i) the mesh values are equally spaced, i.e. $x_i = a + (i-1)h,\ i = 0, 1, \ldots, n$

ii)

$$\begin{aligned} A_i &= A_e \quad \text{if i is even} \\ A_i &= A_o \quad \text{if i is odd} \end{aligned}$$

i.e. there are only two different weights. Express your answer in terms of

$$\bar{g} \equiv \int_a^b g(x)dx \quad \bar{h} \equiv \int_a^b h(x)dx$$

$$g_e \equiv \sum_{i\ even} g(x_i) \quad g_o \equiv \sum_{i\ odd} g(x_i)$$

$$h_e \equiv \sum_{i\ even} h(x_i) \quad h_o \equiv \sum_{i\ odd} h(x_i)$$

11. * Use the method studied in 1.4 for evaluating singular integrals to express

$$\int_0^\infty dx \int_0^\infty dy \frac{f(x,y)}{y - y_0(x)}$$

as the sum of two numerically tractable integrals; one of them two dimensional and the other one dimensional. Assume that$f(x,y)$ falls off rapidly for large x and y such that you can take:

$$f(x,y) = 0 \qquad \begin{aligned} x &> R \text{ for all } y \\ y &> R \text{ for all } x \end{aligned}$$

and that $y_0(x)$ is bounded $0 < y_0(x) < R$.

References

1. *Methods of Numerical Integration*, Philip J. Davis and Philip Rabinowitz, Academic Press, 1975 ISBN 0-12-206350-3

2. *Gaussian Quadrature Formulas*, A. H. Stroud and Don Secrest, Prentice-Hall, 1966

Chapter 2

Introduction to Monte Carlo

How Monte Carlo techniques are used for the solution of physical problems is not well understood by many. A general belief is that they are designed to treat statistical (and other) problems based on probability. While this is certainly, true the techniques are much more powerful and their applications are much more general than this view would imply.

In its purest form, one could perhaps describe the method in terms of a relationship between the geometric and algebraic aspects of theoretical science. Over the years powerful methods have been developed to express the solution of probabilistic problems in analytical terms. With the advent of powerful computers the statistical problems can be modeled directly, opening the possibility of inferring the solution to the analogous analytical problem, thus reversing the originally intended direction. The solution to any problem which has an analytical expression that "looks like" the solution to a probability problem can be obtained by constructing a realization of the random variable problem. A large number of problems seem to fall into this category. This explanation is perhaps a bit too abstract for an introduction. It is hoped that this point will be clearer after the example in the following section.

2.1 Preliminary Notions - - Calculating π

Since it is perhaps not clear how one actually goes about calculating something with random numbers, a simple example is presented in this section to give the reader a feeling for the method. One widespread application of Monte Carlo methods involves calculating multidimensional integrals. We start out in this introduction with a simple two-dimensional integral.

Consider a square with corners at $(\pm 1, \pm 1)$ and a circle inscribed within this square. The square has sides of length 2 and area 4, and the circle a radius of 1 and area π. The area of the circle can be calculated analytically with the double integral:

$$\pi = \int_{-1}^{1} dx \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} dy. \tag{2.1}$$

We might imagine a problem in which it was necessary to evaluate the double integral on the right hand side of Eq. 2.1.

If we select a number of points at random, distributed uniformly in the square, we know that the probability that they will fall within the circle is equal to the ratio of the areas, in this case $\pi/4$. The computer can choose points at random in the ranges $-1 \leq x \leq 1$ and $-1 \leq y \leq 1$ and so arrive at points randomly distributed within the square*.

Under these conditions it is then a simple matter to test if a given point lies within the circle ($x^2 + y^2 \leq 1$). The fraction of points that fall within the circle will be $\pi/4$ Following is a program which calculates π in this manner.

```
  NMC=1000           ! number of Monte Carlo throws
  C=0.               ! zero the number of hits in the circle
  DO 1 I=1,NMC       ! throw NMC points
  X=2.*RAN3(D)-1.    ! RAN3 gives numbers between 0 and 1 so
  Y=2.*RAN3(D)-1.    ! multiply by 2 and subtract 1 to get -1,1
  IF(X**2+Y**2.GT.1.) GOTO 1  ! if outside of circle don't
  C=C+1.             ! count another hit in circle
1 CONTINUE           ! end of MC loop
  PRINT *,4.*C/NMC   ! print answer
  END
```

This short code illustrates three of the principal characteristics of Monte Carlo integrals.

1. The program is relatively easy to write once the algorithm is understood.

2. The technique is simple to extend to higher dimensions. In this case to extend the calculation to 3 dimensions (to compute the ratio of the volume of a sphere to that of a cube, $\frac{4\pi}{3\cdot 8} = \frac{\pi}{6}$) we only need to add the statement "Z=2.*RAN3(D)-1." and alter the "IF" statement to read "IF(X**2+Y**2+Z**2.GT.1.)GOTO 1".

*At the base of all Monte Carlo calculations lies a subroutine called the "random number generator" which calculates a random number uniformly distributed between 0 and 1. Here we use such a program, written in Fortran so it is transportable, called RAN3. It must be called with an argument so that Fortran knows that it is a subroutine, and not a variable, but this argument plays no other role. See the Appendix, Section A.4 for this subroutine and a description of its limitations.

3. The time increase is only linear (sometimes quadratic) with additional dimensions, whereas with classical multidimensional integration it is exponential, i.e. for each additional dimension the time required to do the calculation is *multiplied* by the number of points necessary to represent the function in each direction (usually a factor between 10 and 100).

It is this last point which makes the Monte Carlo method so powerful for treating multi-dimensional problems.

2.2 Evaluation of Integrals by Monte Carlo

Consider an integral of the form

$$\int_a^b g(x) f(x) dx \tag{2.2}$$

where $g(x)$ is positive definite. We can always arrange, by normalization, that

$$\int_a^b g(x) dx = 1 \tag{2.3}$$

so that $g(x)$ can be considered to represent a probability distribution function (*pdf*). Thus, independent of the original physical interpretation of the value of the integral (if it even had one!), it may be regarded as the solution of the (probabilistic) problem of computing the average of the function, $f(x)$, over a probability distribution function $g(x)$. We shall sometimes refer to this average, considered in a statistical sense, as $< f >$]. If we draw a large number, N, of random values of x from the distribution described by $g(x)$, we can directly estimate the average value of $f(x)$ by:

$$< f > \approx \frac{\sum_{i=1}^{N} f(x_i)}{N}; \qquad x_i \text{ drawn from } g(x). \tag{2.4}$$

This equation forms the basis of Monte Carlo evaluations of integrals. The set of N x_i are said to form a realization of the *pdf*, $g(x)$. This means that if we consider a range of size dx about the point x, the number of points found within this range should be $Ng(x)dx$ within statistical uncertainty. At this point the technique for selecting a set of x_i such that they provide a realization of the *pdf*, $g(x)$, is not obvious. This art of "sampling" a function is the principal subject of the remainder of this chapter. There are, however, some simple cases that can be treated using only the random number generator RAN3. Consider the integration of a function over the range $0 \leq x \leq 1$. In this case we can choose $g(x)$ to be a constant and, properly normalized, $g(x) \equiv 1$. Thus,

$$\int_0^1 f(x) dx \approx \frac{\sum_{i=1}^{N} f(x_i)}{N}; \qquad x_i \text{ uniformly distributed} \tag{2.5}$$

As a concrete example, consider

$$< x^2 >= \int_0^1 x^2 dx = \frac{1}{3}.$$

To calculate this integral using the Monte Carlo method one could write the following code.

```
      NMC=1000              ! number of Monte Carlo loops
      SUM=0.
      DO 1 I=1,NMC          ! the Monte Carlo loop
      X=RAN3(D)             ! choose a random number
    1 SUM=SUM+X**2          ! accumulate the sum
      PRINT *,SUM/NMC
      END
```

The Monte Carlo evaluation also provides a direct estimate of the error involved in the calculation of the integral. The average of the square of the function can be calculated at the same time as the average of the function,

$$< f^2 >= \frac{\sum_{i=1}^N f_i^2}{N} \qquad x_i \text{ from } g(x) \tag{2.6}$$

leading to an estimate of the variance of the integrand:

$$\sigma^2 =< f^2 > - < f >^2, \tag{2.7}$$

and the uncertainty of the full calculation will be given by:

$$\frac{\sigma}{\sqrt{N}}. \tag{2.8}$$

Note that σ is the variance of the function f, and will tend to a fixed number for the limit of large N, while the uncertainty in the calculation falls with square root of the number of trials.

