5. Monte Carlo integration

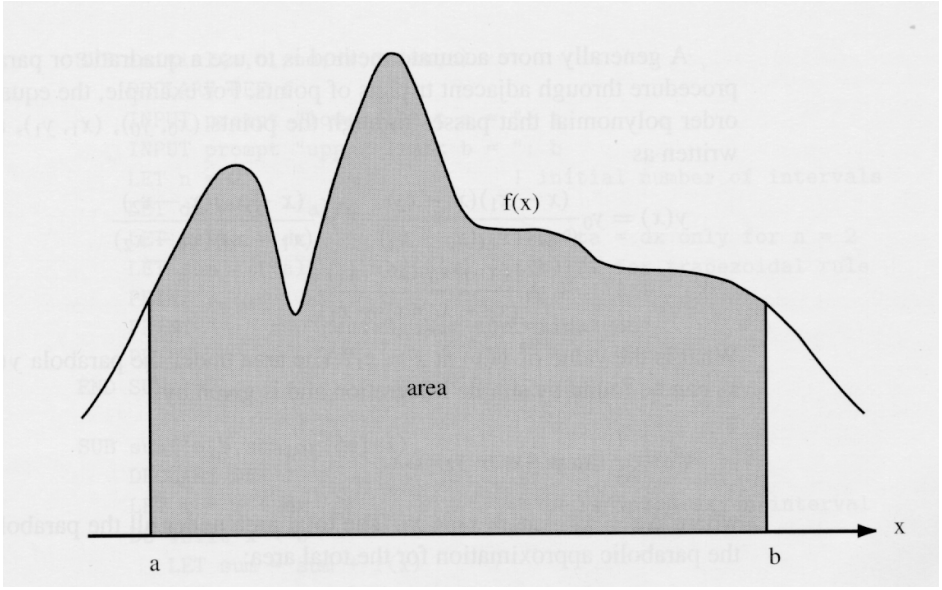
One of the main applications of MC is integrating functions.

At the simplest, this takes the form of integrating an ordinary 1- or multidimensional analytical function. But very often nowadays the function itself is a set of values returned by a simulation (e.g. MC or MD), and the actual function form need not be known at all. Most of the same principles of MC integration hold regardless of whether we are integrating an analytical function or a simulation.

5.1. MC integration

[Gould and Tobochnik ch. 11, Numerical Recipes 7.6]

To get the idea behind MC integration, it is instructive to recall how ordinary numerical integration works. If we consider a 1-D case, the problem can be stated in the form that we want to find the area A below an arbitrary curve in some interval $[a, b]$.



In the simplest possible approach, this is achieved by a direct summation over N points occurring

at a regular interval Δx in x:

$$A = \sum_{i=1}^{N} f(x_i)\Delta x \tag{1}$$

where

$$x_i = a + (i - 0.5)\Delta x \quad \text{and} \quad \Delta x = \frac{b-a}{N} \tag{2}$$

i.e.

$$A = \frac{b-a}{N} \sum_{i=1}^{N} f(x_i)$$

This takes the value of f from the midpoint of each interval. Of course this can be made more accurate by using e.g. the trapezoidal or Simpson's method. But for the present purpose of linking this to MC integration, we need not concern ourselves with that.

In M dimensions, the generalization of this is for an interval $([a_1, b_1], [a_2, b_2], \ldots, [a_M, b_M])$ giving an (M+1)-dimensional volume $V^{(M+1)}$

$$V^{(M+1)} = \frac{(b_1 - a_1)(b_2 - a_2)\cdots(b_M - a_M)}{N_1 N_2 \cdots N_M} \sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} \cdots \sum_{i_M=1}^{N_M} f(\mathbf{x}_i)$$

where

$$\mathbf{x}_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_M})$$

is the M-dimensional vector and each x_i is defined as above in Eq. (2). This can be rewritten as

$$V^{(M+1)} = \frac{V^{(M)}}{N} \sum_{i_1=1}^{N_1} \sum_{i_1=1}^{N_2} \cdots \sum_{i_M=1}^{N_M} f(\mathbf{x}_i) = V^{(M)} \frac{\sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} \cdots \sum_{i_M=1}^{N_M} f(\mathbf{x}_i)}{N} \tag{3}$$

where $V^{(M)}$ is the M-dimensional volume defining the integration "area", and N the total number of points. The latter form shows that this can be interpreted simply as taking the average over f in the interval in question, i.e. this can be written also as

$$V^{(M+1)} = V^{(M)} \langle f \rangle \tag{4}$$

where $\langle\rangle$ denotes the average,

$$\langle f \rangle = \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N}$$

5.1.1. Sampling method

The sampling method for MC integration is very similar to the simple summing rules 1 and 3 given above. Instead of sampling at regular intervals Δx, we now sample at random points, and then take the average over these.

