Optimized sampling for Monte Carlo simulations via dimension reduction

Nabil Kahale*
ESCP Europe
Labex Réfi and Big Data Research Center
75011 Paris, France
nkahale@escpeurope.eu

Abstract

We describe an unbiased optimized algorithm to calculate $E(f(U))$ via Monte Carlo simulation, where U is a vector of d independent random variables, and f is a function whose dependence on the i-th component of U decreases with i. The algorithm samples more often the first components of U. We show that, in certain cases, our algorithm improves upon the standard sampling algorithm by a factor proportional or nearly proportional to d.

1 Introduction

Monte Carlo simulation is often used in machine learning (Andrieu, De Freitas, Doucet and Jordan 2003, Russo and Van Roy 2014, Xiao and Zhang 2014, Allen-Zhu and Yuan 2015, Harikandeh, Ahmed, Virani, Schmidt, Konečnỳ and Sallinen 2015). This paper considers the problem of efficient estimation of $E(f(U))$ via Monte Carlo simulation, where f is a real-valued Borel-measurable function on $\mathbb{R}^d$, $U = (U_1, \ldots, U_d)$, and $U_1, \ldots, U_d$ are independent random variables. In a standard Monte Carlo scheme, this is done by simulating n independent vectors in $\mathbb{R}^d$ having the same distribution as U, and taking the average $f_{MC,n}$ of f over the n vectors. In the related quasi-Monte Carlo method (see (Niederreiter 1992) for an overview), f is evaluated at a predetermined deterministic sequence of points, derived for instance from the Sobol's sequence. In several applications, the efficiency of quasi-Monte Carlo algorithms can be improved by reordering the U_i's and/or making a change of variables, so that the value of $f(U)$ depends mainly by the first few U_i's. For instance, the Brownian bridge construction and principal components analysis have been used (Caflisch, Morokoff and Owen 1997, Acworth, Broadie and Glasserman 1998, Åkesson and Lehoczky 2000) to reduce the error in the valuation of financial derivatives via quasi-Monte-Carlo methods (see (Caflisch 1998) for related results). The relative importance of the first variables can formally be measured by calculating the effective dimension in the truncation sense, a concept defined in (Caflisch, Morokoff and Owen 1997): when the first variables are important, the effective dimension in the truncation sense is low in comparison to the nominal dimension. Concepts related to the effective dimension are studied in (Sobol 2001, Owen 2003, Liu and Owen 2006). It is shown in (Wang and Fang 2003, Wang and Sloan 2005, Wang 2006) that the Brownian bridge and/or principal components analysis algorithms substantially reduce the effective dimension of certain financial instruments in the truncation sense.

As

$$n\mathrm{Variance}(f_{MC,n}) = \mathrm{Variance}(f(U)),$$

the previously mentioned change of variables techniques do not modify $\mathrm{Variance}(f_{MC,n})$, because they do not change the distribution of $f(U)$, even though they decrease the error in QMC schemes. This paper aims to produce unbiased algorithms to estimate $E(f(U))$ by optimizing the tradeoff

*http://nkahale.free.fr

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

between the statistical error and the total running time. We assume that the variance of $f(U)$ is finite. Let τ be the expected time needed to simulate U and to calculate $f(U)$. Thus $T_{MC} = n\tau$ is the expected total running time of the standard MC scheme in n iterations, and

$$T_{MC}\text{Variance}(f_{MC,n}) = \tau\text{Variance}(f(U)). \tag{1}$$

This paper describes a Markov chain Monte Carlo scheme that calculates in n iterations an unbiased estimator f_n of $E(f(U))$ in total expected time T (we ignore the fixed preprocessing cost of the first iteration). We show that $T\text{Variance}(f_n)$ is upper-bounded by the RHS of (1), and can be substantially lower than the RHS of (1) if f depends mainly on its first components. The basic idea behind our algorithm is that, if f depends mainly on its first components, only an initial segment of arguments of f needs to be simulated at each iteration.

