Journal scientific and applied research, vol. 10, 2016
Association Scientific and Applied Research
International Journal

Original Contribution ISSN 1314-6289

COMPUTING HIGH DIMENSIONAL INTEGRALS WITH MONTE CARLO METHODS

Venelin Todorov

IICT, BULGARIAN ACADEMY OF SCIENCES
E-MAIL: venelinltodorov@gmail.bg

Abstract. *High dimensional integrals are usually solved with Monte Carlo algorithms and quasi Monte Carlo algorithms. We are doing numerical testing which compare low discrepancy and Monte Carlo algorithms. It is well known that Sobol algorithm has some advantageous over the other low discrepancy sequences, that's why we use this algorithm for our numerical example. The obtained relative error confirms this superiority of the presented Monte Carlo and quasi Monte Carlo algorithms even when small number of sample points are used. It is very interesting that the presented high dimensional integral gives very low relative error even for computational time less than one second which shows the great importance of the developed algorithms.*

Keywords. *Monte Carlo and quasi Monte Carlo algorithms, multidimensional integrals, quasi Monte Carlo algorithm based on Sobol sequence, plain Monte Carlo algorithm*

Introduction

The Monte Carlo method is a widely used tool in many fields of science. It is well known that Monte Carlo methods give statistical estimates for the functional of the solution by performing random sampling of a certain random variable whose mathematical expectation is the desired functional.

Monte Carlo methods are methods of approximation of the solution to problems of computational mathematics, by using random processes for each such problem, with the parameters of the process equal to the solution of the problem. The method can guarantee that the error of Monte Carlo approximation is smaller than a given value with a certain probability [1].

An important advantage of the Monte Carlo methods is that they are suitable for solving multi-dimensional problems, since the computational complexity increases linearly and not exponentially with the dimensionality. An important advantage of the method is that it allows to compute directly functionals of the solution with the same complexity as to determine the solution.

In such a way this class of methods can be considered as a good candidate for treating innovative problems related to modern areas in quantum physics and finance.

In the last few years new approaches have been developed that outperform standard Monte Carlo in terms of numerical efficiency. It has been found that there can be efficiency gains in using deterministic sequences rather than the random sequences which are a feature of standard Monte Carlo. These deterministic sequences are carefully selected so that they are well dispersed throughout the region of integration. Sequences with this property are known as low discrepancy sequences. These sequences are often more efficient than standard Monte Carlo in evaluating high dimensional integrals if the integrand is sufficiently regular and for many finance applications this is the case. Applications of low discrepancy sequences to finance problems have been discussed by Boyle, Broadie and Glasserman (1997), Cashisch, Morokoff and Owen (1997), Joy, Boyle and Tan (1996), Ninomiya and Tezuka (1996), Tan and Boyle (2000) and Paskov and Traub (1995). The Monte Carlo method has proven to a very useful tool for nu- merical analysis, particularly when the number of dimensions ranging from medium to large. Such problems occur in a broad range of applications in science, physics and engineering. In recent years the Monte Carlo method has also become a popular computational device for problems in finance.

Multidimensional numerical quadratures are of great importance in many practical areas, ranging from atomic physics to finance [2],[3]. Monte Carlo simulation and quasi- Monte Carlo methods are the prevailing methods used to solve multi dimensional problems in different areas. Both methods do not suffer from the dimensional effect. The Monte Carlo method is known to be only accurate with a tremendous amount of scenarios since its rate of convergence is $O(N^{-1/2})$. Quasi Monte Carlo methods use deterministic sequences that have better uniform properties measured by discrepancy. They are usually superior to the Monte Carlo method as they have a convergence rate of $((\log N)^d/N)$, where N is the number of samples and d is the dimensionality of the problem under consideration.

Monte Carlo algorithms for numerical integration

Consider the problem of approximate integration of the multiple integral:

$$\int_{[0,1)^d} f(\mathbf{x})d\mathbf{x} = \int_0^1 d\mathbf{x}^{(1)} \int_0^1 d\mathbf{x}^{(2)} \dots \int_0^1 d\mathbf{x}^{(d)} f(x^{(1)}, x^{(2)}, \dots, x^{(d)}),$$

where $x = (x^{(1)}, \dots, x^{(d)}) \in [0,1)^d$.

For small values of d, numerical integration methods such as Simpson's rule or

the trapezoidal rule can be used to approximate the integral. These methods, however, suffer from the so-called curse of dimensionality and become impractical as d increases.