2.3 Techniques for Direct Sampling

We will now examine methods for selecting a sequence of values of an independent variable, x, such that, in the limit of a large number of such selections, the distribution of values obtained follows the *pdf*, $g(x)$. The process is called "sampling the function $g(x)$". In the cases considered in this section the sampling process can (and should) be verified by sorting the values obtained into bins and comparing with the expected number, a procedure described in more detail below.

2.3.1 Cumulative Probability Distributions

The cumulative distribution function (*cdf*) is defined as

$$F(x) = \int_a^x g(x)dx \tag{2.9}$$

where the range of x is assumed to be from a to b. This function plays a central role in Monte Carlo work. Since $g(x) \geq 0$, $F(x)$ is a non-decreasing function of x. The normalization of $g(x)$ (defined in the range [a,b]) requires $F(b) = 1$. $F(x)$ represents the probability that a random variable drawn from $g(x)$ lies between a and x.

2.3.2 The Characteristic Function $\phi(t)$

The characteristic function,

$$\phi(t) \equiv \int e^{ixt} g(x)dx, \tag{2.10}$$

is very useful in the theory of probability. Here we will use it to derive a formula connecting probability distribution functions.

Consider the quantity, $y(x_1, x_2, \ldots, x_n)$, which is a function of a set of n independent variables $\{x_1, x_2, \ldots, x_n\}$. Each variable, x_i, will be considered as a random variable with its own *pdf*, $g_i(x_i)$. As the x_i take on their distribution of values, y can be calculated for each set $\{x_1, x_2, \ldots, x_n\}$ and, since it will show a distribution of values, may also be regarded as a random variable. It, like any other random variable, will have a distribution of probability, say $g_Y(y)$. Clearly $g_Y(y)$ is completely determined by the set of functions $\{g_i(x_i)\}$ and the functional relationship, $y(x_1, x_2, \ldots, x_n)$. We wish to find the function, $g_Y(y)$, given these conditions.

We may be sure that the function $g_Y(y)$ represents the distribution of the random variable, y, if it reproduces all of the *moments* of the function $y(x_1, x_2, \ldots, x_n)$ as the random variables $\{x_i\}$ take on their distributions of values. Thus we have the set of conditions,

$$\int dy y^m g_Y(y) = \int dx_1 \int dx_2 \ldots \int dx_n y^m(x_1, x_2, \ldots, x_n) g_1(x_1) g_2(x_2) \ldots g_n(x_n) \tag{2.11}$$

$$m = 0, 1, \ldots, \infty$$

Multiplying both sides of the above equation by $\frac{(it)^m}{m!}$ and summing over all m, gives two expressions for the characteristic function for the variable y

$$\phi_Y(t) = \int dy e^{ity} g_Y(y) = \int dx_1 \int dx_2 \ldots \int dx_n e^{iy(x_1, x_2, \ldots, x_n)t} g_1(x_1) g_2(x_2) \ldots g_n(x_n) \tag{2.12}$$

or, inverting the Fourier transform, we find an explicit expression for the *pdf* of y:

$$g_Y(y') = \int dx_1 \int dx_2 \ldots \int dx_n \delta[y' - y(x_1, x_2, \ldots, x_n)] g_1(x_1) g_2(x_2) \ldots g_n(x_n). \tag{2.13}$$

For the case of only a single variable, x, Eq. 2.12 becomes:

$$\phi_Y(t) = \int dy e^{ity} g_Y(y) = \int dx e^{iy(x)t} g(x). \tag{2.14}$$

If $y(x)$ is a monotonic function of x we can invert the equation for $y(x)$ to find $x(y)$ and, by a change of variables, we can write the right hand side of the above equation as

$$\phi(t) = \int dy e^{iyt} g\left[x(y)\right] / \frac{dy}{dx}(y). \tag{2.15}$$

2.3.3 The Fundamental Theorem of Sampling

Consider $F(x)$, the cumulative distribution function itself, as a random variable resulting from the insertion for its argument of a random variable, x, obtained from the sampling of $g(x)$, **the same $g(x)$ from which $F(x)$ was calculated**. Then, for Eq. 2.14 we have, with $F(x) = y(x)$,

$$\int_0^1 e^{iFt} g_F(F) dF = \int_a^b e^{iF(x)t} g(x) dx. \tag{2.16}$$

Hence we have an equation involving the distribution function for the random variable F, $g_F(F)$. From the definition of $F(x)$, $dF = g(x)dx$, so we can rewrite Eq. 2.16, making the transformation of the integration variable to F on the r.h.s., as

$$\int_0^1 e^{iFt} g_F(F) dF = \int_0^1 e^{iFt} dF$$

and we can identify $g_F(F) \equiv 1$. Thus, under these conditions, F is uniformly distributed from 0 to 1. Thus if x is distributed according to $g(x)$, $F(x)$ is distributed uniformly. Of course the inverse holds as well.

Since the function $F(x)$ is non-decreasing[†] it can be inverted to obtain $x(F)$. Thus if we choose a sequence of random variables, F_i, uniformly distributed from 0 to 1, the x_i which result from the expression $x(F)$ will be distributed according to $g(x)$. This method of sampling can be applied to most one-dimensional sampling problems.

2.3.4 Sampling Monomials $0 \leq x \leq 1$

We can use the results of the previous section to sample a power of x over the range 0 to 1. Consider a *pdf*, $g(x)$, with the form:

$$g(x) = 2x \text{ for } 0 \leq x \leq 1.$$

[†] We will actually assume that it is increasing, thus not allowing $g(x)$ to go to zero at any point other than one on the boundary of the region. This will be true for the examples we are about to treat.