Say we pick N points x_i in the interval $[a, b]$ in 1D. The integral then becomes

$$A = \frac{b-a}{N} \sum_{i=1}^{N} f(x_i)$$

which is identical to Eq. 1.

More generally, in M dimensions we have to pick vectors

$$\mathbf{x}_i = (x_1, x_2, \ldots x_M)$$

at random in the interval $([a_1, b_1], [a_2, b_2], \ldots, [a_M, b_M])$ which can be done very easily using uniform random numbers for each dimension at a time. Having N such points, the MC estimate of

the $(M+1)$-dimensional volume below the M-dimensional function $f(\mathbf{x})$ is then

$$V^{(M+1)} \approx V^{(M)} \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N} = V^{(M)} \langle f \rangle \tag{5}$$

where the latter form emphasizes the similarity to the numerical integration.

How does the MC integration work in practice? Let us consider getting the volume of a sphere of radius r. In this case, our region of integration is a circular area of radius r in the xy plane, and the function is easily derived from

$$r^2 = x^2 + y^2 + z^2 \Longrightarrow z = \sqrt{r^2 - (x^2 + y^2)}$$

i.e.

$$f(x, y) = \sqrt{r^2 - (x^2 + y^2)}$$

Integrating this will give half the volume of the sphere. We will do this by selecting points randomly in a square with the range $([-r, r], [-r, r])$, reject those which are outside the circle of radius r, then do the MC sum for the points inside.

In this 2-dimensional case, we could also make a routine which generates points directly within the circle, with no need for a rejection step. But this becomes increasingly complicated in higher dimensions, and will hence not be done.

Here is a code which does the integration. The random number generator is `ran2` identically copied from Numerical Recipes, and hence not written out here. The code is pretty much self-explanatory.

```
#include <math.h>
#include <stdio.h>

main()
{
  int seed=45629;
  float pi=3.14159265358979238;

  int npoints=100000;
  int n,npointsinside;
  float x,y,r,sq,f,fsum,fmean,I;

  float ran2(); /* Random number generator provided */
```

```
r=1.0;
fsum=0.0;
npointsinside=0;
for(n=0;n<npoints;n++) {
  x=2.0*ran2(&seed)-1.0;
  y=2.0*ran2(&seed)-1.0;
  /*
    Evaluate function i.e. calculate sqrt(r^2-(x1^2+x2^2+...+xn^2))
    but only for points inside the 2D circle
  */
  sq=x*x+y*y;
  if (sq < r*r) {
    f=sqrt(r*r-sq);
    fsum+=f;
    npointsinside++;
  }
}
if (npointsinside==0) {
  printf("No points inside. Increase npoints\n");
  exit(0);
}
```

```
  /* MC estimate of <f> */
  fmean=fsum/npointsinside;

  /* Actual integral: 2 V <f> ; volume is now pi r^2 */
  I=2*pi*r*r*fmean;

  printf("Sphere volume is %.6f hits %d\n",I,npointsinside);

}
```

Running the code gives

```
beam.helsinki.fi tests> cc 3Dsphere.c -lm
beam.helsinki.fi tests> a.out
Sphere volume is 4.182319 hits 78575
```

so the answer is quite close to the correct one, $4\pi/3 = 4.18879020$. We will below see how the uncertainty can be estimated.

5.1.1.1. When and why is MC better than numerical integration

Comparison of the sums in Eqs. 3 and 5,

$$V^{(M+1)} = \frac{V^{(M)}}{N} \sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} \cdots \sum_{i_M=1}^{N_M} f(\mathbf{x}_i) \tag{6}$$

vs.

$$V^{(M+1)} \approx \frac{V^{(M)}}{N} \sum_{i=1}^{N} f(\mathbf{x}_i) \tag{7}$$

illustrates a crucial difference: in numerical integration, we need M different sums, but in MC integration only one is enough! This leads us to understand why MC integration is so important in many dimensions. In 1D there really is no major difference, and indeed using methods like Simpson's the conventional numerical integration can easily be made quite accurate and much more efficient than MC integration. But with increasing numbers of dimensions M, doing the M sums becomes increasingly cumbersome, and eventually using the MC approach which only needs one sum will clearly be simpler.

To be more specific about the cumbersomeness, for numerical integration we will always need at least a few points N_i per dimension to get a sensible answer, so the number of summing steps increases as N_i^M. If e.g. $N_i = 5$ for all i (a very low value!), then in 10 dimensions we need $5^{10} \approx 10$ million points to get any kind of sensible answer. But for MC integration we can use the same number of points N for any number of dimensions.

To illustrate this, I did the following test. I calculated the volume of a sphere in M dimensions with direct numerical integration (using the midpoint method) and MC integration. The number of intervals was 20 in the numerical integration in each dimension, and the number of attempts in the MC simulation was always 10^5. This happened to give results of comparable, about ~ 0.5 % accuracy. I timed the result simply with the Unix `time` command.

