Efficient Stochastic Approaches for Multidimensional Integrals in Bayesian Statistics

Chapter · February 2020

DOI: 10.1007/978-3-030-41032-2_52

2 authors:

Venelin Todorov
Bulgarian Academy of Sciences
26 PUBLICATIONS **31** CITATIONS
SEE PROFILE

Ivan Dimov
Bulgarian Academy of Sciences
251 PUBLICATIONS **1,806** CITATIONS
SEE PROFILE

Efficient Stochastic Approaches for Multidimensional Integrals in Bayesian Statistics

Venelin Todorov[1,2], Ivan Dimov[2]

[1]Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Department of Information Modeling, Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
[2]Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Department of Parallel Algorithms, Acad. G. Bonchev Str., Block 25 A, 1113 Sofia, Bulgaria
venelin@parallel.bas.bg,vtodorov@math.bas.bg,
ivdimov@bas.bg

Abstract. A fundamental problem in Bayesian statistics is the accurate evaluation of multidimensional integrals. A comprehensive experimental study of quasi-Monte Carlo algorithms based on Sobol sequence combined with Matousek linear scrambling and a comparison with adaptive Monte Carlo approach and a lattice rule based on generalized Fibonacci numbers has been presented. The numerical tests show that the stochastic algorithms under consideration are efficient for multidimensional integration and especially for computing high dimensional integrals. It is a crucial element since this may be important to be estimated in order to achieve a more accurate and reliable interpretation of the results in Bayesian statistics which is foundational in applications such as machine learning.
Keywords. Monte Carlo and quasi-Monte Carlo algorithms, Multidimensional integrals, Bayesian statistics, Machine learning

1 Introduction

Stochastic techniques have been developed over many years in a range of different fields, but have only recently been applied to the problems in Bayesian statistics [12]. As well as providing a consistent framework for statistical pattern recognition, the stochastic approach offers a number of practical advantages including a solution to the problem for higher dimensions [11]. Multidimensional integrals arise in algebraic analysis for nonidentifiable learning machines [19]. The accurate evaluation of marginal likelihood integrals is a difficult fundamental problem in Bayesian inference that has important applications in machine learning [17]. Conventional approaches to network training are based on the minimization of an error function, and are often motivated by some underlying principle such as maximum likelihood [3].

The Monte Carlo method is known to be only accurate with a tremendous amount of scenarios since its rate of convergence is $O(1/\sqrt{N})$ [5]. 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 [4]. 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. 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 $((\log N)^s/N)$, where N is the number of samples and s is the dimensionality of the problem.

2 Formulation of the problem

A fundamental problem in neural networks is the accurate evaluation of multidimensional integrals. We will primarily be interested in two kinds of integrals. The first has the form

$$\int_{\Omega} p_1^{u_1}(x) \dots p_s^{u_s}(x) dx, \tag{1}$$

where $\Omega \subseteq \mathcal{R}^s$, $x = (x_1, \dots, x_s)$, $p_i(x)$ are polynomials and u_i are integers. The second kind of integrals has the form

$$\int_{\Omega} e^{-Nf(x)} \phi(x) dx, \tag{2}$$

where $f(x)$ and $\phi(x)$ are s-dimensional polynomials and N is a natural number. These integrals are investigated by Shaowei Lin in [11,12]. The asymptotics of such integrals is well understood for regular statistical models, but little was known for singular models until a breakthrough in 2001 due to Sumio Watanabe [19]. His insight was to put the model in a suitable standard form by employing the technique of resolution of singularities from algebraic geometry. To apply the Monte Carlo method we will map the domain Ω into the s dimensional hypercube $[0,1]^s$. There is always such a transformation, for example $2/\pi \arctan(x)$ maps $(0,\infty)$ to $(0,1)$.

Consider the problem of approximate integration of the multiple integral

$$\int_{[0,1)^s} f(\mathbf{x}) d\mathbf{x} = \int_0^1 d\mathbf{x^{(1)}} \int_0^1 d\mathbf{x^{(2)}} \dots \int_0^1 d\mathbf{x^{(s)}} f(x^{(1)}, x^{(2)}, \dots, , x^{(s)}) = \theta \tag{3}$$

where $x = (x^{(1)}, \dots, x^{(s)}) \in [0,1)^s$ and $|\theta| < 1$. Since the variance is finite, the function f must be square-integrable in $[0,1)^s$.

