0.1 Quasi-random (low-discrepancy) sampling

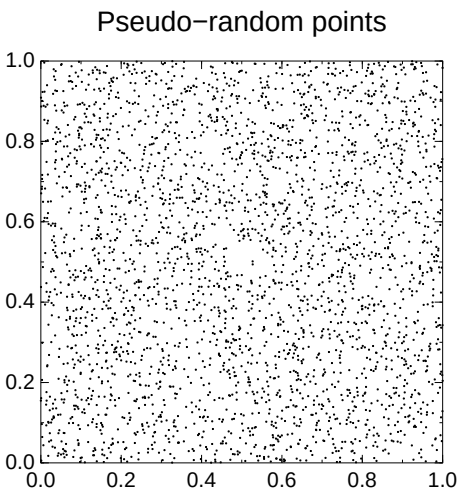

Figure 1: A typical distribution of pseudo-random points in two dimensions.

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.

0.1.1 Lattice sampling

Let α_i, $i = 1, \ldots, d$, (d is the dimension of the integration space) 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

$$\mathbf{x}^{(k)} = \{\mathrm{frac}(k\alpha_1), \ldots, \mathrm{frac}(k\alpha_d)\} \,, \qquad (1)$$

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

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

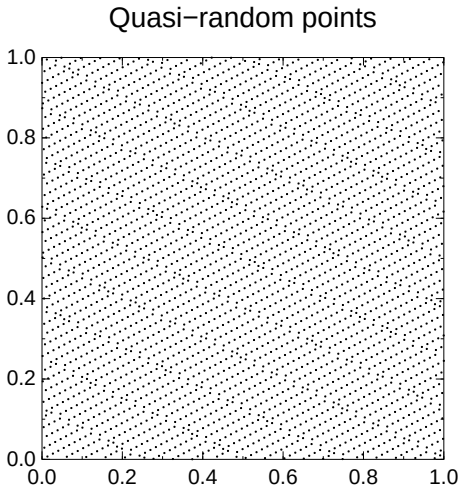

Figure 2: A typical distribution of quasi-random points in two dimensions.

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.

1 Fast Fourier transform

A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT).

For a set of complex numbers x_n, $n = 0, \ldots, N-1$, the DFT is defined as a set of complex numbers c_k,

$$c_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i \frac{nk}{N}} \,, k = 0, \ldots, N-1 \,. \qquad (2)$$

The inverse DFT is given by

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} c_k e^{+2\pi i \frac{nk}{N}} \,. \qquad (3)$$

These transformations can be viewed as expansion of the vector x_n in terms of the orthogonal basis of vectors $e^{2\pi i \frac{kn}{N}}$,

$$\sum_{n=0}^{N-1} \left(e^{2\pi i \frac{kn}{N}}\right) \left(e^{-2\pi i \frac{k'n}{N}}\right) = N\delta_{kk'} \qquad (4)$$

The DFT represent the amplitude and phase of the different sinusoidal components in the input data x_n.

The DFT is widely used in different fields, like spectral analysis, data compression, solution of partial differential equations and others.

1.1 Cooley-Tukey algorithm

In its simplest incarnation this algorithm re-expresses the DFT of size $N = 2M$ in terms of two DFTs of size M,

$$c_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i \frac{nk}{N}}$$

$$= \sum_{m=0}^{M-1} x_{2m} e^{-2\pi i \frac{mk}{M}} + e^{-2\pi i \frac{k}{N}} \sum_{m=0}^{M-1} x_{2m+1} e^{-2\pi i \frac{mk}{M}}$$

$$= \begin{cases} c_k^{(even)} + e^{-2\pi i \frac{k}{N}} c_k^{(odd)} & , k < M \\ c_{k-M}^{(even)} - e^{-2\pi i \frac{k-M}{N}} c_{k-M}^{(odd)} & , k \geq M \end{cases} , \quad (5)$$

where $c^{(even)}$ and $c^{(odd)}$ are the DFTs of the even- and odd-numbered sub-sets of x.

This re-expression of a size-N DFT as two size-$N/2$ DFTs is sometimes called the Danielson-Lanczos lemma. The exponents $e^{-2\pi i \frac{k}{N}}$ are called "twiddle factors".

The operation count by application of the lemma is reduced from the original N^2 down to $2(N/2)^2 + N/2 = N^2/2 + N/2 < N^2$.

For $N = 2^p$ Danielson-Lanczos lemma can be applied recursively until the data sets are reduced to one datum each. The number of operations is then reduced to $O(N \ln N)$ compared to the original $O(N^2)$. The established library FFT routines, like FFTW and GSL, further reduce the operation count (by a constant factor) using advanced programming techniques like precomputing the twiddle factors, effective memory management and others.

1.2 Multidimensional DFT

For example, a two-dimensional set of data $x_{n_1 n_2}$, $n_1 = 1 \ldots N_1$, $n_2 = 1 \ldots N_2$ has the discrete Fourier transform

$$c_{k_1 k_2} = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1 n_2} e^{-2\pi i \frac{n_1 k_1}{N_1}} e^{-2\pi i \frac{n_2 k_2}{N_2}} . \quad (6)$$