The results are as follows. The first column gives the number of dimensions M, the next two the numerical execution time, the next two the MC results in the same way, and the last column the correct answer (known analytically). The times are in seconds.

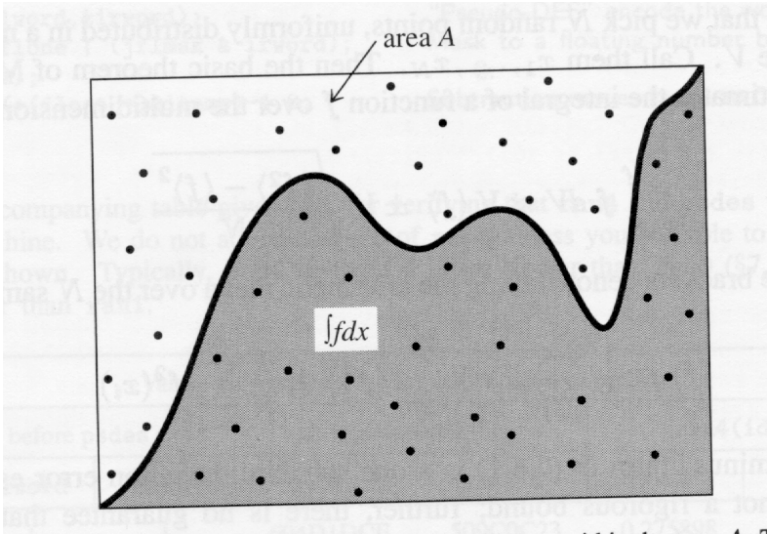
	numerical		MC		
M	time	result	time	result	Correct
2	0.00	3.1524	0.01	3.1435	3.1415
3	0.00	4.1737	0.07	4.1896	4.1887
4	0.00	4.9023	0.08	4.9330	4.9348
5	0.02	5.2381	0.10	5.2787	5.2637
6	0.30	5.1451	0.13	5.1748	5.1677
7	5.02	4.6704	0.15	4.7098	4.7247
8	89.9	3.9595	0.17	4.0479	4.0587
9	1320	3.3998	0.20	3.3191	3.2985

So we see that for $M < 6$ the numerical method is faster, but after that becomes terribly

much slower. What is most interesting is that the time required by the MC method is not rising almost at all, even though the accuracy stays the same. This is what makes it so interesting for high-dimensional integration.

5.1.2. Hit and miss method

There is another approach to MC integration, which is even simpler than the sampling approach. It is essentially the same as the hit-and-miss method used to generate random numbers in a nonuniform distribution. The idea is that we find some region in space of known volume, which encloses the volume we want to integrate, then generate random points everywhere in this region, and count the points which actually do hit the volume we want to handle:



Say the volume of the external region is V_e and the fraction of hits is f_h. Then the volume of the region to be integrated is simply

$$V = V_e f_h$$

This method is actually equivalent to the previous one. This can be understood as follows. Say we are working in M dimensions, the number of trial points is N, and the number of hits is N_h. Then the above equation can be rewritten as follows:

$$V = V_e f = V_e \frac{N_h}{N} = V_e \frac{\sum_{i=1}^{N_h} 1}{N}$$

If we further define an M-dimensional function

$$f(\mathbf{x}) = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is inside the volume} \\ 0 \text{ elsewhere} \end{cases}$$

we can write this as

$$V = V_e \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N}$$

Now we see that this is identical to eq. (5) ! In fact since the average of f is just the fraction of hits f_h, we can simply write this as

$$V = V_e f_h = V_e \langle f \rangle \tag{8}$$

The only difference is that because f now is dimensionless, V and V_e have the same dimension.

5.2. Error analysis of MC integration

In any scientific study where statistics is collected (which is most of experimental and computational science), it is of utmost importance to be able to calculate not only the average of something, but also the uncertainty of the average. For the MC simulation methods described above this can be done as follows.

5.2.1. Sampling method

For the **sampling** method, the error can be obtained very simply. Remember that we consider the volume calculations as a calculation of the $average$ of the f function, $\langle f \rangle$. Then it is natural to calculate the error as the **error of the average** over the sampled values of f, as this is usually done. The general equation for the error of the average $\sigma_{\bar{x}}$ of a set of points x_i is [Pentikäinen]

$$\sigma_{\bar{x}} \approx \frac{\sigma}{\sqrt{N}}$$

where the variance σ^2 is obtained from

$$\sigma^2 = \frac{1}{N-1}\left[\left(\sum_{i=1}^{N} x_i^2\right) - N(\bar{x})^2\right]$$

Combining these and assuming $N >> 1$

$$\sigma_{\bar{x}} \approx \frac{1}{\sqrt{N}}\sqrt{\frac{\sum_{i=1}^{N} x_i^2}{N} - (\bar{x})^2}$$