It is correctly normalized since $\int_0^1 g(x)dx = 1$. Since

$$F(x) = \int_0^x g(x)dx = x^2$$

we have

$$x(F) = \sqrt{F}.$$

Therefore, choosing a number of values of F=RAN3(D), and taking the square root of each one, will give values of x with a linear probability distribution. Treating the more general problem of a power of x:

$$h_n(x) = (n+1)x^n \qquad 0 \leq x \leq 1 \tag{2.17}$$

and

$$F(x) = (n+1)\int_0^x x^n dx = x^{n+1} \tag{2.18}$$

so that

$$x(F) = F^{\frac{1}{n+1}}. \tag{2.19}$$

It is very important to verify that the desired distribution has indeed been obtained for each sampled function. This can be done by a procedure often known as binning. It consists of counting the number of values which fall into a set of specified regions of x and comparing with the distribution expected. For the interval $0 \leq x \leq 1$, 20 bins are often sufficient, leading to a bin size of 0.05. That is, we count the number of times the result falls between 0 and 0.05, 0.05 and 0.10, ..., 0.95 and 1.0. These numbers are then compared with the theoretical expectations. Following is a program illustrating this procedure.

```
   DIMENSION IBIN(20)
   NMC=1000
   DO 1 I=1,NMC              ! start Monte Carlo loop
   F=RAN3(D)                 ! choose an F
   X=SQRT(F)                 ! put any function here
   J=20.*X+1.                ! calculate bin index
   IBIN(J)=IBIN(J)+1         ! one more for bin J
 1 CONTINUE                  ! finished collecting data
   DO 2 J=1,20               ! now print out results
   XMID=0.05*(J-1)+0.025     ! compute mid-point of the bin
   G=2.*XMID                 ! calculate the expected pdf at that
   NTHEORY=NMC*G*0.05        ! point and the expected number of
 2 PRINT 3,XMID,IBIN(J),NTHEORY  ! points based on the pdf g
 3 FORMAT(F6.2,2I5)
   END
```

Note that the bin is indexed by the value "above", i.e. bin one has a values which fall between 0 and 0.05; bin two, those between 0.05 and 0.10; etc. In some cases it is not sufficient to compare with the *pdf*, $g(x)$, calculated at the mid-point of the bin. A more accurate value can be obtained, since the integral over the bin is also known. Since the probability that x lies between x and $x+\delta$ is $F(x+\delta)-F(x)$ (δ need not be small) we can replace the "DO 2" loop with:

```
   DO 2 J=1,20              ! now print it out
   XLOW=0.05*(J-1)          ! compute low point of the bin
   XHIGH=0.05*J             ! compute high point of the bin
   G=XHIGH**2-XLOW**2       ! calculate the integrated pdf at that
   NTHEORY=NMC*G            ! point and the expected number of
 2 PRINT 3,XMID,IBIN(J),NTHEORY  ! points based on the pdf g
```

We are now in a position to illustrate some of the characteristics of Monte Carlo integration. Consider the integral:

$$\int_0^1 x^4 dx = \frac{1}{5}. \tag{2.20}$$

Taking advantage of the fact that we can write the integrand as the product of two factors, $g(x)f(x)$, there are a number of options for performing this integration with Monte Carlo. We could consider $g(x)$ to be unity and $f(x)$ to be x^4 as we did before, but there are, in addition, the following possibilities:

$$\begin{array}{cccl} g(x) & f(x) & \text{Integral} & \text{Monte Carlo result} \\ 2x & x^3 & \frac{1}{2}\int_0^1(2x)(x^3)dx & \frac{1}{2}\frac{\sum_{i=1}^N x_i^3}{N} \text{ with } x_i \text{ from } g(x)=2x \\ 3x^2 & x^2 & \frac{1}{3}\int_0^1(3x^2)(x^2)dx & \frac{1}{3}\frac{\sum_{i=1}^N x_i^2}{N} \text{ with } x_i \text{ from } g(x)=3x^2 \\ 4x^3 & x & \frac{1}{4}\int_0^1(4x^3)(x)dx & \frac{1}{4}\frac{\sum_{i=1}^N x_i}{N} \text{ with } x_i \text{ from } g(x)=4x^3 \\ 5x^4 & 1 & \frac{1}{5}\int_0^1(5x^4)(1)dx & \frac{1}{5}\frac{\sum_{i=1}^N 1}{N} \text{ with } x_i \text{ from } g(x)=5x^4 \end{array} \tag{2.21}$$

The last case gives the exact answer since the Monte Carlo evaluation gives 1/5 without ever throwing a random number. This limit to the absurd is taken to illustrate the point that the ability to sample a function exactly is equivalent to being able to integrate it.

Comparing the variance of the integrand to the value of the integral we find:

$$\frac{\sigma}{<f>} = \frac{m}{\sqrt{2m+1}} \tag{2.22}$$

where m is the power in the sums in Eqs. 2.21. **The more of the variation of the integrand that can be included in the sampling function, rather than the evaluated function, the more accurate the integral will be, since the evaluated function will approach a constant.** It is thus of tantamount importance to sample the function, $g(x)$, such that a majority of the evaluations take place where it is large. A carefully chosen sampling function can save orders of magnitude in the computing time for a given level of accuracy. As usual, the error in counting data decreases only as $\frac{1}{\sqrt{N}}$ and the calculator has little control over this fact of nature, but he does have control over the coefficient which multiplies this factor by the judicious choice of a sampling function.

2.3.5 Sampling Functions $0 \le x \le \infty$

The Exponential Function

An exponential is easy to sample and very common. The *pdf* is:

$$g_e(x) = pe^{-px}, \tag{2.23}$$

and the *cdf*:

$$F(x) = 1 - e^{-px}. \tag{2.24}$$

Since we can invert this equation we can find the expression for sampling the exponential as:

$$x = -\frac{1}{p}\ln(1-F)$$

We can simplify this expression a little by observing that, if F is uniformly distributed, so is $1-F$. Thus we can use for the sampling equation:

$$x = -\frac{1}{p}\ln F. \tag{2.25}$$

Other Algebraically Invertible Functions

In order to be able to directly sample a function it must have certain properties:

1. It must be normalizable and positive definite so it can represent a *pdf*.

2. It must be integrable analytically.

3. The integral (anti-derivative) must be invertible

As a first example consider the function $\frac{1}{(a+bx)^n}$. Its integral is:

$$\int \frac{dx}{(a+bx)^n} = \frac{-1}{(n-1)b(a+bx)^{n-1}} \qquad n>1 \tag{2.26}$$

The normalization integral is:

$$\int_0^\infty \frac{dx}{(a+bx)^n} = \frac{1}{(n-1)ba^{n-1}}$$

so that the *pdf* is given by:

$$g(x) = \frac{(n-1)ba^{n-1}}{(a+bx)^n} \qquad \equiv g_1(x) \tag{2.27}$$

and

$$F(x) = 1 - \frac{1}{(1+\frac{b}{a}x)^{n-1}}. \tag{2.28}$$

Solving for x, the sampling equation is:

$$x = \frac{a}{b}(F^{-\frac{1}{n-1}} - 1). \tag{2.29}$$

Another example of a different type is based on the integral:

$$\int \frac{dx}{a^2+b^2x^2} = \frac{1}{ab}\tan^{-1}\frac{bx}{a}.$$

so

$$g(x) = \frac{2ab}{\pi(a^2+b^2x^2)} \qquad \equiv g_2(x). \tag{2.30}$$

The *cdf* is,

$$F(x) = \frac{2}{\pi}\tan^{-1}\frac{bx}{a} \tag{2.31}$$

and the sampling equation,

$$x = \frac{a}{b}\tan\frac{\pi F}{2}. \tag{2.32}$$

Other possibilities are:

$$\int \frac{xdx}{(a^2+x^2)^{n+1}} = -\frac{1}{2n(a^2+x^2)^n} \qquad \rightarrow g_3(x) \tag{2.33}$$

and

$$\int \frac{xdx}{(a^2+b^2x)x^{\frac{1}{2}}} = \frac{2}{ab}\tan^{-1}\frac{bx^{\frac{1}{2}}}{a} \qquad \rightarrow g_4(x) \tag{2.34}$$

