

Unit XIII

Monte Carlo Methods

These materials were developed for use at Grinnell College and neither Grinnell College nor the author, Mark Schneider, assume any responsibility for their suitability or completeness for use elsewhere nor liability for any damages or injury that may result from their use elsewhere.

Unit XIII Monte Carlo Methods

In our first unit, we used random numbers to investigate the propagation of errors. What we did, in essence, was to use random numbers to generate artificial data. This practice is used heavily in physics, and it is almost an unofficial rite of passage that every graduate student of physics must write at least one Monte Carlo simulation program before completing a thesis. The use of random chance gives rise to the term Monte Carlo in reference to the famed mecca of games of chance in Monaco, however the glamour fades before we get much past the name. Usually Monte Carlo methods are used for calculating things that are otherwise incalculable, in particular complicated systems, where we have to average or integrate over a burdensome number of degrees of freedom. Koonin (Computational Physics, p185) puts it very persuasively:

Systems with a large number of degrees of freedom are often of interest in physics. Among these are the many atoms in a chunk of condensed matter, the many electrons in an atom, or the infinitely many values of a quantum field at all points in a region of space-time. The description of such systems often involves (or can be reduced to) the evaluation of integrals of very high dimension. For example, the classical partition function [an intermediate calculational tool in thermodynamics] for a gas of A atoms . . . is proportional to [a] . . . $3A$ -dimensional integral The straightforward evaluation of an integral like this by one of the quadrature formulas . . . is completely out of the question except for the very smallest values of A . To see why, suppose that the quadrature allows each coordinate to take on 10 different values (not a very fine discretization), so that the integrand must be evaluated at 10^{3A} points. For a modest value of $A = 20$ and a very fast computer capable of some 10^7 evaluations per second, this would take some 10^{53} seconds, more than 10^{34} times the age of the universe.

For these sorts of calculations, it actually becomes more efficient to discard any common sense about where to sample the function we are trying to integrate, and in fact just sample it completely randomly. We'll see a couple of different ways of accomplishing this, and then see how one integrates a potentially infinite-dimensional integral in a process usually termed a Monte Carlo simulation.


Guidebook Entry XIII.1: A Straightforward Integral

As our first Monte Carlo task, we will attempt to integrate a simple function that we can do easily analytically. This is clearly not a good application for Monte Carlo techniques, but should be edifying. The integral we will evaluate is

$$\int_0^2 (x^2 + 2) dx.$$

What is the analytical value of this definite integral?

Now to evaluate this by Monte Carlo techniques, we will choose a large number (say 100 for starters) of x values randomly distributed between 0 and 2, and calculate $f(x) = x^2 + 2$ at each of these x values. Use Excel to do this: make a column that simply numbers the random x values 1, 2, 3, etc., a column of the x values themselves, and a column of the $f(x)$ values. Finally, find the average $f(x)$ value, and multiply that times the integration width of 2. How does this compare with the analytical value?

Now, let's consider convergence issues a bit. Let's look at the cumulative sum. Create a new column that is equal to the sum of $f(x)$ values above it, so the n 'th entry is the sum of the first n $f(x)$ values. Create another column to the right of this which is the estimate of the integral at each point, that is, the cumulative sum times 2 divided by n . Graph this final column as a line chart. You should observe that the integration appears to converge, but not always to the actual value! Of course, it will, eventually, if the random number generator is good, but if it has a bad start, it may find it hard to make up for that. Try resetting the random numbers a few times by hitting  a few times. Describe how the graph changes.

Now, to get a handle on how well we can trust the values, in contrast to how well it seems to converge, we should repeat the calculation a number of times, and take the standard deviation of the set of results. Make a number (at least 10) of similar columns in your sheet, and take the standard deviation of the resulting integral values. Are your values clustered around the correct value within about one standard deviation?

Hit \bar{a} again, and check your results again. Is the standard deviation about the same? Do the values cluster around the real value within about a standard deviation?

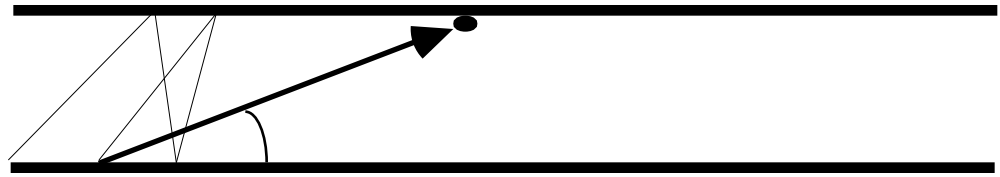
Now repeat this experiment with 10 times as many numbers in each column. How much better does the integral become as measured by the standard deviation? Can you guess how the integral uncertainty depends on the number of random points you take?