Now for MC integration the points x_i are actually the values of the function f! Using the same

notation as above for the average,

$$\langle f \rangle = \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N} \qquad \text{and } \langle f^2 \rangle = \frac{\sum_{i=1}^{N} f^2(\mathbf{x}_i)}{N}$$

we thus get that the error of the MC integration is

$$\sigma_{V\langle f \rangle} \approx V \sqrt{\frac{\overline{\langle f^2 \rangle - \langle f \rangle^2}}{N}}$$

and, to reiterate, the whole equation for the MC integration

$$\int f dV \approx V\langle f \rangle \pm V \sqrt{\frac{\overline{\langle f^2 \rangle - \langle f \rangle^2}}{N}} \tag{9}$$

Note that this gives the so called **one sigma** (1 σ) error. This means that if the data would be distributed in a Gaussian distribution, the probability that the true value is within the one sigma error is about 2/3 (68.3 % to be more precise). One can also correspondingly give broader confidence intervals, 2 σ, 3 σ etc. with increasingly large probabilities that the true value is within the measured value plus minus its error bar. (FIGURE DRAWN ON LECTURE)

From this we also see why MC integration is particularly useful in multidimensions - the accuracy increases as $\sqrt{N}$, but there is no dependence on the number of dimensions here! So to get comparable accuracy, one only needs to collect the same number of points N regardless of dimension. This is in clear contrast to direct numerical integration, where the number of points needed increases as

$$N_1^d$$

where d is the number of dimensions and N_1 the number of points needed in one dimension.

5.2.1.1. Caution: this may not be right

But there are two possible problems with this error estimate.

The first is that **there is absolutely no guarantee the f points do have a Gaussian distribution**, and hence the error given by the equation (9) should be understood as only a rough idea of what level of uncertainty may be expected.

A better estimate of error can be obtained if it is possible to collect enough $f(\mathbf{x}_i)$ points to see what shape the distribution actually has, then analyze the error behaviour of this distribution. If it is not some well-known distribution, it is always possible to use MC simulation of the data distribution to deduce how the error should be calculated; this will be discussed in the next section.

The second problem is that **even if the data does have a Gaussian distribution, Eq. 9 is not right if N is very low!** The correct error is obtained first by calculating the function averages using $N-1$ instead of N in the denominator,

$$\langle f \rangle = \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N-1} \qquad \text{and } \langle f^2 \rangle = \frac{\sum_{i=1}^{N} f^2(\mathbf{x}_i)}{N-1}$$

then calculating the error of the average:

$$\sigma_{V\langle f \rangle} \approx V \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}}$$

then multiplying the error thus obtained with the so called Student's dilatation factor. It is [Laurikainen's data analysis notes]

N :	2	3	4	6	11
Student's dilatation factor:	1.84	1.32	1.20	1.11	1.05

We see that this correction is probably needed essentially only when $N < 10$ (an error of an error of less than 5% is almost certainly not meaningful in most contexts). Since in most MC simulations N certainly is larger than 10, this is not a problem. Note, however, that if you look at larger confidence intervals (2σ, 3σ etc.) the correction becomes significantly larger even for larger N.

Unfortunately in many cases the shape of the f function is not known, and evaluating it is so slow that it can be sampled at so few points (a few ten or even less) that it is not possible to deduce what distribution it actually has. In such cases, the most common way to go is to simply use equation (9) and hope it is in the right ballpark. Of course if the magnitude of the error bars is crucial for supporting the conclusions you want to draw, this is not acceptable!

In published work, the general trend seems to be that if nothing is said about the way the error is calculated, it is implicit that the errors given are 1 σ errors calculated assuming Gaussian statistics.

Most physicists probably are not even aware of the possible problems with this, so now that you are, you already are doing better than the majority!

5.2.2. Hit and miss method

The hit-and-miss method gave the volume of an integrated area simply as

$$V = V_e f = V_e \frac{N_h}{N}$$

where N_h is the number of hits, and N the number of trials.

To find the error of the hit and miss method, we utilize our observation that the hit-and-miss method is actually just a variety of the sampling method. Eq. (8):

$$V = V_e f_h = V_e \langle f \rangle$$

where f was our artificial function:

$$f(\mathbf{x}) = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is inside the volume} \\ 0 \text{ elsewhere} \end{cases}$$

So the error is now as above,

$$\sigma_{V_e \langle f \rangle} \approx V_e \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}}$$

with

$$\langle f \rangle = \frac{\sum_{i=1}^{N} f(\mathbf{x}_i)}{N} \qquad \text{and } \langle f^2 \rangle = \frac{\sum_{i=1}^{N} f^2(\mathbf{x}_i)}{N}$$

