1 Monte Carlo quadratures

Monte Carlo integration is a numerical quadrature where the abscissas are chosen randomly and no assumptions about smoothness of the integrand are made, not even that the integrand is continuous.

Plain Monte Carlo algorithm distributes points (in a process called "sampling") uniformly from the integration region using either uncorrelated pseudo-random or correlated quasi-random sequences of points.

Adaptive algorithms, such as VEGAS and MISER, distribute points non-uniformly, attempting to reduce integration error, using "importance" and "stratified" sampling, correspondingly.

1.1 Multi-dimensional integration

One of the problems in multi-dimensional integration is that often the integration region Ω is quite complicated, with the boundary not easily described by simple functions. However it is usually a lot easier to find out whether a given points lies within the integration region or not. Therefore a popular strategy is to create an auxiliary rectangular volume V which contains the integration volume Ω an an auxiliary function F which coincides with the integrand inside the volume Ω and is equal zero outside. Then the integral of the auxiliary function over the (simple rectangular) auxiliary volume is equal the original integral.

Unfortunately the auxiliary function is non-continuous at the boundary and thus the ordinary quadratures which assume continuous integrand will fail badly here while the Monte Carlo quadratures will do just as good (or bad) as with continuous integrand.

1.2 Plain Monte Carlo sampling

Plain Monte Carlo is a quadrature with equal weights and non-optimised random abscissas,

$$\int_V f(\mathbf{x})dV = w\sum_{i=1}^{N} f(\mathbf{x}_i)\ , \tag{1}$$

where $\mathbf{x}$ a point in the multi-dimensional integration space. One free parameter, w, allows one condition to be satisfied, that basically is the overall normalisation, or, in other words the quadrature has to integrate exactly a constant function. This gives $w = V/N$,

$$\int_V f(\mathbf{x})dV = \frac{V}{N}\sum_{i=1}^{N} f(\mathbf{x}_i) = V\langle f\rangle\ . \tag{2}$$

According to the *central limit theorem* the error estimate ϵ is close to

$$\epsilon = V\frac{\sigma}{\sqrt{N}}\ , \tag{3}$$

where σ is the variance of the sample,

$$\sigma^2 = \langle f^2\rangle - \langle f\rangle^2. \tag{4}$$

The $1/\sqrt{N}$ convergence of the error is quite slow (albeit independent upon the smoothness of the integrand).

1.3 Importance sampling

Suppose that the points are distributed not uniformly but with some density ρ. The number of points Δn in the volume ΔV is thus equal $\Delta n = \rho\Delta V$. Then the estimate of the integral is given as

$$\int f(\mathbf{x})dV \approx \sum_{i=1}^{N} f(x_i)\Delta V_i = \left\langle \frac{f}{\rho}\right\rangle\ . \tag{5}$$

The corresponding variance is now

$$\sigma^2 = \left\langle \left(\frac{f}{\rho}\right)^2\right\rangle - \left\langle \frac{f}{\rho}\right\rangle^2\ , \tag{6}$$

Apparently if the ration f/ρ is close to a constant, the variance is reduced. It is tempting to take $\rho = |f|$ and sample directly from the function. However in practice it is typically expensive to evaluate the integrand. Therefore a better strategy is to build an approximate density in the product form, $\rho(x, y, \ldots, z) = \rho_x(x)\rho_y(y)\ldots\rho_z(z)$, and then sample from this approximate density. A popular routine of this sort is called VEGAS. The sampling from a given function can be done using the Metropolis algorithm which we shall not discuss here.

1.4 Stratified sampling

Another popular approach is actually very close to our recursive adaptive integration routine (only with random quadrature and in multi-dimensional space): distribute more points where the error estimate is largest. However the "dividing by two" strategy does not work for multi-dimensions as the number of sub-volumes grows way too fast to keep track of. Instead one estimates in which dimension a subdivision should bring the most dividends and only subdivides along this dimension. This is called *stratified sampling* and here is the algorithm,

```
function strata(f, a[], b[], n, acc) :
```

```
estimate the average and the variance
using n-points plain Monte Carlo;

if the error is less than acc:

    return the average and the
    variance;

else:

    for each dimension :

        subdivide the volume in two
        along the dimension;

        estimate the sub-variances in
        the two sub-volumes;

    pick the dimension with the
    largest sub-variance;

    subdivide your volume in two along
    this dimension;

    dispatch two recursive calls to
    each of the sub-volumes;

    estimate the grand average and
    grand variance;

    return the grand average and grand
    variance;
```

1.5 Quasi-random (low-discrepancy) sampling

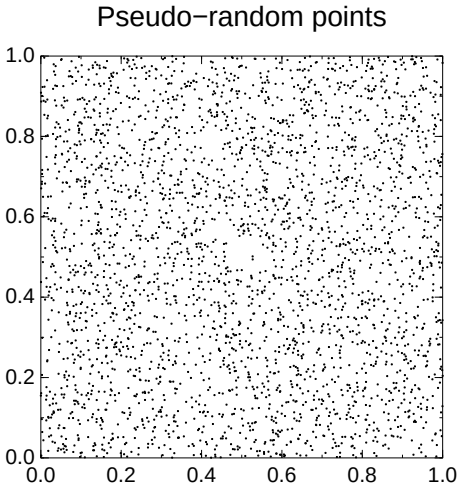

Figure 1: A typical distribution of pseudo-random points in two dimensions.

Pseudo-random sampling has high discrepancy[1] – it typically creates regions with hight density of points and other regions with very low density, see Figure 1. In other words with pseudo-random sampling there is a probability that all the n points would fall into one and the same half of the region and none into the other half.

Quasi-random sequences avoid this phenomenon by distributing points in a highly correlated manner with a specific requirement of low discrepancy, see Figure 2. Quasi-random sampling is like a computation on a grid where the grid constant must not be known in advance as the grid is ever gradually refined and the points are always distributed uniformly over the region. The computation can be stopped at any time.

The central limit theorem does not work in this case as the points are not statistically independent. Thus the variance can not be used as an estimate of the error.

1.5.1 Lattice sampling

Let α_i, $i = 1, \ldots, d$, (d is the dimension) be a set of cleverly chosen irrational numbers, like square roots of prime numbers. Then the kth point (in the unit volume) of the sampling sequence will be given as

$$\mathbf{x}^{(k)} = \{\mathrm{frac}(k\alpha_1), \ldots, \mathrm{frac}(k\alpha_d)\} \,, \qquad (7)$$

where $\mathrm{frac}(x)$ is the fractional part of x.

A problem with this method is that a high accuracy arithmetics (e.g. `long double`) might be needed in order to generate a reasonable amount of quasi-random numbers.

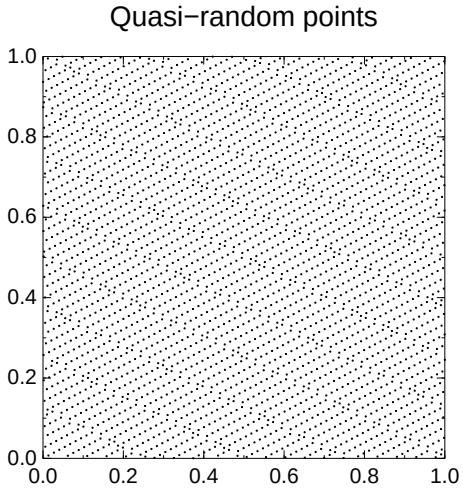

Figure 2: A typical distribution of quasi-random points in two dimensions.

[1] Discrepancy is a measure of how unevenly the points are distributed over the region.