This is a consistent theme with Monte Carlo calculations; the accuracy improves like the square root of the number of iterations involved in the calculation. This fact provides a very straightforward way of estimating the accuracy of any calculation; repetition provides the standard deviation, which we assume drops like the square root of the number of iterations and can therefore extrapolate to an error estimate on our final, best value.

In this next example, you will perform an "integration" that is much more open-ended, where the number of dimensions could be infinite. It is much more typical to think of this sort of calculation as a simulation rather than an integration, although integrations like this can be done analytically as well.

Guidebook Entry XIII.2: Gas Flow in Two Dimensions

In this exercise, we imagine that we have individual atoms moving through a pipe. We will consider the pipe here to be two dimensional for ease of calculation, although it is not much more difficult in three dimensions. The atom starts at the left end, and each time it hits a wall, it suffers a direction-randomizing collision.



Assume the width of the channel is 1, the length of the channel is 10, and the velocity of the atom is 1.

How long does it take for the atom to get across to the other side of the channel, given v and the angle θ ?

How far down the channel does the atom move before hitting the opposite wall?

Now create a spreadsheet that calculates this motion. Create columns for position, angle, parallel velocity, and perpendicular velocity.

Start the position at 0.

Select the first angle to be randomly distributed from $-\pi/2$ to $\pi/2$. Calculate the parallel and perpendicular velocity components. Assume the atom starts in the middle of the channel. What should the next position be?

Now generate a new random angle for the new direction after the collision, only now ranging from 0 to π . Fill in the velocity components, and update the next position appropriately.

Now this is actually started, you may fill the columns down.

Now for the interpretation. To find out how quickly atoms move through this "tube," we want to know the ratio of atoms that get out the far end to the total number started. If the position becomes negative before it reaches $x = 10$, this means the atom came back out the same end it started; if it reached 10 without ever going negative, it successfully exited the far end.

Look at your sheet. What is the fate of this particular atom?

Now reset your random numbers with a $\text{Apple} =$, which is equivalent to starting a new atom. How about the fate of this one?

Run 100 atoms. What fraction get out?

Thus far we have used random numbers distributed uniformly over an interval. However, it is often necessary to generate random number distributed according to a probability distribution. One general technique for doing this is attributed to Von Neumann. It goes as follows. We wish to generate x values consistent with a probability distribution $f(x)$, which has a maximum value of 1. Generate two random numbers, the first scaled to cover the range you desire. Take the second one, and compare that to f evaluated at the first random value. If f is greater than

the second random number, keep the first random number. If f is less than the second random number, throw away the first random number, and start again. This works reliably to generate numbers according to any distribution, but has the obvious disadvantage that you have to generate many more numbers to get the job done.

The Von Neumann technique for generating this non-uniform distribution is particularly inefficient if the probability distribution is strongly peaked, but with wide (and important) tails. An example of this is if one is generating random penetration distances of gamma or x rays in material, where the distance traveled is distributed according to an exponential distribution, which runs, in principle, to infinity. In the next activity, we will develop an efficient technique for generating such random numbers.

Guidebook Entry XIII.3: Generating Exponentially Distributed Interaction Lengths.

We start with the assumption that interaction distances are distributed randomly, but proportional to the distribution e^{-kx} . In other words, a gamma ray is traveling through some material until it hits an atom and interacts (say by photoelectric effect). If we made a histogram of distances traveled from emission to interaction (number of incidences versus distance), we would develop a curve that was a decreasing exponential with a decay constant of k . We can also say that the probability of any given gamma ray having an interaction distance between s and $s + \Delta s$ is proportional to the integral

$$\int_s^{s + \Delta s} e^{-kx} dx$$

First, show that the *normalized* distribution (one whose integral over the full range of interest from zero to infinity is one) is given by

$$P(x) = ke^{-kx}.$$

We would like to stretch out the random number distribution so that they match this exponential distribution. For example, we would like to have a mechanism such that whenever we get a random number in the lower half of the generator range (i.e. between 0 and 0.5) we generate a distance somewhere between 0 and $s_{1/2}$ defined implicitly by the relationship

$$\int_0^{s_{1/2}} ke^{-kx} dx = 1/2.$$

Solve this for $s_{1/2}$.

Now that you have the general idea, convince yourself that the way to stretch your random numbers into the correct probability distribution is that a random number RND generated uniformly in the range from zero to one gives a random distance s_r given implicitly by

$$\int_0^{s_r} P(x) dx = RND.$$

Rearrange this expression for our $P(x)$ to find s_r explicitly in terms of RND.

I have written this last expression in a way that allows us to use it for any probability distribution we wish to mimic. Sometimes we end up with an expression that is difficult to invert, or we have a function that we don't know how to integrate. If this is the case, we can always take the route of creating a rough table of the cumulative integral numerically. We then can use some of the interpolation techniques we learned early in the semester to interpolate these values to find the correct value associate with a generated random number. This may also be quicker than trying to evaluate a complicated analytical expression with a lot of transcendental functions.