For small values of s, 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 s increases beyond 3 or 4. One viable technique for larger values of s is to use sampling methods such that the estimator of θ becomes $\hat{\theta} = \frac{1}{N} \sum_{n=0}^{N-1} f(x_n)$ where $P_N = \{x_0, x_1, \dots, x_{N-1}\} \in [0,1)^s$ is some point set. Different techniques are available for selecting these point sets.

When the integration nodes P_N are N independently and identically distributed random points in $[0,1)^s$, the above sampling method becomes the standard Monte Carlo integration method and the resulting estimator $\hat{\theta}^{MC} \equiv \theta^{MC}$ is known as as the Monte Carlo estimator. Important properties of $\hat{\theta}^{MC}$ are as follows:

- It is an unbiased estimator of θ with variance σ^2/N; i.e.

$$\mathbb{E}[\hat{\theta}^{MC}] = \theta \text{ and } Var[\hat{\theta}^{MC}] = \sigma^2/N$$

- The Strong Law of Large Numbers asserts that $\hat{\theta}^{MC}$ converges to θ almost surely.
- The Central Limit Theorem guarantees that the distribution of $\hat{\theta}$ converges asymptotically to a normal distribution with mean θ and variance σ^2/N as $N \to \infty$. In other words, the error $|\theta - \hat{\theta}^{MC}|$ converges probabilistically at a rate of $O(N^{-1/2})$.
- The rate of convergence $O(N^{-1/2})$ is independent of s.

3 Description of the stochastic methods

3.1 Scrambled Sobol Sequence

Much of the efforts to improve Monte Carlo methods are in construction of variance reduction methods which speed up the computation or to use quasirandom sequences. A quasirandom 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. The Sobol sequence [1] is the most widely deployed low-discrepancy sequence, and is used for calculating multi-dimensional integrals. For example, Paskov uses a quasi-Monte Carlo sequence - the Sobol sequence - to find the present value of securities which involves up to 360-dimensional integrals [15]. Owen first proposed the idea of scrambling this sequence in a manner that maintained its low discrepancy [14]. One of his motivations was to use these scrambled sequences to provide quasi-Monte Carlo methods with simple error estimates like those in normal Monte Carlo. In fact, it is now common for the Sobol sequence as well as (t, m, s)-nets and (t, s)-sequences to be used with scrambling. However, many have pointed out that scrambling is often difficult to implement and time consuming [2]. Implementing scrambled digital sequences is discussed in [9]. We use a random linear scramble combined with a random digital shift, described by J. Matousek in [13]. Generally this scrambling improves the results produced by the Sobol sequence.

3.2 Adaptive algorithm

The adaptive algorithm [7] has higher accuracy and faster convergence than the plain Monte Carlo integration as can be seen from the tables below. The only drawback is the higher computational time. The algorithm is described below [6].

Algorithm

1. **Input data**: *number of points N, constant ε (estimation for the variance), constant δ (stop criterion; estimation for the length of subintervals on every coordinate).*
2. **For** $j = 1,\ M^s$:
   - **2.1. Calculate** *the approximation of I_{Ω_j} and the variance D_{Ω_j} in subdomain Ω_j based on N independent realizations of random variable θ_N*;
   - **2.2. If** $(\mathrm{D}_{\Omega_j} \geq \varepsilon)$ **then**

2.2.1. Choose *the axis direction on which the partition will perform*,

2.2.2. Divide *the current domain into two* (G_{j_1}, G_{j_2}) *along the chosen direction*,

2.2.3. If *the length of obtained subinterval is less than* δ **then go to** *step 2.2.1* **else** $j = j_1$ $(G_{j_1}\ is\ the\ current\ domain)$ *and* **go to** *step 2.1*;

2.3. Else if $(\mathrm{D}_{\Omega_j} < \varepsilon)$ *but an approximation of* $I_{G_{j_2}}$ *has not been calculated yet*, **then** $j = j_2$ $(G_{j_2}\ is\ the\ current\ domain\ along\ the\ corresponding\ direction)$ *and* **go to** *step 2.1*;