The crude Monte Carlo method has rate of convergence $O(N^{-1/2})$ which is independent of the dimension of the integral, and that is why Monte Carlo integration is the only practical method for many high-dimensional problems. Much of the efforts to improve Monte Carlo are in construction of variance reduction methods which speed up the computation or to use quasi-random sequences [1]. A quasi-random or low discrepancy sequence, such as the Faure, Halton, Hammersley, Niederreiter or Sobol sequences, is ”less random” than a pseudorandom number sequence, but more useful for such tasks as approximation of integrals in higher dimensions, and in global optimization. This is because low discrepancy sequences tend to sample space ”more uniformly” than random numbers. We modify the algorithm for Sobol sequence that is an adapation of the INSOBL and GOSOBL routines in ACM TOMS Algorithm 647 and ACM TOMS Algorithm. The original code can only compute the ”next” element of the sequence. The revised code allows the user to specify the index of the desired element. The algorithm has a maximum spatial dimension of 40 since MATLAB doesn't support 64 bit integers. A remark by Joe and Kuo shows how to extend the algorithm from the original maximum spatial dimension of 40 up to a maximum spatial dimension of 1111. The FORTRAN90 and C++ versions of the code has been updated in this way, but updating the MATLAB code has not been simple, since MATLAB doesn't support 64 bit integers. We use algorithm that generates a new quasirandom Sobol vector with each call. The routine adapts the ideas of Antonov and Saleev . The parameters of the algorithm are an integer $DIMNUM$, the number of spatial dimensions. The algorithm starts with integer $SEED$, the ”seed” for the sequence. This is essentially the index in the sequence of the quasi-random value to be generated. On output, $SEED$ has been set to the appropriate next value, usually simply $SEED + 1$. If $SEED$ is less than 0 on input, it is treated as though it were 0. An input value of 0 requests the first (0-th) element of the sequence. Output is the real $QUASI(DIMNUM)$, the next quasi-random vector [5].

Consider an example with $d = 40$. In order to apply such formulae, we generate a grid in the d-dimensional domain and take the sum (with the respective coefficients according to the chosen formula) of the function values at the grid points. Let a grid be chosen with 10 nodes on the each of the coordinate axes in the d-dimensional cube $G = [0, 1]^d$. In this case we have to compute about 10^{40} values of the function f(x). Suppose a time of $10^{-7}s$ is necessary for calculating one value of the function.

Therefore, a time of order 10^{33}s will be necessary for evaluating the integral (remember that 1 year = 31536×10^3s, and that there has been less

than 9×10^{10}s since the birth of Pythagoras). So to compute 40 dimensional integral with deterministic formula we need $10^{33}/31556926 = 3 \times 10^{25}$ years. Consider a plain Monte Carlo algorithm for this problem with a probable error of the same order. The algorithm itself consists of generating N pseudo random values (points) (PRV) in G; in calculating the values of f(x) at these points; and averaging the computed values of the function. For each uniformly distributed random point in G we have to generate 40 random numbers uniformly distributed in [0, 1]. The probable error is:

$$\epsilon \leq 0.6745||f||_{L_2}\frac{1}{\sqrt{N}}.$$

From above equations and using that $\varrho \leq cMh^3$ we obtain

$$N \approx \left(\frac{0.6745||f||_{L_2}}{cM}\right)^2 \times h^{-6}.$$

Suppose that the expression in front of h^{-6} is of order 1. For our example $h = 0{,}1$, we have $N \approx 10^6$; hence, it will be necessary to generate $40 \times 10^6 = 4 \times 10^7$ PRV. Usually, two operations are sufficient to generate a single PRV. Suppose that the time required to generate one PRV is the same as that for calculating the value of the function at one point in the domain. Therefore, in order to solve the problem with the same accuracy, a time of $4 \times 10^7 \times 2 \times 10^7 \approx 8s$ will be necessary. The advantage of the Monte Carlo algorithms to solve such problems is obvious.

Numerical example

We will test the two methods for evaluating the following 40 dimensional integral:

$$\int_{[0,1]^{40}} exp(\sum_{i=1}^{40} 0.1x_i) \approx 7.51322858.$$

For the crude Monte Carlo algorithm we use the Mersenne Twister number generator. The algorithm used by this method, developed by Nishimura and Matsumoto, generates double precision values in the closed interval $[2^{-53}, 1-2^{-53}]$, with a period of $(2^{19937}-1)/2$. As can been seen from the table for 40 dimensional integral, the low discrepancy sequence produces more rapid convergence, and lower errors, than the pseudorandom sequence. This is as anticipated since the pseudo randomly obtained averages converge at the rate $O(N-1/2)$, while the quasi randomly obtained averages converge at a rate closer to $O(N^{-1})$. For larger number of points, the advantage of using either pseudorandom or quasi-random methods in place of more conventional quadrature formulas should become even more pronounced.