In this final exercise, we will use Excel and our results from the previous activity on interaction lengths to calculate the efficiency of a gamma ray detector. In all likelihood, if one wanted to know something like this with precision, Excel or Mathematica become prohibitively slow, and one has to resort to programming in a compiled language such as C or Pascal, although all of the techniques we have used here in Excel are easily implemented in the compiled language as well.

Guidebook Entry XIII.4: A Detector Efficiency Calculation

Suppose we wish to calculate how efficiently a detector counts gamma rays. Let's imagine that the gamma rays are emitted from a point that we will choose as our coordinate system origin. The detector we'll take to be a cube of length 2 on a side, so it extends from -1 to 1 in each dimension.

The strategy will be to generate a random direction for the gamma ray, then generate a random distance for the gamma to travel through the detector

material before interacting. We find the location of this point in space, and if it is within the detector volume, then we say the gamma was detected; if not, the gamma must have escaped.

First, somewhere up at the top of your sheet, save a spot to keep k , the interaction constant in our distance distribution.

Now we need to generate the random direction. We can generate angles if we wish, but those are not uniformly distributed. A standard trick is to generate a random vector in a unit cube by generating three random numbers in the range -1 to 1. We then check to see if this vector lies within the unit sphere. If it does, we take it and normalize it to produce a unit direction vector. If it doesn't, we throw it away. In Excel it is a little difficult to say start over again to produce a new vector, so we will just set a logical value (actually a number) to tell us whether to keep this whole simulated event or not.

So, to get you started, create the following columns: x , y , z , r , and test column you can label "good vector" or something like that. The entries under x , y , and z give the random numbers from -1 to 1, r is just given by

$$r = \text{SQRT}(x^2 + y^2 + z^2)$$

(where x, y , and z are actually cell labels). The test for a good vector is most easily done using the IF command, whose syntax is like

$$=IF(r < 1, 1, 0)$$

which returns a 1 if the logical test $r < 1$ is true, or a zero if the test is false.

Now add three more columns that produce normalized vector components, x_{norm} , y_{norm} , and z_{norm} . What are the expressions for these unit vectors?

Don't worry about the "bad" vectors, we'll throw them away later.

Now use the expression you got in GE XIII.3 to change a random number into a random interaction distance with our exponential character. Put this in the next column, which you might label s .

Now you can calculate the final positions of interaction, labeled x_{final} , y_{final} , and z_{final} . What is an expression for one of these? Enter them all into your sheet.

Now check to see if the final position is within the detector volume, that is checking to see if

$$ABS(x_{\text{final}}) < 1$$

and similar expressions for y and z. You can put the result of this in an "in detector" column using the IF command again. You may wish to use the logical AND command, which has the cumbersome syntax

AND(logic expression 1, logic expression 2)

and yes, you will have to use that twice to check all three dimensions.

Finally, let's do the statistics. We want to know how many of the good vectors give rise to "in detector" events. The easiest way to do this is to create another column, call it "good and in", which is the product of "good vector" and "in detector" and is therefore 1 only if we have a good vector that ends up in the detector. Now we can fill this whole row of entries down (maybe 100 rows?) and look at the efficiency, which is the sum of the "good and in" column divided by the "good vector" column. What sort of efficiency do you get when $k = 1$?

How big an error would you ascribe to your efficiency calculation? Explain how you determined it.

A few minor comments. It is easy to see how this case would be difficult (although not impossible) to integrate directly. It becomes much more difficult to integrate if we place the source off the center, or distribute it, or if we make the shape of the detector a little odd, or if we allow several different interaction processes, some of which force us to change direction and the k value after each interaction. All of this is completely realistic, and sometimes important to model.

You may wonder if it really makes any difference to throw away the initial directions that didn't fall into the sphere--they still are random after all. In this particular case it is very important to do so. The direction vectors that we discard tend to fall in the corners obviously, which means they are directed toward a corner of the detector, and therefore have more material to pass through. If we fail to discard them, we will overestimate the efficiency of our detector noticeably.

Finally, a possible shortcut to take. Because of the symmetry of this system, we really only need to generate directions into the eighth of a sphere given by x , y , and z all positive. This saves a little time in generating the events, and then you don't need to take absolute values when you check to see if you are in the detector.

Seems pretty minor, and in this case is dwarfed by other calculations such as the square roots and the logarithms, but Monte Carlo programs often become quite complicated and are limited by computing time and patience; one can easily produce a sloppy program that will require a year or a lifetime to produce a usable result, when a few observations of symmetry can sometimes save a great deal of computing time and make a seemingly impossible computation possible.