One function which can be very useful, since it allows an arbitrary power dependence at the origin while keeping the same fall-off at infinity, is:

$$g(x) = \frac{(n+1)a^2x^n}{(x^2+a^2)^{\frac{n+3}{2}}} \qquad \equiv g_5(x) \tag{2.35}$$

with

$$F(x) = \left(\frac{x}{\sqrt{a^2+x^2}}\right)^{n+1}, \tag{2.36}$$

and the sampling equation,

$$x = \frac{a}{\sqrt{F^{-\frac{2}{n+1}} - 1}}. \tag{2.37}$$

A function that allows the sampling of a nuclear density, among other physical interpretations, is:

$$g(x) = \frac{ap}{\ln\frac{a+b}{b}}\frac{1}{a+be^{px}} \equiv g_6(x) \tag{2.38}$$

with

$$F(x) = \frac{\ln\frac{a+b}{b+ae^{-px}}}{\ln\frac{a+b}{b}} \tag{2.39}$$

sampled by,

$$x = \frac{1}{p}\ln\left\{\frac{a}{b\left[\left(1+\frac{a}{b}\right)^F - 1\right]}\right\}. \tag{2.40}$$

A summary of the properties of these functions is found in Table 2.1

Sampling a Gaussian Distribution

The Gaussian function is not easily sampled in this direct manner. While the *pdf* is given by:

$$g_g(x) = \frac{2}{\sqrt{\pi}}e^{-x^2} \quad 0 \le x \le \infty \tag{2.41}$$

and the *cdf* by

$$F(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-x^2}dx \equiv \mathrm{erf}(x), \tag{2.42}$$

the error function is not simply invertible. However one can use the trick often employed with Gaussians, of transforming from rectangular to polar coordinates for two functions. We first notice that for two functions $g_g(x)$:

$$g_g(x)g_g(y)dxdy = \frac{4}{\pi}e^{-r^2}rdrd\theta$$

	$g(x)$	$F(x)$	x	$g(0)$	$g(\infty)$	$F(0)$	$1-F(\infty)$
1)	$\frac{(n-1)ba^{n-1}}{(a+bx)^n}$	$1-\frac{1}{(1+\frac{b}{a}x)^{n-1)}}$	$\frac{a}{b}(F^{-\frac{1}{n-1}}-1)$	$const$	$\frac{1}{x^n}$	x	$\frac{1}{x^{n-1}}$
2)	$\frac{2ab}{\pi(a^2+b^2x^2)}$	$\frac{2}{\pi}\tan^{-1}\frac{bx}{a}$	$\frac{a}{b}\tan\frac{\pi F}{2}$	$const$	$\frac{1}{x^2}$	x	$\frac{1}{x}$
3)	$\frac{2na^{2n}x}{(a^2+x^2)^{n+1}}$	$1-\frac{1}{(1+\frac{x^2}{a^2})^n}$	$a\sqrt{F^{-\frac{1}{n}}-1}$	x	$\frac{1}{x^{2n+1}}$	x^2	$\frac{1}{x^{2n}}$
4)	$\frac{ab}{\pi(a^2+b^2x)x^{\frac{1}{2}}}$	$\frac{2}{\pi}\tan^{-1}\frac{bx^{\frac{1}{2}}}{a}$	$\frac{a^2}{b^2}\tan^2\frac{\pi F}{2}$	$\frac{1}{x^{\frac{1}{2}}}$	$\frac{1}{x^{\frac{3}{2}}}$	$x^{\frac{1}{2}}$	$\frac{1}{x^{\frac{1}{2}}}$
5)	$\frac{(n+1)a^2x^n}{(x^2+a^2)^{\frac{n+3}{2}}}$	$\left(\frac{x}{\sqrt{a^2+x^2}}\right)^{n+1}$	$\frac{a}{\sqrt{F^{-\frac{2}{n+1}}-1}}$	x^n	$\frac{1}{x^3}$	x^{n+1}	$\frac{1}{x^2}$
6)	$\frac{ap}{\ln(\frac{a+b}{b})}\frac{1}{(a+be^{px})}$	$\frac{\ln\frac{a+b}{b+ae^{-px}}}{\ln\frac{a+b}{b}}$	$\frac{1}{p}\ln\{\frac{a}{b[(1+\frac{a}{b})^F-1]}\}$	$const$	e^{-px}	x	e^{-px}

Table 2.1: Examples of *pdf*'s Which can be Sampled Directly

$$= \frac{2}{\pi}e^{-r^2}(2rdr)d\theta$$

Thus, to sample the same area (same probability) we must take a slice of

$$e^{-r^2}2rdr = e^{-z}dz; \quad z = r^2$$

and

$$\frac{2}{\pi}d\theta.$$

Transforming back to Cartesian coordinates:

$$x = r\cos\theta, \quad y = r\sin\theta$$

the sampling functions are:

$$x = \sqrt{-\ln F_1}\cos 2\pi F_2 \tag{2.43}$$

and

$$y = \sqrt{-\ln F_1}\sin 2\pi F_2 \tag{2.44}$$

where we have extended the range of θ to go $0 \to 2\pi$ so that 2.43 and 2.44 actually give x and y in the range $-\infty \to \infty$. If values from $0 \to \infty$ are desired the 2π should be replaced by $\frac{\pi}{2}$. This method is known by the name of "Box-Müller" and gives two independent Gaussian variables by calling on two random variables distributed uniformly.

2.3.6 Brute-force Inversion of F(x)

One can always numerically integrate $g(x)$ and invert $F(x)$ but this is often not very satisfactory especially if the *pdf* is changing rapidly. Since what is needed is a table of x in even steps of F, a possible procedure is to set a series of targets for F at equal intervals. Such an algorithm might work as follows for a modest array of 100 values for the *cdf*:

1. Set first index for the array `X`, `M=1`
2. Set the first target, $F_T = 0.01$, previous target=0., $x_0 = 0$.
3. Start the integral of $g(x)$ in steps of ϵ from the previous target and look for the new target value, i.e. compute $F(x = x_0 + \epsilon), F(x = x_0 + 2\epsilon), \ldots$ until $F \geq F_T$. If this process took less than some minimum number of steps (since the location of the target position in x may be inaccurate) restart with a smaller ϵ. If it took more than necessary, increase the size of ϵ for the next target.
4. Record the value $x_0 + n\epsilon$ in the array for x, `X(M)`.
5. Increment F_T by 0.01 and M by 1. Quit if M=100 (i.e. if $F_T = 1$). Otherwise go back to step 3.

To sample $g(x)$, take the value `X(100*RAN3(D)+1)`. This example would only provide 100 different random values although interpolation is possible. In a realistic case the table would be much larger. While this method is clumsy, it is capable of sampling functions of a few dimensions in the same way as described above with each variable being stepped through its range. Variables should be alternately increased and decreased to avoid the discontinuity which would accompany a large change in the variable (See Figure 2.1).

At each step in the 2-dimensional space, check against the target F and if it is reached, record x and y in two tables indexed by m. To sample the function choose U randomly and take

$$x = x(NU+1)$$

and

$$y = y(NU+1).$$

$$\begin{array}{cccccccc}
 & x & \rightarrow & & & & & \\
y & 0 & 0\rightarrow & \epsilon\rightarrow & 2\epsilon\rightarrow & \ldots\rightarrow & (n-1)\epsilon\rightarrow & n\epsilon \\
\downarrow & & & & & & & \downarrow \\
 & \epsilon & 0\leftarrow & \epsilon\leftarrow & 2\epsilon\leftarrow & \ldots\leftarrow & (n-1)\epsilon\leftarrow & n\epsilon \\
 & \downarrow & & & & & & \\
 & 2\epsilon & 0\rightarrow & \epsilon\rightarrow & 2\epsilon\rightarrow & \ldots\rightarrow & (n-1)\epsilon\rightarrow & n\epsilon \\
 & & & & & & & \downarrow \\
 & 3\epsilon & 0\leftarrow & \epsilon\leftarrow & 2\epsilon\leftarrow & \ldots\leftarrow & (n-1)\epsilon\leftarrow & n\epsilon \\
 & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{array}$$

Figure 2.1: Strategy for Incrementing Variables. By following the direction of the arrows the *cdf* will be accumulated in a continuous manner.

