$ and sort them in ascending order of p_i obtaining the pairs $(q_1, a_1), (q_2, a_2) \ldots$ Set $n = N$ and repeat the following procedure until $n = 0$. Set $Y_{a_1} = Nq_1$ and $J_{a_1} = a_n$. Delete the pairs (q_1, a_1) and (q_n, a_n) from the list. Insert $(q_n - (1/N - q_1), a_n)$ into the list in the proper place. Decrease n by 1.
2. Sample u, a uniform rn in $(0, 1)$ and divide $1 + N * u = k + x$ with k and integer and $0 < x < 1$.
3. If $x < Y_k$ our sample is k
4. Otherwise our sample is J_k.

Continuous Distributions

To sample a distribution for a continuous variable x, use the method above by letting $N \to \infty$. This is called the *mapping method*. Suppose we want to sample $p(x)dx$.

1. Analytically or numerically form the cumulative distribution:
$$F(x) = \int_{-\infty}^{x} dyp(y) \tag{1}$$
2. Sample a uniform rn $u \in (0, 1)$.
3. Solve the equation:
$$F(x) = u \tag{2}$$
for x. Then x is sampled from $p(x)dx$.

This is a completely general method. We can generalize this further if u is sampled from something other than the uniform distribution. If $g(u) = x$ then the probabilities of x and u are related by the equation:
$$p(x)dx = p(u)du. \tag{3}$$

Hence:

$$p(x)|dg/du| = p(u) \tag{4}$$

The inverse cumulant is a solution to this differential equation.

Two common examples of using the mapping method:

1. $p(x) = (a+1)x^a$ for $0 < x < 1$ with $-1 < a$. Then $F(x) = x^{a+1}$ and hence $x = u^{1/(a+1)}$. Although the form is simple, in general it requires computing two special functions, the log and exponential.
2. $p(x) = \exp(-x)$ for $0 < x$. Then $F(x) = 1 - \exp(-x)$ and $x = -\ln(u)$.

The mapping method is a straightforward procedure, but it may not be the most efficient or simple. The problem is that the inversion may be awkward to carry out, although with a tabulation it can always be made quite fast.

The Rejection Method

Another general method is the *rejection method.* The procedure is as follows:

1. Sample x from $q(x)dx$. ($q(x)$ is some pdf).
2. Accept the value x with probability $r(x)$. To do this we compare $r(x)$ to u (a udrn) and accept if $r(x) < u$. Otherwise we go back to 1.

What is the distribution of accepted x's? It is proportional to $q(x)r(x)dx$. Hence to sample $p(x)$, we must choose $r(x) = cp(x)/q(x)$ with c some constant. In fact c must be chosen so that $r(x) \leq 1$; so that $r(x)$ is a probability. In fact $1/c$ is the average number of times that step 1 is executed. The optimal value of c is $c = min(q(x)/p(x))$. [Clearly the optimal value of $q(x)$ is if we could sample $p(x)$ directly.]

In general the efficiency of a sampling method is simply the rate at which it can deliver random numbers. For the mapping method it is $1/CPUtime$ for a single rn. For the rejection method it is $c/CPUtime$ for a single trial rn. One can use a approximate mapping to make a q close to p, and then a final rejection step (with a c close to one) to sample the exact distribution. The rejection method is another illustration of the Monte Carlo rule: *under sampling can be very bad.* One can get very inefficient sampling because of a single region where $q(x) << p(x)$.

Composition Methods

Mappings can be thought at unary operations on a single random number. What if if do something on with 2 or more numbers, such as add or multiply them together. These are composition methods. First consider adding together several random numbers. Suppose x is sampled from $p(x)$ and y from $q(y)dy$. Then the sum $z = x + y$ is sampled from the convolution:

$$p(z)dz = \int dxp(x)q(z-x) \tag{5}$$

The characteristic function is the Fourier transform of p: $G(k) = \int dxp(x)e^{ikx}$. By the convolution property of Fourier transforms: $G_z(k) = G_x(k)G_y(k)$. By inverse Fourier transforming one can find the distribution of z. In fact, this is the way you can prove the central limit theorem. Multiplication of random numbers can be treated by using logarithms (to convert to additions). Another interesting operation is to take the maximum of several uniform random numbers.

Sampling the normal distribution

This is a common but moderately difficult function to sample. Here are a few of the ways.

1. Sample N uniformly distributed rns u_i in (0,1) and set $\eta = \sqrt{\frac{12}{N}}[\Sigma_{k=1}^{N}(u_k - 1/2)]$. Then η has mean zero and unit variance and for large N will approach a normal random number. In fact N can be fairly small. (For $N = 12$ we don't even need to do the division.) The disadvantage is that one is loosing randomness in the pseudo-random number stream by grouping things by N's so the correlation between successive ndrn's may be much higher than between the udrn's. Also it is impossible for the magnitude of η to exceed $\sqrt{3/N}$ so large excursions are not only rare but impossible.

2. The direct mapping method would be to invert the cumulative distribution, that would involve computation of the inverse error function. You can find algorithms for doing that in the literature.

3. There is a much simpler procedure which involves generating a pair of normal distributed random numbers $(\eta_1, \eta_2) = (rcos(\theta), rsin(\theta))$ where $\theta = 2\pi u_1$ and $r = \sqrt{-2\ln(u_2)}$. (This is often called the Box-Mueller method.) So for the price of computing 2 udrn and 4 special functions we can generate 2 ndrns. (I will prove this is correct in class.)

4. We can improve this last method by sampling the angle (really the x and y components) directly using the rejection method. The pair (x, y) is sampled in the square $-1 < x, y < 1$ and values of $r^2 = x^2 + y^2 > 1$ are rejected (new pairs are sampled until a satisfactory pair is generated). Then the two gaussian numbers generated are $\sqrt{-2\ln(u_2)/(x^2 + y2)}(x, y)$. Hence at the price of rejecting a few pairs (acceptance rate is $\pi/4$) one does not need to compute any trig functions.

How can we generate a multivariate normal distribution, that pairs or triplets of guassian variates where the fluctuations are correlated. Here I give a simple algorithm for random vectors $\mathbf{x}$ with D components with a given (known) covariance matrix $\nu_{i,j}$.

1). (Before the sampling) do a Choleski decomposition of ν. This means to find a "square root" S of $\nu_{=}SS^T$. One can assume that S is a lower triangular matrix so that $S_{ij} = 0$ if $j > i$. then loop $i = 1, D$

$$S_{ii} = \sqrt{\nu_{ii} - \sum_{k=1}^{i} S_{ik}^2} \tag{6}$$

$$S_{ij} = \nu_{ij} - [\sum_{k} S_{ik}S_{jk}/S_{jj} \tag{7}$$

2. sample the vector $\mathbf{y}$ from an uncorrelated normal distribution with mean zero and unit variance.

3. Then $\mathbf{x} = S\mathbf{y}$ is a correlated normally distributed RN with zero mean.

References

Knuth, "Art of Computer Programming"
Kalos and Whitlock, "Monte Carlo Methods"
Abramowitz and Stegan, "Handbook of Mathematical Functions"
"Numerical Recipes
Rubenstein, "Simulation and the Monte Carlo Method"