But taking into account the simple form of f we see that this can be simplified, $\langle f \rangle$ is just N_h/N, and so is $\langle f^2 \rangle$! Hence the error becomes

$$\sigma_{V_e\langle f \rangle} \approx V_e \sqrt{\frac{(N_h/N) - (N_h/N)^2}{N}} = V_e \sqrt{\frac{N_h - N_h^2/N}{N^2}} = V_e \frac{\sqrt{N_h - N_h^2/N}}{N}$$

and we can write the equation for the integrated volume and its error in the hit-and-miss method as

$$V = V_e \frac{N_h}{N} \pm V_e \frac{\sqrt{N_h - N_h^2/N}}{N} \tag{10}$$

Finally, if $N_h << N$ we can simplify further to get

$$V = V_e \frac{N_h}{N} \pm V_e \frac{\sqrt{N_h}}{N} \qquad \textbf{if} \qquad N_h << N$$

5.3. Utilizing nonuniform random numbers

Above we assumed the random numbers are generated in a M-dimensional box. But in case the volume to be integrated fills only a small fraction of that box, the fraction of misses can be enormous. In that case it may be highly advantageous to find some other distribution which better encloses the volume we are integrating over, and generate random numbers only in this. If the enclosing function is such that we know how to generate random numbers in this distribution analytically, the savings in time can be enormous.

This is actually exactly what was done in the previous section when we discussed the **combined analytical-rejection method** to generate random numbers. The algorithm there was

1° Generate a uniformly distributed number $u = P_u(0,1)$

2° Generate a number distributed as $g(x)$: $x = G^{-1}(u)$

3° Generate a uniformly distributed number $y = P_u(0, ag(x))$

4° If $y > f(x)$ this is a **miss:** return to 1°

5° Otherwise this is a **hit:** return x

In this case, if we would be integrating a function $h(x, y)$ over a 2-dimensional volume bound by the x axis and $f(x)$, then we would just change step 5° to return $both$ x and y, then evaluate h in the point (x, y).

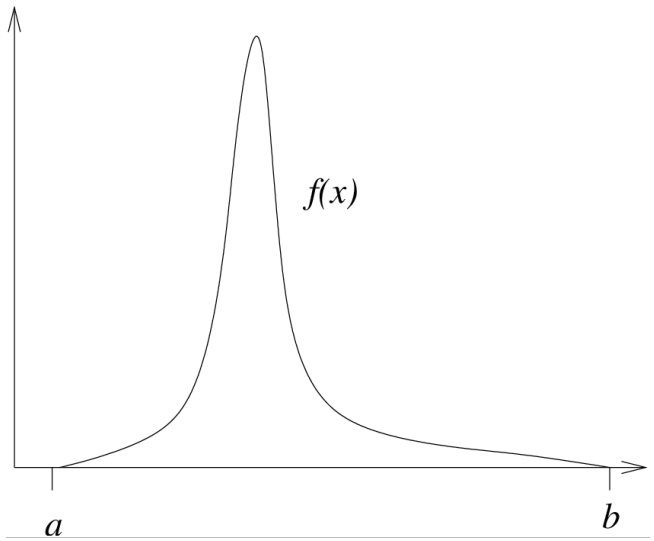
Of course, with increasing dimensions this becomes increasingly complicated.

5.4. Importance sampling

[Karimäki notes]

One approach to improving the MC accuracy is reducing the variance σ^2 in the data. Remember that σ^2 for any non-constant data distribution is a quantity which goes towards some finite, non-zero value when $N \longrightarrow \infty$, whereas the error of course goes to 0 with increasing N. Since the MC error is $\propto \frac{\sigma}{\sqrt{N}}$, it is clear that if we can reduce the variance σ^2 the error will also go down for the same N.

This is also simple to understand intuitively. Consider e.g. the following functional shape we would want to integrate:



It is quite obvious that most of the integral comes from the region of the peak. But if we generate points evenly in the interval $[a, b]$, most points won't be in the peak area, and their contribution to the total will be relatively small.

In fact, some simply thought indicates that the least effort will be spent in case the distribution is fairly flat. In that case the variance σ^2 will become smaller.

The idea behind importance sampling is to transform $f(x)$ into another, flatter function which is then MC integrated. Of course there has to be a back-transformation to give the original integral we really want.

So let's say we have a function $g(x)$ normalized over the integration interval $[a, b]$ which gives the

property that

$$\frac{f(x)}{g(x)}$$

is fairly flat. $g(x)$ has to be > 0 for all x in the interval. We now want to calculate

$$I = \int_a^b f(x)dx = \int_a^b \frac{f(x)}{g(x)} g(x)dx = \int_a^b \frac{f(x)}{g(x)} dG(x)$$

where

$$G(x) = \int_a^x g(x)dx$$

is the integral of $g(x)$. If we now make a variable change $r = G(x)$ we get

$$I = \int_{G(a)}^{G(b)} \frac{f(G^{-1}(r))}{g(G^{-1}(r))} dr$$

which gives the same integral, but more efficiently because the integrand is flatter than the original. Evaluating this with MC is done in the same way as for any other function,

$$I = \frac{1}{N} \sum_{i=1}^{N} \frac{f(G^{-1}(r_i))}{g(G^{-1}(r_i))}$$

where the r_i are uniform random numbers.