This method has some interest in that one can store anything in the arrays which correspond to a given value of U. For example, one could choose SU$_3$ matrices uniformly distributed in the volume element of their defining parameters by this means.

2.3.7 The Rejection Technique

For sampling a variable over a finite range, the rejection technique is often very useful. To set up this method it is first necessary to find the maximum of the *pdf* $g(x)$, g_{max} and renormalize to form the quantity

$$\tilde{g}(x) = \frac{g(x)}{g_{max}}$$

Procedure:

1. Choose a random value of x uniformly in the range $a \leq x \leq b$
2. Choose a second uniformly distributed random number y,
3. If $\tilde{g}(x) \geq y$ return (i.e. use) the value of x picked. Otherwise return to step 1.

In Figure 2.2 consider that the space between the vertical bars is dx. Then the area above the curve will be rejected and the area below accepted. Since the partitioning of the two areas is according to the height of the curve the resulting probability follows that of the curve. Note that if we had not normalized the maximum of the modified $g(x)$ to unity there might be two possible outcomes. If the maximum value were less than one the distribution would still be correct, we would simply waste time for those values of the random number chosen between the maximum value and unity which would always be rejected. If the maximum value were greater than one, however, the distribution would be incorrect since all values of x for $g(x)$ greater than one would have equal probability.

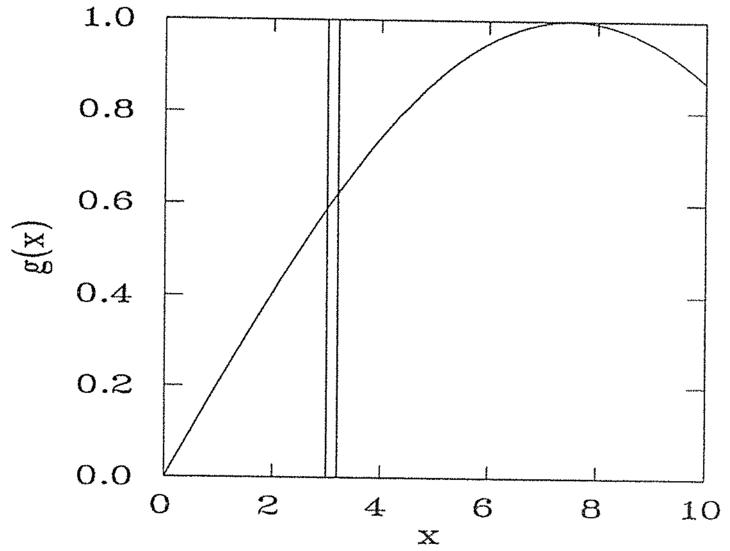

Figure 2.2: Logic of the Rejection Method. If the point chosen falls within the two lines below the curve it is part of that area and is accepted. If it falls above it is not part of the area and is rejected.

2.3.8 Sums of Random Variables

Suppose that we choose x_1 from g_1
x_2 from g_2

What is the distribution of the random variable $y \equiv x_1 + x_2$? Using the characteristic function:

$$\phi_Y(t) = \int_0^\infty e^{iyt} g_Y(y) = \int_0^\infty \int_0^\infty e^{i(x_1+x_2)t} g_1(x_1) g_2(x_2) dx_1 dx_2$$

$$\phi_Y(t) = \int_0^\infty e^{ix_1t} g(x_1)dx_1 \int_0^\infty e^{ix_2t} g_2(x_2)dx_2 = \phi_1(t)\phi_2(t)$$

or that the sum of random variables leads to the product of characteristic functions.

Example: If

$$g_1(x) = ae^{-ax} = g_2(x)$$

then

$$\phi_1(t) = \phi_2(t) = \frac{a}{a - it}$$

so

$$\phi_Y(t) = \frac{a^2}{(a - it)^2}$$

and

$$g_Y(y) = a^2 y e^{-ay}.$$

Thus, by sampling the exponential twice and adding the two values, one can sample this (normalized) *pdf*. In general the sum of n samples of an exponential *pdf* gives a sample of the function

$$\frac{a^n}{(n-1)!} y^{n-1} e^{-ay}. \tag{2.45}$$

The exponentials need not have the same range.

Example: If

$$g_1(x_1) = ae^{-ax_1} \text{ and } g_2(x_2) = be^{-bx_2} \tag{2.46}$$

the sum $y = x_1 + x_2$ gives

$$g_Y(y) = \frac{ab}{(b-a)}(e^{-ay} - e^{-by}) \tag{2.47}$$

Note that we are sampling the difference of two exponentials, not to be confused with an entirely different technique for sampling the sum of two distributions which is treated in the next section.

Example: If

$$g_1(x_1) = ae^{-ax_1}; \qquad g_2(x_2) = be^{-bx_2}; \qquad g_3(x_3) = ce^{-cx_3}$$

the sum $y = x_1 + x_2 + x_3$ gives

$$g_Y(y) = -\frac{abc}{(a-b)(b-c)(c-a)}\left[(b-c)e^{-ay} + (c-a)e^{-by} + (a-b)e^{-cy}\right].$$

Consider choosing $a \to 2a$, $b \to 2b$ and $c = a + b$. Thus if we take the sum of x_1 from e^{-2ax_1}
x_2 from e^{-2bx_2}
x_3 from $e^{-(a+b)x_3}$

the result is a sample from the normalized distribution:

$$g_Y(y) = \frac{2ab(a+b)}{(b-a)^2}(e^{-ay} - e^{-by})^2 \tag{2.48}$$

which behaves as an exponential for large y, and as y^2 near the origin. This turns out to be a useful result which we will use in Chapter 11.

2.3.9 Selection on the Random Variables

The rejection technique discussed before was on the probability distribution function, $g(x)$, i.e. if $\tilde{g}(x) \geq$RAN3 we kept x. Here we are discussing the comparison of two (or more) random *variables* and accepting or rejecting the values depending on some condition.

As an example suppose we choose

$$\begin{array}{ll} x_1 & \text{from } g_1(x) \\ x_2 & \text{from } g_2(x) \end{array}$$

and keep x_1 iff $x_1 \geq x_2$.

What is the distribution of x_1?

The problem just stated is one in conditional probability and the solution can be expressed as the product of two factors. First, x_1 must fall in some range dx_1 about x_1 so that the first factor in the probability is $g_1(x_1)dx_1$.
Secondly we must have $x_2 \leq x_1$ so that the total probability is given by the first factor multiplied by the probability that x_2 is less than x_1,

$$F_2(x_1) = \int_a^{x_1} g_2(x)dx.$$

Thus the combined probability for the event described is

$$g_1(x)F_2(x)dx$$

The procedure just outlined provides a method for sampling the product of two functions. If a function can be written as the product of a *pdf* and a *cdf* it can be sampled. Since all events (results of the two separate sampling procedures) are not retained, the probability is not normalized. If we use all values generated by this technique we must make a correction for the acceptance rate. This simply corresponds to the fact that

$$\mathcal{N} \equiv \int_a^b g_1(x)F_2(x)dx < 1.$$

From the definition of $\mathcal{N}$ we see that it is equal to the total probability (over all values of x in the desired range) that an x drawn from $g_1(x)$ will be greater than an x drawn from $g_2(x)$ so it can be calculated in the selection process. Of course we must correct the integral being evaluated for this fact. To be more explicit, if we want to integrate

$$\int_a^b g_1(x)F_2(x)f(x)dx$$

then we must write

$$\int_a^b g_1(x)F_2(x)f(x)dx = \mathcal{N}\int_a^b \frac{g_1(x)F_2(x)}{\mathcal{N}}f(x)dx = \mathcal{N}\int_a^b G(x)f(x)dx$$

where $G(x)$ represents the normalized *pdf* which results from the rejection procedure being discussed.