2 The algorithm description

Our algorithm simulates in iteration $k+1$ the first N_k arguments of f and keeps the remaining arguments unchanged, where N_k is a suitably chosen random integer in $[1, d]$. More formally, let q_i, $0 \le i \le d-1$, be a decreasing sequence of d positive numbers, with $q_0 = 1$. By convention, $q_d = 0$. Let N_k, $k \ge 1$, be a sequence of random integers in $[1, d]$ such that $\Pr(N_k > i) = q_i$ for $0 \le i \le d-1$ and $k \ge 1$. Consider a family $U_{i,k}$ of random variables, $1 \le i \le d$ and $k \ge 1$, such that $U_{i,k}$ and U_i have the same distribution. Assume the random variables N_k, $k \ge 1$, and $U_{i,k}$, $1 \le i \le d$, $k \ge 1$, are independent. Define the random d-dimensional sequence (V_k), $k \ge 1$, where $V_k = (V_{i,k})$, $1 \le i \le d$, as follows: $V_{i,1} = U_{i,1}$ and

$$V_{i,k+1} = \begin{cases} U_{i,k+1} & \text{if } i \le N_k \\ V_{i,k} & \text{otherwise.} \end{cases}$$

In other words, V_{k+1} is obtained from V_k by re-simulating the first N_k components of V_k, and keeping the remaining components unchanged. For $0 \le i \le d$, let

$$C(i) \triangleq \text{Variance}(E(f(U_1, \ldots, U_d)|U_{i+1}, \ldots, U_d)). \tag{2}$$

Thus, $C(0) = \text{Variance}(f(U))$, $C(d) = 0$, and we can interpret $C(i)$ as the variance captured by the last $d-i$ components of U. Hence, we expect $C(i)$ to be small if f depends mainly on its first i arguments. As explained in the full version of the paper, the $C(i)$'s can be estimated via Monte Carlo simulation. Set

$$f_n = \frac{f(V_1) + \cdots + f(V_n)}{n}.$$

The following theorem, which will be shown in the full version of the paper, gives a bound on the variance of f_n in terms of the $C(i)$'s.

Theorem 2.1

$$n\text{Variance}(f_n) \le C(0) - 2C(1) + 2\sum_{i=1}^{d-1} \frac{C(i) - C(i+1)}{q_i}. \tag{3}$$

Furthermore, the LHS of (3) *converges to its RHS as* n *goes to infinity.*

For $0 \le i \le d$, let T_i be the expected time needed to simulate V_{k+1} and to calculate $f(V_{k+1})$ when $N_k = i$. By convention, $T_0 = 0$. We will assume for simplicity of presentation that T_i is a strictly increasing function of i, and that $T_d = \tau$. Note that T_i may depend on the way $f(V_{k+1})$ is calculated. For instance, assume that $f(x_1, \ldots, x_d) = g(\sum_{j=1}^{d} x_j)$ for $(x_1, \ldots, x_d) \in \mathbb{R}^d$, where g is a real-valued function on $\mathbb{R}$ that can be calculated in constant time, and that each U_i can be simulated in constant time. Then $f(V_{k+1})$ can be calculated in a naive manner when $N_k = i$ by simulating $U_{j,k+1}$, $1 \le j \le i$, and calculating the sum of all components of V_{k+1}, which takes $\Theta(d)$ time. But $f(V_{k+1})$ can be calculated more efficiently in $\Theta(N_k)$ time as follows. Let $S_k = \sum_{j=1}^{d} V_{j,k}$ be the sum of the components of V_k. Since $S_{k+1} = S_k + \sum_{j=1}^{N_k}(V_{j,k+1} - V_{j,k})$, S_{k+1} can be calculated recursively in $\Theta(N_k)$ time. Thus $f(V_{k+1}) = g(S_{k+1})$ can be calculated in $\Theta(N_k)$ time, and so $T_i = \Theta(i)$.