So we have to be able to determine G^{-1}. But there is an alternative: it is actually enough that we can generate random points distributed as $g(x)$ by any means (not necessarily analytically). In this case the MC sum is

$$I = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i^{(g)}))}{g(x_i^{(g)})}$$

where the $x_i^{(g)}$ **are random numbers distributed as** $g(x)$**.**

What is actually the advantage of this compared to using hit-and-miss MC integration with the combined rejection method for some function $ag(x)$? In that case, using $g(x)$ reduces the number of misses a lot, and since we integrate in one more dimension, all points have equal value (1), and hence the variance should be low as well? As far as I can see, the main advantage is that now we do not have to go up in dimensions, nor require that $ag(x) > f(x)$, which may sometimes be difficult or impossible to prove, especially in high dimensions.

Let us make this concrete with a simple example. Say we want to evaluate part of a Gaussian function,

$$I = \int_0^1 e^{-x^2} dx$$

In this region, the function decreases from 1 to $1/e$. The simple exponential function e^{-x} does the

same, so let's use that for $g(x)$. We first have to normalize g, so we calculate

$$\int_0^1 e^{-x}dx = -\frac{1}{e} + 1 = 1 - \frac{1}{e} = \frac{e-1}{e}$$

and see that our normalized weighting function is

$$g(x) = \frac{e^{-x}e}{(e-1)}$$

Then

$$G(x) = \int_0^x \frac{e^{-x}e}{e-1}dx = \frac{(1-e^{-x})e}{e-1}$$

and

$$G^{-1}(u) = -\log\left(1 - u\frac{e-1}{e}\right)$$

Does this seem complicated? Well, it is not. An entire code to do all this is given here (awk/C):

```
gawk -v N=$1 'BEGIN {
        srand();
        e=exp(1);
        sum=0.0;
        for(i=0;i<N;i++) {
                r=-log(1-rand()*(e-1)/e);
                foverg=exp(-r*r)/( exp(-r)*e/(e-1) );
                sum+=foverg;
        }
        print sum/N;
        exit;
}'
```

Is this really advantageous, then? I did a comparison of this vs. direct MC integration of the same function. The results are here:

	Direct sampling		Importance sampling	
N	result	time	result	time
10000	0.743275	0.20	0.746915	0.06
100000	0.746407	0.20	0.746801	0.52
1000000	0.746966	2.21	0.746829	4.95
10000000	0.746771	21.72	0.746830	48.77

(Note that using the scripting language awk for this is of course hideously slow compared to C or Fortran)

The correct answer is, to 6 decimals, 0.746824. So we see that although for the same N importance sampling is about two times slower, it gets closer to the correct answer for a much smaller number of iterations. At $N = 10000000$ the error in the direct method is $71/10^6$, but only $10/10^6$ in the importance sampled method. So clearly it is worth using the importance sampling.

5.4.1. Control variates

The idea here is similar to importance sampling. We want to replace the function to be integrated by something flatter to reduce the variance and thus the error of the data. But instead of division, we use subtraction.

The operation is simply

$$I = \int_a^b = \int_a^b (f(x) - g(x))dx + \int_a^b g(x)dx$$

and the idea is to find a $g(x)$ such that

- Var$(f - g) <$ Var(f)
- $\int_a^b g(x)$ dx is known.

The integral $\int_a^b (f(x) - g(x))dx$ is evaluated with ordinary MC. This approach has the following advantages compared to importance sampling:

- It does not matter if $g(x)$ is zero somewhere in the interval

- We do not need to know how to distribute random numbers according to $g(x)$

- g can also be negative

The variance of $f - g$ is

$$\mathrm{Var}(f - g) = \mathrm{Var}(f) + \mathrm{Var}(g) - 2\mathrm{Cov}(f, g)$$

where the last term is the covariance between f and g. In order that this be useful, we should thus have

$$2\mathrm{Cov}(f, g) > \mathrm{Var}(g)$$

i.e. f and g are positively correlated (have similar shapes).

5.5. Stratified sampling

Importance sampling is a fine way to improve on MC integration, but has the disadvantage that the function, or at least its overall shape, has to be known. However, often it is not. As mentioned above, f may actually be a number returned by some other, hugely complicated simulation. This could e.g. be an MD, electronic structure or another MC simulation.

In principle one could first do a set of simulations to determine the rough shape of f on a numeric grid, then use this numeric data as the weighting function g (remember that it is possible to form random numbers also for a function existing only in numeric form). This would probably work fine in 1 or only a few dimensions. But in a high number of dimensions M, we would need memory proportional to N_{points}^{M} to store the numeric function.