2.4. Else if $(\mathrm{D}_{\Omega_j} < \varepsilon)$ *but there are subdomains along the other axis directions*, **then go to** *step 2.1*;

2.5. Else *Accumulation in the approximation* I_N *of* I.

3.3 Lattice rules

Let n be an integer, $n \geq 2$ and $a = (a_1, a_2, \ldots a_s)$ be an integer vector modulo n. A set of the form [18]

$$P_n = \left\{ \left\{ \frac{ak}{n} \right\} = \left(\left\{ \frac{a_1 k}{n} \right\}, \ldots, \left\{ \frac{a_s k}{n} \right\} \right) \mid k = 1, \ldots, n \right\} \tag{4}$$

is called a lattice point set, where $\{x\}$ denotes the fractional part of x. The vector a is called a lattice point or generator of the set. This article restricts itself to sets of the form (4), which are also called node sets of rank-1 lattices [16]. The earlier development of these lattices is covered in detail in the monograph of Hua and Wang (1981) [8]. As one can see, the formula for the lattice point set is simple to program. The difficulty lies in finding a good value of a, such that the points in the set are evenly spread over the unit cube. The choice of good generating vector, which leads to small errors, is not trivial. Complicated methods from theory of numbers are widely used, for example Zaremba's index or error of the worst function. Korabov [10] consider the following vectors: $g = (1, a, a^2, \ldots, a^{s-1})\ mod\ N, 1 \leq a \leq N-1, gcd(a, N) = 1$.

This method can be applied only for number of points $n_l = F_l^{(s)}$, i.e., only for generalized Fibonacci number of points. This set used the generating vector [18] $a = (1, F_{l+1}^{(s)}, ..., F_{l+s-1}^{(s)}))$, $n_l = F_l^{(s)}$, where $F^{(s)}$ is the corresponding generalized Fibonacci number of dimensionality s: $F_{l+s}^{(s)} = F_l^{(s)} + F_{l+1}^{(s)} + ... + F_{l+s-1}^{(s)}, l = 0, 1, \ldots$ with initial conditions: $F_0^{(s)} = F_1^{(s)} = \ldots = F_{s-2}^{(s)} = 0, F_{s-1}^{(s)} = 1$, for $l = 0, 1, \ldots$.

4 Numerical examples

We considered three different examples of 4,7,10 and 30-dimensional integrals, respectively, for which we have computed their reference values.

Example 1. s = 4.

$$\int\limits_{[0,1]^4} x_1 x_2^2 e^{x_1 x_2} \sin(x_3) \cos(x_4) \approx 0.108975. \tag{5}$$

Example 2. s = 7.

$$\int\limits_{[0,1]^7} e^{1-\sum\limits_{i=1}^{3} sin(\frac{\pi}{2}.x_i)} \times arcsin\left(sin(1)+\frac{\sum\limits_{j=1}^{7} x_j}{200}\right) \approx 0.7515. \tag{6}$$

Example 3. s = 10.

$$\int\limits_{[0,1]^{10}} \frac{4x_1x_3^2e^{2x_1x_3}}{(1+x_2+x_4)^2} e^{x_5+\cdots+x_{10}} \approx 14.808435. \tag{7}$$

Example 4. s= 30.

$$\int\limits_{[0,1]^{30}} \frac{4x_1x_3^2e^{2x_1x_3}}{(1+x_2+x_4)^2} e^{x_5+\cdots+x_{20}} x_{21}\ldots x_{30} \approx 3.244. \tag{8}$$

The results are given in the tables below. We make a comparison between plain Monte Carlo (CRUDE), Adaptive Monte Carlo approach (ADAPT), Fibonacci lattice sets (FIBO), Sobol sequence (SOBOL) and Matousek scrambling for Sobol sequence (SCRAMBLE). Each table contains information about the stochastic approach which is applied, the obtained relative error (RE), the needed CP-time in seconds and the number of points. Note that when the FIBO method is tested, the number of sampled points are always generalized Fibonacci numbers of the corresponding dimensionality.