An outline of the procedure to sample $g_1(x)F_2(x)$ is as follows:

1. N_T=0
2. choose x_1 from g_1 and x_2 from g_2
3. $N_T = N_T + 1$
4. if $x_1 \geq x_2$ return $x = x_1$, otherwise go to step 2

The process goes from 2 through 4 each time until a value is returned. The Monte Carlo sum will need to be corrected by the factor N/N_T where N is the number of Monte Carlo loops i.e. the number of times this selection procedure is called and a result returned.

Example:

To sample a linear *pdf* we can take

$$g_1(x) = g_2(x) = 1.$$

Here $F_2(x) = x$ and the product function is the unnormalized *pdf*; $\bar{g}(x) = x$. Since the procedure outlined always gives a value of x it *is* normalized and corresponds to $G(x) = 2x$. So to integrate just x we must correct by a factor of $\frac{1}{2}$. In other words, we are integrating x, but the Monte Carlo integrates $2x$ because it always does normalized distributions.

In this particular simple case we don't have to reject at all since we can keep the larger of the two values. This provides a more efficient way to sample $g(x) = 2x$: generate two uniform random variables and use the larger of the two. In the same way, to sample $3x^2$; generate 3 uniform random variables and keep the largest, etc.

If we enforce the reverse condition, i.e. choose x_1 from g_1, x_2 from g_2 and keep x_1 if $x_1 \leq x_2$ then clearly we are sampling

$$[1 - F_2(x)]g_1(x)dx.$$

Generating 3 uniform random numbers and keeping the *middle* one samples

$$6x(1-x)$$

since it corresponds to $[1 - F(x)]g(x)F(x)$ where $g(x)$ is uniform between 0 and 1.

Clearly one can consider more complicated functions. One example is the integral:

$$\int_0^\infty e^{-x}\mathrm{erf}(x)dx.$$

$$\text{Let: } \begin{aligned} g_1(x) &= e^{-x} \\ F_2(x) &= \mathrm{erf}(x) \end{aligned}$$

which means that we sample $g_1(x_1)$ ($\equiv$ exponential) and $g_2(x_2)$ ($\equiv$ Gaussian) and accept if $x_1 \geq x_2$. Since the Monte Carlo sum is evaluating unity over this sample, the result *is* unity *except* for the acceptance fraction i.e. the integral *is* the acceptance fraction. The exact value of this integral is known to be

$$e^{\frac{1}{4}}\mathrm{erfc}(\frac{1}{2}) \approx 0.615690.$$

Clearly a large number of possibilities are available using combinations of the other methods, in particular the functions in Table 2.1. Counting the number of rejections in an algorithm is the basis of one technique for calculating the energy of many-body systems (see Chapter 11).

As one practical application we can generate a function with any power for small x, but an exponential behavior for large x by combining the exponential function with the *cdf* from $g_5(x)$ i.e.

$$\left(\frac{x}{\sqrt{a^2+x^2}}\right)^{n+1} be^{-bx}. \tag{2.49}$$

Choose x from the exponential distribution, y from $g_5(y)$ and return x if it is greater than y, otherwise try again. Remember that the normalization must be kept as a separate factor.

2.3.10 The Sum of Probability Distribution Functions

Suppose that we can write a probability distribution function as a sum of other *pdf* 's:

$$G(x) = \sum_{n=1}^{N} a_n g_n(x) \qquad a_n > 0 \tag{2.50}$$

with:

$$\int_a^b g_n(x)dx = 1 \quad \text{and} \quad \sum_{n=1}^{N} a_n = 1. \tag{2.51}$$

The second of these last two conditions follows from the first. We can sample the function, $G(x)$ with the following simple algorithm:

1. Choose n with probability a_n

2. Sample $g_n(x)$ to give x

To carry out the first step, make an array such that:

```
      H(1)=A(1)
      H(2)=H(1)+A(2)
      H(3)=H(2)+A(3)
        .   .
        .   .
        .   .
      H(N)=H(N-1)+A(N)          ! =1
```

Then, after choosing a random number, U, a series of tests will determine which value of n is appropriate. That is, if $U \leq H(1)$ then n=1, else if $U \leq H(2)$ then n=2, else etc. As a simple example, consider the sampling procedure for

$$\frac{a}{2}e^{-ax} + \frac{b}{2}e^{-bx}. \tag{2.52}$$

First throw a random number F
    if $F > \frac{1}{2}$ sample be^{-bx};   else sample ae^{-ax}

This method can be used to sample a variety of different functions, even infinite series in certain special cases. Consider the function:

$$f(x) = \frac{ae^{-bx}}{1 - ae^{-bx}} \qquad 0 < a < 1; \;\; 0 \leq x < \infty$$

$$= \frac{1}{1 - ae^{-bx}} - 1 = \sum_{n=1}^{\infty} a^n e^{-nbx}.$$

Since

$$\int_0^\infty f(x)dx = \sum_{n=1}^{\infty} \frac{a^n}{nb} = -\frac{1}{b}\ln(1-a)$$

and the individual normalized *pdf* 's are:

$$g_n(x) = nbe^{-nbx},$$

then

$$f(x) = \frac{1}{b}\sum_{n=1}^{\infty}\frac{a^n}{n}g_n(x),$$

and the normalized *pdf* is:

$$G(x) = -\frac{\sum_{n=1}^{\infty}\frac{a^n}{n}g_n(x)}{\ln(1-a)}.$$

For large n, a sampling of $g_n(x)$ will return small values of x. If the precise location of points near the origin is not crucial the series may be truncated at some large value of $n(= n_0)$.

$$G(x) \approx \sum_{n=1}^{n_0} \alpha_n g_n(x)$$

$$\alpha_n = -\frac{\frac{a^n}{n}}{\ln(1-a)}$$

with

$$\beta_n = \sum_{i=1}^{n} \alpha_i \qquad \beta_0 \equiv 0$$

Throw U =RAN3 if $U > \beta_{n_0}$ take $x = 0$
    if $\beta_n \geq U > \beta_{n-1}$ take x from $g_n(x)$.

Special Cases

The technique of summing two (or more) positive *pdf* 's can be applied in some cases to sample polynomials where the coefficients are not all positive. Consider the case of a *pdf* represented by:

$$g(x) = a + bx + cx^2 \qquad -1 \leq x \leq 1. \tag{2.53}$$

where we assume that it is normalized i.e., $a + \frac{c}{3} = \frac{1}{2}$. Since $g(x)$ must be positive definite, it must be that $a > 0$. We will assume that $c > 0$ as well, as is the case for the sum of squares of linear functions, often arising for cross-section data. In order that $g(x)$ remain positive in the region of interest, $-1 \leq x \leq 1$, it must be true that $b^2 < 4ac$[‡] since the minimum value of $g(x)$ will occur at x_0 given by $g'(x_0) = 0$ or

[‡] We assume that the minimum in the function occurs (at x_0) within the region $-1 \leq x \leq 1$. If it does not the method can still be made to work provided that the polynomial does not have a zero, i.e. that $a > \frac{b^2}{4c}$.