Since $\Pr(N_k = i) = q_{i-1} - q_i$ for $1 \le i \le d$ and $1 \le k \le n-1$, and since we are ignoring the time needed for the first iteration,

$$\begin{aligned} T &= (n-1)\sum_{i=1}^{d}(q_{i-1} - q_i)T_i \\ &= (n-1)\sum_{i=0}^{d-1} q_i(T_{i+1} - T_i). \end{aligned}$$

Hence

$$\text{Variance}(f_n)T \le (C(0) - 2C(1) + 2\sum_{i=1}^{d-1}\frac{C(i) - C(i+1)}{q_i})(\sum_{i=0}^{d-1} q_i(T_{i+1} - T_i)). \tag{4}$$

The right-hand side of (4) coincides with that of (1) if $q_0 = \cdots = q_{d-1} = 1$, i.e. when $N_k = d$ almost surely for $k \ge 1$. But minimizing the right-hand side of (4) subject to the constraints $1 = q_0 \ge \cdots \ge q_{d-1} > 0$ often leads to bounds better than (1). Note that a similar optimization problem was considered in (Rhee and Glynn 2015) in a different context. It follows from (4) that

$$\text{Variance}(f_n)T \le 2(\sum_{i=0}^{d-1}\frac{C(i) - C(i+1)}{q_i})(\sum_{i=0}^{d-1} q_i(T_{i+1} - T_i)). \tag{5}$$

Remark 2.1 *A simple calculation shows that* (4) *and* (5) *still hold if the sequence* $(C(i))$, $0 \le i \le d$, *is replaced by any sequence* $(C'(i))$ *such that* $C'(i) \ge C(i)$ *for* $0 \le i \le d$, *with* $C'(d) = 0$. *Similarly,* (4) *and* (5) *still hold if the sequence* (T_i), $0 \le i \le d$, *is replaced by any sequence* (T'_i) *such that* $T'_i \ge T_i$ *for* $0 \le i \le d$, *with* $T'_0 = 0$.

3 Examples

3.1 A Lipschitz function

Assume that $f(x_1, \ldots, x_d) = g(\sum_{j=1}^{d} x_j)$ for $(x_1, \ldots, x_d) \in \mathbb{R}^d$, where g is a real-valued 1-Lipschitz function on $\mathbb{R}$ that can be calculated in constant time. For instance, $f(x_1, \ldots, x_d) = \max(\sum_{j=1}^{d} x_j - K, 0)$, where K is a constant, satisfies this condition. Assume further that $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_d > 0$, where σ_i is the standard deviation of U_i, and that each U_i can be simulated in constant time. We will show in the full version of the paper that $C(i) \le C'(i)$ for $0 \le i \le d$, with

$$C'(i) = \sum_{j=i+1}^{d} \sigma_j^2.$$

On the other hand, it was shown in Section 2 that $T_i \le ci$, where c is an absolute constant independent of d. We now set $q_i = \sigma_{i+1}/\sigma_1$ for $0 \le i < d$. By Remark 2.1, we can replace $C(i)$ and T_i in (5) by $C'(i)$ and ci, respectively, which implies that

$$\text{Variance}(f_n)T \le 2c(\sum_{i=1}^{d}\sigma_i)^2. \tag{6}$$

On the other hand, $\text{Variance}(f(U)) \le \sum_{i=1}^{d}\sigma_i^2$ since $\text{Variance}(f(U)) = C(0)$, with equality when g is the identity function. Thus the RHS of (1) is upper bounded by $cd\sum_{i=1}^{d}\sigma_i^2$, with equality (up to a constant) when g is the identity function. By the Cauchy-Schwarz inequality, for this choice of q_i's, the bound (6) on the performance of our algorithm is, up to a multiplicative constant, always better than that of the standard MC algorithm when g is the identity function. Furthermore, our algorithm outperforms the standard MC algorithm by a factor of $\Theta(d)$ in certain cases (e.g. if g is the identity function, and $\sigma_i = i^{-2}$).