Table 1: The relative error for 40 dimensional integral

N	Sobol	time	Crude	time
1000	1.20e-3	0.07	4.64e-2	0.001
10000	1.33e-3	0.69	1.20e-2	0.02
100000	6.33e-5	6.60	2.05e-3	0.19
1000000	5.46e-6	72.7	1.75e-3	1.47
10000000	1.17e-7	782.3	8.25e-4	14.81

Table 2: Computational complexity for 40 dimensional integral

time in seconds	Crude	Sobol
1	1.52e-3	8.86e-4
5	1.16e-3	7.65e-5
10	3.66e-4	5.03e-5
20	3.75e-5	8.75e-6
60	2.70e-5	3.41e-6

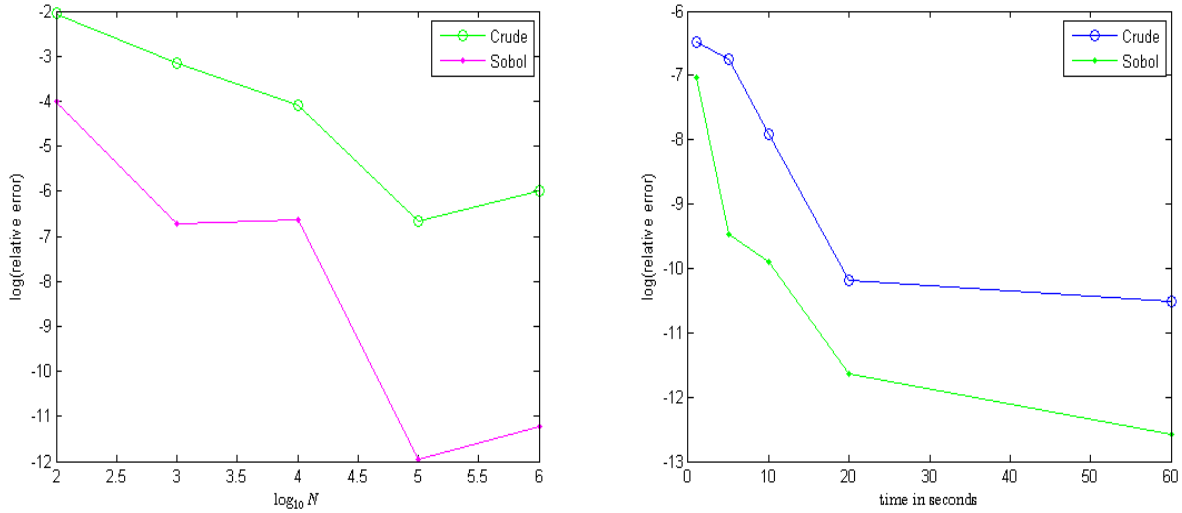

Fig. 1. Relative error and computational time for 40 dimensional integral with Monte Carlo and quasi Monte Carlo methods

It is very interesting that an error of $1.17e-7$ can be achieved for 40 dimensional integral. If someone need an error of 1% this can be done in less than a second which is impressive results and it is of great importance for applied scientists. It can be seen that only for 1s the two methods under consideration gives relative error of $1.52e-3$ and $8.86e-4$ respectively which is already a sufficient accuracy. For 60s the results are even more precise.

Conclusion

In this paper we analyze the performance of quasi Monte Carlo and crude Monte Carlo algorithm for multidimensional integrals. The Sobol quasi-random sequence is compared with the pseudorandom sequence and the results are very precise even for 40 dimensional integral, which shows the strength of the presented algorithm for higher dimension $d \leq 40$. This multidimensional integral can be applied to various problems where data is taken in randomized way [4]. They are often used in physical problems and are most useful when it is difficult or impossible to use other mathematical methods. These improved methods are the only possible algorithms for high dimensional integrals, because we see that the deterministic algorithms need an impossible amount of time and become impractical.

Acknowledgement

This work was supported by Program for career development of young scientists, BAS, Grant No. DFNP-91/04.05.2016 as well as the Bulgarian National Science Fund under Grant DFNI I02-20/2014.

References

[1]. Dimov I., Monte Carlo Methods for Applied Scientists, New Jersey, London, Singapore, World Scientific, 2008, 291p.

[2]. Dzhurov V. APPLICATION OF SUMMARIZED FUNCTIONS FOR INFORMATION SOURCE PROTECTION. Journal Scientific and Applied Research, 2013, No 3, pp. 51-56.

[3]. Dzhurov V., Kostova M., Dzhurov K. Application of Probability Neural Networks for Classification of Explosives with Blasting Action. Mathematics in Industry, Network Applications in Industry, Cambridge Scholars Publishing, 2014, pp. 254-285.

[4]. Slavova A., Kostova M., Dzhurov V., Tsakoumis A., Mladenov V.. Processing of Radioholographic Image with CNN Neural Network.// WSEAS Transactions on Signal Processing, 2005, No 1, pp. 67-72, ISSN 1790-5022.

[5]. Sobol I., Asotsky D., Kreinin A., Kucherenko S.. Construction and Comparison of High-Dimensional Sobol' Generators, 2011, Wilmott Journal, Nov, pp. 64-79