$x_0 = -\frac{b}{2c}$ and is

$$g(x_0) = a - \frac{b^2}{4c}.$$

Writing:

$$g(x) = c(x - x_0)^2 + 2\left(a - \frac{b^2}{4c}\right)\frac{1}{2},$$

we have expressed the polynomial as the sum of two positive definite functions with the second in the form of a probability times a *pdf* [$\frac{1}{2}$ is the probability distribution function for a uniform distribution $-1 \leq x \leq 1$]. Thus, after choosing a first random number U_1,

if $U_1 < 2(a - \frac{b^2}{4c})$ select from a uniform distribution $-1 \leq x \leq 1$,

else sample a quadratic random variable, z^2, $z \equiv x - x_0$.

Using

$$\mathcal{N} \equiv \int_{-1}^{1}(x - x_0)^2 dx = \int_{-1-x_0}^{1-x_0} z^2 dz = \frac{2}{3}(1 + 3x_0^2) \tag{2.54}$$

with

$$F(z) = \frac{1}{\mathcal{N}}\int_{-1-x_0}^{z} z^2 dz = \frac{1}{3\mathcal{N}}\left[z^3 + (1 + x_0)^3\right] \tag{2.55}$$

we find

$$x = x_0 + \left[2(1 + 3x_0^2)U_2 - (1 + x_0)^3\right]^{\frac{1}{3}} \tag{2.56}$$

where U_2 is a second uniformly distributed random number. Note that some care is necessary in extracting the cube root since the computer will attempt to take the logarithm of a quantity which may be negative. To avoid this problem, write

$$A^{\frac{1}{3}} = \mathrm{sign}(A)|A|^{\frac{1}{3}}. \tag{2.57}$$

2.4 The Metropolis Algorithm

While the techniques just studied for sampling have a certain elegance, they have one serious flaw: it is only with the greatest of difficulty that they can sample multi-dimensional functions. On the other hand, the Metropolis algorithm is designed with exactly that in mind. Without it, Monte Carlo methods would be much less powerful than they are. It turns out that it also leads to a new way of representing a function which, in turn, leads to a new way of solving physical problems, one very well suited to many-body systems. Those methods will be studied in Chapters 6 and 11.

2.4.1 The Method Itself

The algorithm is based on the concept of a random walk. In the two-dimensional classical random walk, a point is started at the origin of an $x - y$ coordinate system and during successive time intervals is allowed to move one unit in any (rectangular) direction with equal probability. Thus it might move one step to the right, another step to the right, one step up, one down, etc. The probabilistic description of this process is that of an expanding Gaussian probability distribution which, in the limit of some very long time, would cover the entire 2-dimensional space.

If a similar random walker is started off at some point in a multi-dimensional space (a walker is specified by giving values of all of the coordinates in the space) and allowed to take a vectorial step (i.e. all components are changed at once) he will eventually find his way into all parts of the space. However, for the present application, we do not want the walker to simply cover all parts of the space equally, we would like him to spend more time in those regions where the function to be sampled is large. In this way the ensemble of the positions of the walker provide a *realization* of the *pdf* since the probability of finding a walker in a given region should be the same as $g(x_1, x_2, \ldots, x_N)$. In other words, if one could bin up all of the positions of all of the steps of the walker the probabilities would follow the *pdf*.

$$\begin{aligned}
x_1' &= x_1 + \delta\{-1, 1\} \\
y_1' &= y_1 + \delta\{-1, 1\} \\
z_1' &= z_1 + \delta\{-1, 1\} \\
x_2' &= x_2 + \delta\{-1, 1\} \\
y_2' &= y_2 + \delta\{-1, 1\} \\
z_2' &= z_2 + \delta\{-1, 1\} \\
\vdots\ &\ \vdots \\
x_M' &= x_M + \delta\{-1, 1\} \\
y_M' &= y_M + \delta\{-1, 1\} \\
z_M' &= z_M + \delta\{-1, 1\}
\end{aligned}$$

Figure 2.3: Movement of Walkers

The problem is then how to modify the rules of the walk so that the this occurs. We first show *how* the algorithm works and then discuss *why* it works.

The algorithm works as follows. Suppose that we wish to represent some multidimensional *pdf*, $g(\mathbf{R})$. For a walker, say at some point $\mathbf{R}$, we propose a random step (of a uniform limit, δ, in each coordinate) to some point $\mathbf{R}'$. For example, if $\mathbf{R}$ is a 3-dimensional variable for M particles then the movements are outlined in Figure 2.3. In this Figure {-1,1} denotes a random number in the range [-1,1]. Each occurrence of this symbol in these equations indicates a *different* random number.

After proposing the step in this manner, the value of the desired *pdf* is computed at the new and old positions[§], giving $g(\mathbf{R}')$ and $g(\mathbf{R})$ and the ratio

$$q \equiv \frac{g(\mathbf{R}')}{g(\mathbf{R})}$$

is formed. The step is then accepted with probability q. This means that, if q is greater than one, it is accepted. If q is less than one, a random number uniformly distributed between 0 and 1 is generated and only if q is greater than this number is the step accepted.

The step may be proposed in a number of ways, not just the one mentioned above. For example, in a finite interval problem the entire range might be used. In practical cases several walkers are carried along simultaneously in order to cover the space and so that an estimate of the value of the quantity being calculated can be made at each step.

As a simple example let us sample the *pdf*:

$$g(x) = \frac{\pi}{2}\sin \pi x \qquad 0 \le x \le 1.$$

We begin by initializing a set of walkers. In a real problem we would try to start them off in a distribution as closely resembling the expected final distribution as possible, but in this case we will simply take them to be uniformly distributed. We then need to initialize the values of the *pdf* at the "old" points to provide the denominator of q at the time of the first test. The code (without including binning for verification) might look something like:

```
      DIMENSION X(5000),G(5000)       ! room for 5000 walkers
      NWALK=5000                      ! number of walkers
      DELTA=.1                        ! maximum step size
      PI=ACOS(-1.)
      DO 1 I=1,NWALK                  ! initialize loop
      X(I)=RAN3(D)
    1 G(I)=SIN(PI*X(I))               ! normalization doesn't matter
      DO 2 K=1,100                    ! do 100 steps
```

[§] In practice the value of $g(\mathbf{R})$ will be available from the previous step

```
      DO 3 I=1,NWALK                     ! for each walker
      XT=X(I)+DELTA*(2.*RAN3(D)-1.)      ! propose new value
      IF (XT.GT.1.) GOTO 3               ! proposed step
      IF (XT.LT.0.) GOTO 3               ! must be in range
      GT=SIN(PI*XT)                      ! calculate proposed pdf
      Q=GT/G(I)
      IF (Q.LE.RAN3(D)) GOTO 3           ! accept or reject
      X(I)=XT                            ! if accept, update values
      G(I)=GT
    3 CONTINUE
    2 CONTINUE
      END
```