Table 1. Algorithm comparison of the rel.errors for the 4-dimensional integral.

# of points	CRUDE	t,s	ADAPT	t,s	FIBO	t,s	SOBOL	t,s	SCRAMBLE	t,s
1490	5.25e-3	0.002	2.08e-4	0.07	1.01e-3	0.004	9.46e-4	0.43	3.78e-3	0.47
10671	1.83e-3	0.01	2.98e-4	1.09	8.59e-5	0.02	5.28e-4	1.4	6.10e-4	1.59
20569	6.59e-4	0.02	2.44e-4	1.74	3.89e-5	0.03	3.52e-5	4.32	1.97e-5	4.54
39648	1.04e-3	0.06	8.26e-5	4.58	3.01e-5	0.07	2.68e-5	7.77	9.67e-6	8.26
147312	3.78e-3	0.15	7.03e-5	11.98	3.71e-6	0.24	2.29e-6	23.7	1.40e-6	27.91

Numerical results show essential advantage for the lattice sets algorithm based on Fibonacci generalized numbers (in 2-4 orders) and Sobol scramble sequence in comparison with Crude algorithm and Adaptive algorithm for the lower dimensional cases. The results for relative errors corresponding to FIBO and Sobol are similar especially for higher sample number (see Tables 1,2,3). If the computational time is fixed he advantage of Fibonacci lattice sets in terms of relative error in comparison with Adaptive Monte Carlo and Crude Monte Carlo is clearly seen (see Figures 1,2,3). For 30 dimensional integral the scramble sequence outperforms the FIBO method and Sobol sequence by at least 2 orders - see Table 4. The experiments show that the Matousek linear scrambling for Sobol sequence is the best method in terms of lower relative errors with increasing the dimensionality of the integral.

Table 2. Algorithm comparison of the rel.errors for the 7-dimensional integral.

# of points	CRUDE	t,s	ADAPT	t,s	FIBO	t,s	SOBOL	t,s	SCRAMBLE	t,s
2000	6.39e-3	0.14	4.44e-4	1.62	2.81e-3	0.23	5.45e-3	1.04	2.51e-3	1.42
7936	8.51e-3	0.64	8.04e-4	6.90	1.38e-3	0.87	1.28e-3	2.08	1.16e-3	3.08
15808	2.58e-3	0.95	1.98e-4	11.26	9.19e-4	1.73	9.65e-4	3.26	7.58e-4	5.89
62725	2.55e-3	2.54	2.38e-4	29.27	2.78e-5	3.41	5.18e-4	12.3	3.11e-4	15.64
124946	2.22e-3	6.48	1.47e-4	76.46	6.87e-5	6.90	1.09e-4	25.4	8.22e-5	31.41

Table 3. Algorithm comparison of the rel.errors for the 10-dimensional integral.

# of points	CRUDE	t,s	ADAPT	t,s	FIBO	t,s	SOBOL	t,s	SCRAMBLE	t,s
1597	2.38e-2	0.002	2.43e-3	1.18	4.39e-3	0.003	6.31e-3	0.02	1.46e-3	0.05
17711	1.61e-2	0.02	8.27e-4	1.07	1.81e-3	0.04	5.31e-4	0.11	1.83e-4	0.21
121393	8.84e-3	0.15	4.42e-4	9.45	1.20e-3	0.16	1.78e-4	1.21	3.12e-5	1.47
832040	3.74e-3	0.75	5.48e-5	77.21	1.19e-5	0.70	3.24e-5	12.1	8.25e-6	14.41
3524578	5.12e-3	6.35	8.33e-6	256.37	2.63e-6	6.45	4.57e-6	121.5	7.71e-7	139.1