Stratified sampling does not have these problems. It can be used for any function, even when nothing is known about its shape, and does not require storage increasing with the dimensionality.

Generating random numbers in a stratified or quasi-random manner was already discussed in the last section. Now we see why this may be important: as we derived above, the error of the ordinary MC method decreases as

$$\frac{1}{\sqrt{N}}$$

for true and pseudo-random numbers. But as mentioned in the last section, with stratified sampling or quasi-random numbers one can achieve at best an error decreasing as

$$\frac{1}{N}$$

which may give an enormous saving in time.

5.5.0.1. Motivation of 1/N dependence

[Numerical Recipes ch. 7.8.]

I will now explain the idea of why stratified sampling is (can be) more efficient than direct Monte Carlo, although I will not derive the $1/N$ formula.

Recall that the error of MC integration normally is

$$\sigma_{V\langle f\rangle} \approx V\sqrt{\frac{\langle f^2\rangle - \langle f\rangle^2}{N}} = V\sqrt{\frac{\sigma^2}{N}}$$

where σ^2 is the variance.

Let us now use the following notation: $\langle\langle f \rangle\rangle$ is the true average of the integral in the region of interest, $\langle f \rangle$ the basic MC integration estimate of this:

$$\langle\langle f \rangle\rangle = \frac{1}{V} \int f dV \qquad \text{and} \qquad \langle f \rangle = \frac{1}{V} \sum_i f(x_i)$$

The variance of the MC integration, $\sigma^2_{\langle f \rangle} = \mathrm{Var}(\langle f \rangle)$ is related to the variance of the true function,

$$\mathrm{Var}(f) = \langle\langle f^2 \rangle\rangle - \langle\langle f \rangle\rangle^2$$

by

$$\mathrm{Var}(\langle f \rangle) = \frac{\mathrm{Var}(f)}{N}$$

when $N \to \infty$.

The point of generating random numbers for stratified sampling was to divide V into a number of boxes, then generate one or more random numbers in each box.

Let's now consider the simplest possible case of dividing V into two equally large regions a and b, both of which are sampled at $N/2$ points. Then the MC integration gives an alternative estimate $\langle f \rangle'$ of the true integral $\langle\langle f \rangle\rangle$,

$$\langle f \rangle' = \frac{1}{2} \left(\langle f \rangle_a + \langle f \rangle_b \right)$$

and the variance of this is now

$$\begin{aligned}
\mathrm{Var}(\langle f\rangle') &= \frac{1}{4}\left[\mathrm{Var}(\langle f\rangle_a) + \mathrm{Var}(\langle f\rangle_b)\right] \\
&= \frac{1}{4}\left[\frac{\mathrm{Var}_a(\langle\langle f\rangle\rangle)}{N/2} + \frac{\mathrm{Var}_b(\langle\langle f\rangle\rangle)}{N/2}\right] \\
&= \frac{1}{2N}\left[\mathrm{Var}_a(\langle\langle f\rangle\rangle) + \mathrm{Var}_b(\langle\langle f\rangle\rangle)\right]
\end{aligned}$$

On the other hand, let us calculate the variance of the true function,

$$\begin{aligned}
\mathrm{Var}(f) &= \mathrm{Var}(\langle\langle f\rangle\rangle) = \langle\langle f^2\rangle\rangle - \langle\langle f\rangle\rangle^2 \\
&= \frac{1}{V}\left(\int_a f^2 dV + \int_b f^2 dV\right) - \left(\frac{1}{V}\int_a f dV + \int_b f dV\right)^2 \\
&= \frac{1}{2}\overbrace{\frac{2}{V}\int_a f^2 dV}^{\langle\langle f^2\rangle\rangle_a} + \frac{1}{2}\overbrace{\frac{2}{V}\int_b f^2 dV}^{\langle\langle f^2\rangle\rangle_b}
\end{aligned}$$