If we add a binning procedure to verify the behavior of the algorithm we have:

```
      DIMENSION X(5000),G(5000),IBIN(21),NTHEORY(21)
      NWALK=5000                     ! number of walkers
      DELTA=.1                       ! maximum step size
      PI=ACOS(-1.)
      DO 1 I=1,NWALK                 ! initialize loop
      X(I)=RAN3(D)
    1 G(I)=SIN(PI*X(I))               ! normalization doesn't matter
      DO 4 J=1,20                     ! bin starting values
    4 IBIN(J)=0
      DO 5 I=1,NWALK
      J=20.*X(I)+1.
    5 IBIN(J)=IBIN(J)+1
      DO 6 J=1,20
      XMIN=(J-1)*.05
      XMAX=J*.05
    6 NTHEORY(J)=NWALK*(COS(PI*XMIN)-COS(PI*XMAX))/2.
      PRINT 7,(J,J=1,20)
      PRINT 7,(IBIN(J),J=1,20)
      PRINT 7,(NTHEORY(J),J=1,20)       ! print binned results
    7 FORMAT(I3,19I4)
      DO 2 K=1,100                      ! do 100 steps
      DO 3 I=1,NWALK                    ! for each walker
      XT=X(I)+DELTA*(2.*RAN3(D)-1.)     ! propose new value
      IF (XT.GT.1.) GOTO 3              ! proposed step
      IF (XT.LT.0.) GOTO 3              ! must be in range
      GT=SIN(PI*XT)                     ! calculate proposed pdf
      Q=GT/G(I)
      IF (Q.LE.RAN3(D)) GOTO 3       ! accept or reject
      X(I)=XT                        ! if accept, update values
      G(I)=GT
    3 CONTINUE
      IF (MOD(K,20).NE.O) GOTO 2     ! bin every 20 steps
      DO 14 J=1,20                    ! start binning
   14 IBIN(J)=0
      DO 15 I=1,NWALK
      J=20.*X(I)+1.
   15 IBIN(J)=IBIN(J)+1
      PRINT 7,(J,J=1,20)
      PRINT 7,(IBIN(J),J=1,20)
      PRINT 7,(NTHEORY(J),J=1,20)    ! print binned results
    2 CONTINUE
      END
```

2.4.2 Why It Works

To understand why the Metropolis algorithm works, think of the flow of probability like the flow of something material. There are some areas more dense than others with a constant flux of material between the areas. There are many possible scenarios for creating flow such that the equilibrium distribution is the one that represents $g(\mathrm{R})$. Our problem is easy, in a sense, because we only need to find one method to control the flow of probability in this dynamic equilibrium such that we reach a given steady state. Actually that is not quite true since, among all of the possible choices, we would like to find the one which would reach the equilibrium limit the most quickly.

Consider the conditions which must prevail when the equilibrium state is reached. If $P(\mathrm{R}',\mathrm{R})$ is the probability of a flow from R to R′ and $g(\mathrm{R})$ the *pdf* at R then, since the large probability of flow from the low density to high density must match the small probability of flow from high density to low density (so that the densities remain constant), we must have:

$$P(\mathrm{R}',\mathrm{R})g(\mathrm{R}) = P(\mathrm{R},\mathrm{R}')g(\mathrm{R}') \qquad (2.58)$$

This equation simply states that the rate of flow toward high density over the rate of flow away from high density is the inverse of the ratio of the densities.

We can now break the probability $P(\mathrm{R}',\mathrm{R})$ into the product of two factors corresponding to a conditional probability. We shall express this probability by first proposing the change $\mathrm{R} \rightarrow \mathrm{R}'$ and then making a test to see if the proposed change

is accepted. Thus we will rewrite

$$P(\mathrm{R}',\mathrm{R}) = A(\mathrm{R}',\mathrm{R})T(\mathrm{R}',\mathrm{R}) \qquad (2.59)$$

where $T(\mathrm{R}',\mathrm{R})$ is the probability that we propose a point near R′ starting at R and $A(\mathrm{R}',\mathrm{R})$ is the probability that such a "move" will be accepted. Any combination "AT" which satisfies Eq. 2.58 will work, some may work faster than others, some may be easier than others to implement. To simplify our task we choose

$$T(\mathrm{R}',\mathrm{R}) = T(\mathrm{R},\mathrm{R}'). \qquad (2.60)$$

The balance equation 2.58 is now reduced to:

$$A(\mathrm{R}',\mathrm{R})g(\mathrm{R}) = A(\mathrm{R},\mathrm{R}')g(\mathrm{R}'). \qquad (2.61)$$

It only remains for us to pick a function "A" which satisfies this equation. It is useful to define the quantity,

$$q(\mathrm{R}',\mathrm{R}) = \frac{g(\mathrm{R}')}{g(\mathrm{R})}. \qquad (2.62)$$

Consider the function:

$$A(\mathrm{R}',\mathrm{R}) \equiv \min[1, q(\mathrm{R}',\mathrm{R})]. \qquad (2.63)$$

By examining Eq. 2.61 for the case that $q < 1$ and $q > 1$ it can be seen that is satisfied in both situations. It is also seen that the condition for acceptance is indeed the one outlined in the previous section.

There is an alternate form:

$$A(\mathrm{R}',\mathrm{R}) \equiv \frac{q(\mathrm{R}',\mathrm{R})}{1+q(\mathrm{R}',\mathrm{R})} \qquad (2.64)$$

which also works.

2.4.3 Comments on the Algorithm

One thing to note is that the normalization of the *pdf* never enters into the equations. This is convenient in that it is not necessary to know specifically how to normalize the functions. It is very useful if one is dealing with a multidimensional probability densities having a complicated form. In fact, it is exactly in this instance that the method finds its greatest utility. However, if we wish to compute an integral which is not really in the form of Eq. 2.2, then we must know the normalization by other means. For example, if we wished to do the integral in Eq. 2.2 by sampling the entire integrand, as we did in Section 2.3.9, we would simply obtain unity. Of course that is what we obtained in that instance for the sum, but in that case the sampling method was capable of returning the normalization of the function at the same time since it was a rejection technique. This feature of the Metropolis algorithm is a great benefit or a difficulty depending on the type problem to be solved.

The vector containing the walkers represents the *pdf* in the sense that more walkers are concentrated in the region where the function is large. This can be a very useful way to represent a function. For simple functions it may seem like a poor (and far-fetched) expression but for multidimensional functions it is very efficient. If we wish to average over a function, the values are only needed where it is large. The usual representation *at all points in the space* becomes incredibly wasteful. Consider a mesh of only 10 points in each dimension. For thirty particles in three dimensions we would need 10^{90} tabulated values. This number far exceeds any memory one could expect to have (indeed it far exceeds the number of particles in the universe). With the Monte Carlo representation, problems with the order of a thousand particles can be solved. The direct utility of this representation of functions is limited by the fact that they can be used only to calculate integrals. There are, however, special techniques which can be employed to calculate the desired quantities with these functions. These techniques are treated in Chapter 11.

Problems

Problems marked with * are meant to be solved without the aid of a computer.

1. Code the Monte Carlo solution for the area of a circle to find π. Extend it to 3 dimensions to get π from the volume of a sphere $\frac{4\pi}{3}$. If you use RAN3 call it several times between usages for the second part of this problem to avoid the correlations among 3 successive values.

2. Integrate x, x^2, x^3, x^4 and x^5 from 0 to 1 using Monte Carlo with x chosen uniformly from 0 to 1.

3. Generate distributions for $h_n(x) = (n+1)x^n \quad 0 \le x \le 1$, put them in bins and compare with the expected *pdf* for $n = 1$, 2 and 3.

4. Perform the integral:

$$\int_0^1 x^5 dx = \frac{1}{n+1}\int_0^1 h_n(x)x^{5-n}dx$$

for $n = 0, 1, 2, 4, 5$ by Monte Carlo. Compare the relative accuracy by examining the variance computed from Eq. 2.7

5. * Write an equation to sample x from the normalized probability distribution function

$$g(x) = \tfrac{3x^2}{(1+x^3)^2} \qquad 0 \le x \le \infty.$$

Note that

$$\int_0^x g(x)dx = \frac{x^3}{1+x^3}.$$

6. * Derive the normalized form, the cumulative distribution function, $F(x)$, and the sampling equation for:

$$g_3 : \quad \int \frac{xdx}{(a^2+x^2)^{n+1}} = -\frac{1}{2n(a^2+x^2)^n}$$

and

$$g_4 : \quad \int \frac{dx}{(a^2+b^2x)x^{\frac{1}{2}}} = \frac{2}{ab}\tan^{-1}\frac{bx^{\frac{1}{2}}}{a}.$$