3.2 Linearly decreasing probabilities

Proposition 3.1 *If $q_i = (i+1)^{-1}$ for $0 \leq i \leq d-1$, and for $0 \leq i \leq d$ and some constant c, $T_i \leq ci$, then*

$$\text{Variance}(f_n)T \leq 2c(\ln(d)+1)\sum_{i=0}^{d-1} C(i). \tag{7}$$

Proof: A simple calculation shows that

$$\sum_{i=0}^{d-1} \frac{C(i)-C(i+1)}{q_i} = \sum_{i=0}^{d-1} C(i).$$

Thus, by replacing T_i in (5) with ci, it follows that

$$\text{Variance}(f_n)T \leq 2c(\sum_{i=0}^{d-1} C(i))(\sum_{i=0}^{d-1} q_i).$$

We conclude the proof by noting that

$$\sum_{i=0}^{d-1} q_i \leq \ln(d)+1.$$

□

Thus, if $C(i) \leq \gamma^i C(0)$, where $\gamma \in (0,1)$ is an absolute constant, (this is the case in the previous example if the σ_i's are exponentially decreasing and g is the identity function), our algorithm improves upon the standard Monte Carlo algorithm by a factor of $\Theta(d/\ln(d))$.

References

Acworth, P. A., Broadie, M. and Glasserman, P. (1998). A comparison of some Monte carlo and quasi Monte carlo techniques for option pricing, *in* H. Niederreiter, P. Hellekalek, G. Larcher and P. Zinterhof (eds), *Monte Carlo and Quasi-Monte Carlo Methods 1996*, Vol. 127 of *Lecture Notes in Statistics*, Springer New York, pp. 1–18.

Åkesson, F. and Lehoczky, J. P. (2000). Path generation for quasi-Monte carlo simulation of mortgage-backed securities, *Management Science* **46**(9): 1171–1187.

Allen-Zhu, Z. and Yuan, Y. (2015). Even faster accelerated coordinate descent using non-uniform sampling, *arXiv preprint arXiv:1512.09103* .

Andrieu, C., De Freitas, N., Doucet, A. and Jordan, M. I. (2003). An introduction to MCMC for machine learning, *Machine learning* **50**(1-2): 5–43.

Caflisch, R. E. (1998). Monte carlo and quasi-Monte carlo methods, *Acta Numerica* **7**: 1–49.

Caflisch, R. E., Morokoff, W. J. and Owen, A. B. (1997). Valuation of mortgage backed securities using Brownian bridges to reduce effective dimension, *Journal of Computational Finance* **1**: 27–46.

Harikandeh, R., Ahmed, M. O., Virani, A., Schmidt, M., Konečnỳ, J. and Sallinen, S. (2015). Stopwasting my gradients: Practical SVRG, *Advances in Neural Information Processing Systems*, pp. 2251–2259.

Liu, R. and Owen, A. B. (2006). Estimating mean dimensionality of analysis of variance decompositions, *Journal of the American Statistical Association* **101**(474): 712–721.

Niederreiter, H. (1992). *Random number generation and quasi-Monte Carlo methods*, Vol. 63, SIAM.

Owen, A. B. (2003). The dimension distribution and quadrature test functions, *Statistica Sinica* **13**(1): 1–18.

Rhee, C.-h. and Glynn, P. W. (2015). Unbiased estimation with square root convergence for sde models, *Operations Research* **63**(5): 1026–1043.

Russo, D. and Van Roy, B. (2014). An information-theoretic analysis of thompson sampling, *Journal of Machine Learning Research* .

Sobol, I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte carlo estimates, *Mathematics and computers in simulation* **55**(1-3): 271–280.

Wang, X. (2006). On the effects of dimension reduction techniques on some high-dimensional problems in finance, *Operations Research* **54**(6): 1063–1078.

Wang, X. and Fang, K.-T. (2003). The effective dimension and quasi-Monte carlo integration, *Journal of Complexity* **19**(2): 101 – 124.

Wang, X. and Sloan, I. H. (2005). Why are high-dimensional finance problems often of low effective dimension?, *SIAM Journal on Scientific Computing* **27**(1): 159–183.

Xiao, L. and Zhang, T. (2014). A proximal stochastic gradient method with progressive variance reduction, *SIAM Journal on Optimization* **24**(4): 2057–2075.