$$
\begin{aligned}
& - \left(\frac{1}{2} \overbrace{\frac{2}{V} \int_a f dV}^{\langle\langle f \rangle\rangle_a} \right)^2 - \left(\frac{1}{2} \overbrace{\frac{2}{V} \int_b f dV}^{\langle\langle f \rangle\rangle_b} \right)^2 - \frac{1}{2} \overbrace{\frac{2}{V} \int_a f dV}^{\langle\langle f \rangle\rangle_a} \overbrace{\frac{2}{V} \int_b f dV}^{\langle\langle f \rangle\rangle_b} \\
= \quad & \frac{1}{2}\langle\langle f^2 \rangle\rangle_a + \frac{1}{2}\langle\langle f^2 \rangle\rangle_b - \frac{1}{4}\langle\langle f \rangle\rangle_a^2 - \frac{1}{4}\langle\langle f \rangle\rangle_b^2 - \frac{1}{2}\langle\langle f \rangle\rangle_a \langle\langle f \rangle\rangle_b \\
= \quad & \underbrace{\frac{1}{2}\langle\langle f^2 \rangle\rangle_a + \frac{1}{2}\langle\langle f^2 \rangle\rangle_b - \frac{1}{2}\langle\langle f \rangle\rangle_a^2 - \frac{1}{2}\langle\langle f \rangle\rangle_b^2}_{} \underbrace{+\frac{1}{4}\langle\langle f \rangle\rangle_a^2 + \frac{1}{4}\langle\langle f \rangle\rangle_b^2 - \frac{1}{2}\langle\langle f \rangle\rangle_a \langle\langle f \rangle\rangle_b}_{} \\
= \quad & \overbrace{\frac{1}{2}\langle\langle f^2 \rangle\rangle_a - \frac{1}{2}\langle\langle f \rangle\rangle_a^2}^{1/2 \mathsf{Var}_a(\langle\langle f \rangle\rangle)} \overbrace{+\frac{1}{2}\langle\langle f^2 \rangle\rangle_b - \frac{1}{2}\langle\langle f \rangle\rangle_b^2}^{1/2 \mathsf{Var}_b(\langle\langle f \rangle\rangle)} + \frac{1}{4}\left(\langle\langle f \rangle\rangle_a - \langle\langle f \rangle\rangle_b\right)^2 \\
= \quad & \frac{1}{2}\mathsf{Var}_a(\langle\langle f \rangle\rangle) + \frac{1}{2}\mathsf{Var}_b(\langle\langle f \rangle\rangle) + \frac{1}{4}\left(\langle\langle f \rangle\rangle_a - \langle\langle f \rangle\rangle_b\right)^2
\end{aligned}
$$

Comparison with the previous equations gives

$$
\begin{aligned}
\mathsf{Var}(\langle f \rangle') &= \frac{1}{N}\left(\mathsf{Var}(f) - \frac{1}{4}\left(\langle\langle f \rangle\rangle_a - \langle\langle f \rangle\rangle_b\right)^2 \right) \\
&= \mathsf{Var}(\langle f \rangle) - \frac{1}{4N}\left(\langle\langle f \rangle\rangle_a - \langle\langle f \rangle\rangle_b\right)^2
\end{aligned}
$$

Since the square has to be ≥ 0, we see that the variance (and hence accuracy of the MC simulation) in the two intervals is at most the same as that for the single interval. And if there is large variation between $\langle\langle f \rangle\rangle_a$ and $\langle\langle f \rangle\rangle_b$, it can be considerably less!

A similar calculation for larger numbers of intervals will give a similar result: the stratified sampling can considerably reduce the variance and error for the same number of samples N. And one does not even have to require that each subregion has the same number of points. One can show that the optimal allocation is achieved when the number of points in each subinterval i is proportional to σ_i in that interval.

At best this can lead to a $1/N$ dependence.

Unfortunately there is no guarantee one can always achieve the $1/N$ dependence – it is very much dependent on the application. In practice, for a new kind of MC integration it is probably best simply to test whether an advantage over $\frac{1}{\sqrt{N}}$ is achievable by some stratified sampling or quasi-random number method.

5.6. Combined and advanced methods

[Numerical recipes]

It is perfectly possible, and sometimes quite useful, to combine the importance and stratified sampling methods. Remember that importance sampling essentially focuses the effort on the regions which contribute most to the integral, while stratified sampling improves on the convergence. Thus one could e.g. do importance sampling first to flatten the distribution, then stratified sampling to reduce the variance and further improve on the convergence.

Moreover, it is possible to make parts of the routine adaptive. For instance, in the VEGAS routine used widely in elementary particle physics, a separable weight function is constructed for M dimensions

$$g(x_1, x_2, ..., x_M) = g_{x_1}(x_1) g_{x_2}(x_2) \cdots g_{x_M}(x_M)$$

The idea of the separation is to avoid the need for N_{points}^M storage space. Each g_{x_j} can be stored as a 1-dimensional numerical array with N_{points} elements, reducing storage needs to $N_{\mathrm{points}} M$

Moreover, each subfunction g_{x_j} can be shown to ideally have the shape

$$g_{x_1}(x_1) \propto \sqrt{\int dx_1 \int dx_2 \cdots \int dx_M \ldots \frac{f^2(x_1, x_2, ..., x_M)}{g_{x_2}(x_2) \cdots g_{x_M}(x_M)}}$$

and correspondingly for the other x_j. The idea is that while sampling f, also sample f^2 and hence calculate a new g_{x_j} on every iteration step. Thus we have an adaptive importance sampling scheme! Unless the function to be sampled has certain special shapes, this does improve on convergence a lot.

In addition, the VEGAS routines also sometimes does stratified sampling. But we will not discuss more here. If interested, you can read about this, and another advanced MC scheme in Numerical Recipes ch. 7.8.