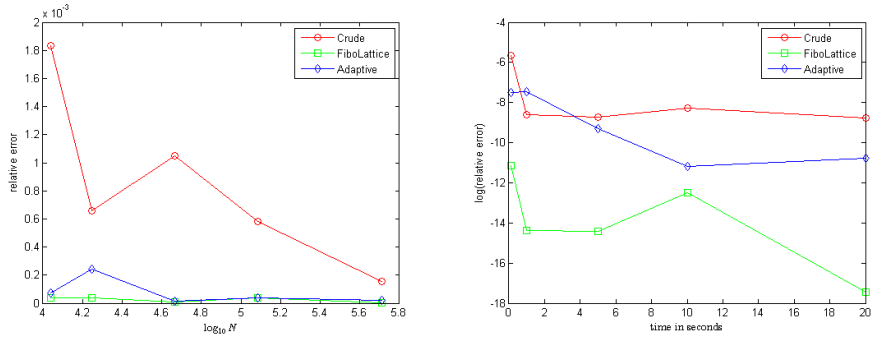

Fig. 1. Comparison of the RE for the 4-dimensional integral with different stochastic methods.

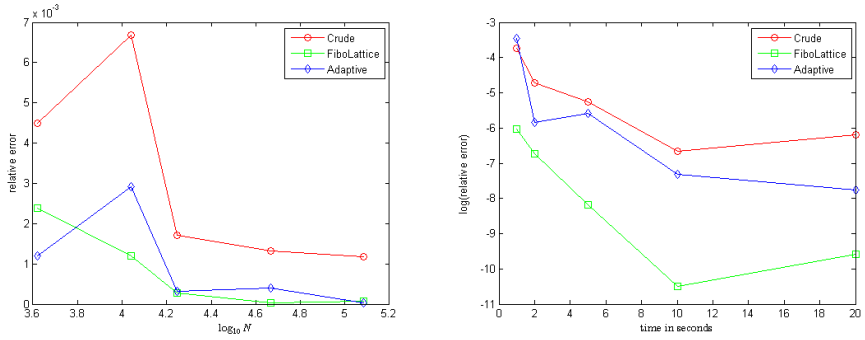

Fig. 2. Comparison of the RE for the 7-dimensional integral with different stochastic methods.

Table 4. Algorithm comparison of the rel.errors for the 30-dimensional integral.

# of points	SCRAMBLE	t,s	SOBOL	t,s	FIBO	t,s
1024	5.78e-2	0.53	1.18e-1	0.42	8.81e-1	0.02
16384	1.53e-2	5.69	8.40e-2	4.5	6.19e-1	0.14
131072	1.35e-3	42.1	1.18e-2	30.2	2.78e-1	1.16
1048576	6.78e-4	243.9	9.20e-3	168	9.86e-2	8.61

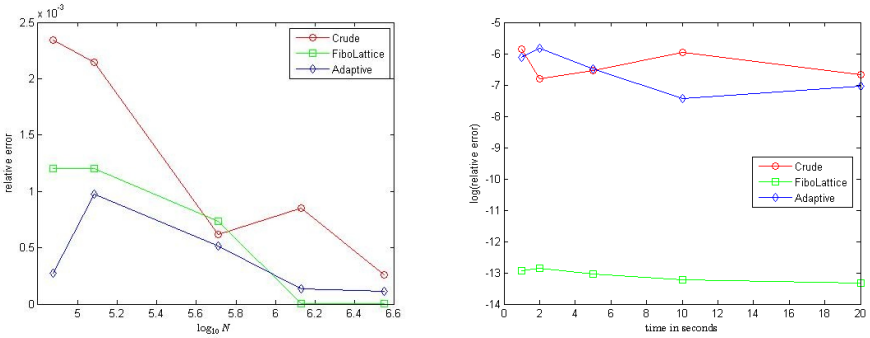

Fig. 3. Comparison of the RE for the 10-dimensional integral with different stochastic methods.

5 Conclusion

In this paper we compare the performance of different quasi-Monte Carlo and Monte Carlo algorithms for multidimensional integrals connected with Bayesian statistics used in machine learning. A comprehensive experimental study of Crude Monte Carlo, Adaptive Monte Carlo algorithm, Fibonacci lattice sets, Sobol sequence and Matousek scrambling for Sobol sequence has been done. All algorithms under consideration are efficient for evaluation the problem under consideration. These approaches are the only possible algorithms for high dimensional integrals because the deterministic algorithms needs a very large amount of time for the evaluation of the integral. Adaptive algorithm is a useful Monte Carlo technique which needs very small number of points and its strength shows when the integrand is not smooth. This study shows that quasi-Monte Carlo algorithms outperform the Monte Carlo methods for smooth functions. The Matousek scrambling for Sobol sequence gives better results compared to the Sobol sequence with increasing dimensionality of the multiple integral. Clearly, the progress in the area of artificial neural networks and machine learning is closely related to the progress in reliable algorithms for multidimensional integration.

Acknowledgement

The first author Venelin Todorov is supported by the Bulgarian National Science Fund under Young Scientists Project "Efficient Stochastic and Deterministic Methods and

Algorithms for Large-Scale Problems of Computational Mathematics", by the Bulgarian National Science Fund under Projects DN 12/5-2017 "Efficient Stochastic Methods and Algorithms for Large- Scale Problems" and DN 12/4-2017 "Advanced Analytical and Numerical Methods for Nonlinear Differential Equations with Applications in Finance and Environmental Pollution", by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security" (ICTinSES), contract No D01205/23.11.2018, financed by the Ministry of Education and Science and by the Grant BG05M2OP001-1.001-0003, financed by the Science and Education for Smart Growth Operational Program (2014-2020) and co-financed by the EU through the European structural and Investment funds.

References

1. Antonov I., Saleev V., An Economic Method of Computing LP Tau-Sequences, USSR Computational Mathematics and Mathematical Physics, Volume 19, 1980, pp. 252-256.
2. Atanassov E., A new efficient algorithm for generating the scrambled sobol sequence. In Numerical Methods and Applications (LNCS 2542), pp. 81-90, New York, 2003.
3. Bendavid J., Efficient Monte Carlo Integration Using Boosted Decision Trees and Generative Deep Neural Networks. arXiv preprint arXiv:1707.00028 2017.
4. Boyle P., Lai Y., Tan K., *Using Lattice Rules to Value Low-Dimensional Derrivative Contracts*, 2001.
5. Dimov I., Monte Carlo Methods for Applied Scientists, New Jersey, London, Singapore, World Scientific, 2008, 291p.
6. Dimov I., Georgieva R., Monte Carlo Algorithms for Evaluating Sobol' Sensitivity Indices. Math. Comput. Simul. 81(3): pp. 506–514, 2010.
7. Dimov I., Karaivanova A., Error analysis of an adaptive Monte Carlo method for numerical integration, *Mathematics and Computers in simulation* **47** pp. 201–213, 1998.
8. Hua, L.K. and Wang, Y., *Applications of Number Theory to Numerical analysis*, 1981.
9. Hong H.S., Hickernell F.J., "ALGORITHM 823: Implementing Scrambled Digital Sequences," ACM Transactions on Mathematical Software, Vol. 29, No. 2, pp. 95-109, 2003.
10.] Korobov, N.M.: Number-Theoretical Methods in Approximate Analysis, Fizmatgiz, Moscow, 1963.
11. Lin S., "Algebraic Methods for Evaluating Integrals in Bayesian Statistics," Ph.D. dissertation, UC Berkeley, May 2011.
12. Lin, S., Sturmfels B., Xu Z.: Marginal Likelihood Integrals for Mixtures of Independence Models, Journal of Machine Learning Research, Vol. 10, pp. 1611-1631, 2009.
13. Matousek J., On the L2-Discrepancy for Anchored Boxes, Journal of Complexity. Vol. 14, No. 4, 1998, pp. 527-556.
14. Owen A., Variance and Discrepancy with Alternative Scramblings. ACM Trans. on Computational Logic., pp. 1-16, 2002.
15. Paskov S., Computing High Dimensional Integrals with Applications to Finance. preprint Columbia Univ., 1994.
16. Sloan I.H. and Kachoyan P.J., Lattice methods for multiple integration: Theory, error analysis and examples, SIAM J. Numer. Anal. 24, pp. 116–128, 1987.
17. Song, J., Zhao, S., Ermon, S., A-nice-mc: Adversarial training for mcmc. In Advances in Neural Information Processing Systems, pp. 5140-5150, 2017.
18. Wang Y., Hickernell F., *An historical overview of lattice point sets*, 2002.
19. Watanabe S., Algebraic analysis for nonidentifiable learning machines. NeuralComput.(13), pp. 899-933, April 2001.