The Bias-Variance Dilemma of the Monte Carlo Method

Zlochin Mark and Yoram Baram

Technion - Israel Institute of Technology, Technion City, Haifa 32000, Israel
{zmark,baram}@cs.technion.ac.il

Abstract. We investigate the setting in which Monte Carlo methods are used and draw a parallel to the formal setting of statistical inference. In particular, we find that Monte Carlo approximation gives rise to a bias-variance dilemma. We show that it is possible to construct a biased approximation scheme with a lower approximation error than a related unbiased algorithm.

1 Introduction

Markov Chain Monte Carlo methods have been gaining popularity in recent years. Their growing spectrum of applications ranges from molecular and quantum physics to optimization and learning. The main idea of this approach is to approximate the desired target distribution by an empirical distribution of a sample generated through the simulation of an ergodic Markov Chain. The common practice is to construct a Markov Chain in such a way as to insure that the target distribution is invariant. Much effort has been devoted to the development of general methods for the construction of such Markov chains. However, the fact that approximation accuracy is often lost when invariance is imposed has been largely overlooked. In this paper we make explicit the formal setting of the approximation problem, as well as the required properties of the approximation algorithm. By analogy to statistical inference, we observe that the desired property of the algorithms is a good rate of convergence, rather than unbiasedness. We demonstrate this point by numerical examples.

2 Monte Carlo Method

2.1 The Basic Idea

In various fields we encounter the problem of finding the expectation:

$$\hat{a} = E[a] = \int a(\theta)Q(\theta)d\theta \tag{1}$$

where $a(\theta)$ is some parameter-dependent quantity of interest and $Q(\theta)$ is the parameter distribution (e.g. in Bayesian learning $a(\theta)$ can be the vector of output values for the test cases and $Q(\theta)$ is the posterior distribution given the data).

G. Dorffner, H. Bischof, and K. Hornik (Eds.): ICANN 2001, LNCS 2130, pp. 141–147, 2001.

Since an exact calculation is often infeasible, an approximation is made. The basic Monte Carlo estimate of $\hat{a}$ is:

$$\bar{a}_n = \frac{1}{n}\sum_{i=1}^{n} a(\theta_i) \tag{2}$$

where θ_i are independent and distributed according to $Q(\theta_i)$. In the case where sampling from $Q(\theta)$ is impossible, an *importance sampling* [2] can be used:

$$\bar{a}_n = \frac{1}{n}\sum_{i=1}^{n} a(\theta_i)w(\theta_i) \tag{3}$$

where θ_i are distributed according to some $P_0(\theta_i)$ and $w(\theta_i) = \frac{Q(\theta_i)}{P_0(\theta_i)}$.

In many problems, especially high dimensional ones, independent sampling cannot provide accurate estimates in reasonable time. An alternative is the Markov Chain Monte Carlo (MCMC) approach, where the same estimate as in (2) is used, but the sampled values $\theta^{(i)}$ are not drawn independently. Instead, they are generated by a homogeneous ergodic distribution Q [5].

2.2 Approximation Error and the Bias-Variance Dilemma

Let us consider a yet more general approximation scheme where the invariant distribution of the Markov Chain is not Q but some other distribution Q'. Moreover, we may consider non-homogeneous Markov Chains with the transition probabilities $T_t(\theta^{(t+1)} \mid \theta^{(t)})$ depending on t.

Since $\bar{a}_n$ is a random variable, its *bias, equilibrium bias and variance* are defined as follows:

$$\begin{aligned} Bias_n &= E\{\bar{a}_n\} - \hat{a} \\ Bias^{eq} &= \lim_{n\to\infty} Bias_n \\ Var_n &= E\{\bar{a}_n - E\{\bar{a}_n\}\}^2 \end{aligned}$$

where E denotes expectation with respect to the distribution of $\bar{a}_n$.

In addition, we may define the *initialization bias*, $Bias_n^{init} = Bias_n - Bias^{eq}$, as a measure of how far the Markov Chain is from equilibrium.

The common practice is to try to construct a homogeneous (i.e. $T(\cdot \mid \cdot)$ independent of time) reversible Markov Chain with $Q(\theta)$ as invariant distribution, hence with the equilibrium bias equal to zero.

However the quality of the approximation is measured not by the bias, but by an average squared estimation error:

$$Err_n = E[\bar{a}_n - \hat{a}]^2 = Bias_n^2 + Var_n = (Bias^{eq} + Bias_n^{init})^2 + Var_n \tag{4}$$

From the last equation it can be seen that the average estimation error has three components. Therefore, there is a possibility (at least potentially) to reduce the total approximation error by balancing those three. Moreover it shows that, since the initialization bias and the variance depend on the number of iterations, the algorithm may depend on the number of iterations as well.

Batch Versus Online Estimation. A similar trade-off between the components of the error appears in statistical inference, where it is known as "the bias-variance dilemma". The analogy to the statistical inference setting can be further extended by making a distinction between two models of MCMC estimation:

Online estimation - the computation time is potentially unlimited and we are interested in as good an approximation as possible at any point in time or, alternatively, as rapid a rate of convergence of Err_n to zero as possible .

Batch estimation - the allowed computation time is limited and known in advance and the objective is to design an algorithm with as low an average estimation error as possible.

Note that the batch estimation paradigm is more realistic than the online estimation, since, in practice, the time complexity is of primary importance. In addition, in many instances of Bayesian learning, the prior distribution is not known but is chosen on some *ad-hoc* basis and the data is usually noisy. Therefore, it makes little sense to look for a time-expensive high accuracy approximation to the posterior. Instead, a cheaper rough approximation, capturing the essential characteristics of the posterior, is needed.

The traditional MCMC method overlooks the distinction between the two models. In both cases, the approximation is obtained using a homogeneous (and, usually, reversible) Markov Chain with the desired invariant distribution. It should be clear, however, that such an approach is far from optimal. In the online estimation model, a more reasonable alternative would be considering a non-homogeneous Markov Chain that rapidly converges to a rough approximation of Q and whose transition probabilities are modified with time so as to insure the asymptotic invariance of Q (hence consistency). A general method for designing such Markov Chains is described in the next subsection.

In the batch estimation model the invariance of Q may be sacrificed (i.e. equilibrium bias may be non-zero) if this facilitates a lower variance and/or initialization bias for the finite computation time.

Mixture Markov Chains. Suppose that we are given two Markov Chains - the first, M_1, is unbiased, while the second, M_2, is biased, but more rapidly mixing. Let their transition probabilities be T_t^1 and T_t^2 respectively. It will usually be the case that for small sample sizes, M_2 will produce better estimates, but as n grows, the influence of the bias becomes dominant, making M_1 preferable.

In order to overcome this difficulty, we may define a *mixture Markov chain* with mixing probabilities p_t, for which the transition is made according to T_t^1 with probability $1 - p_t$ and according to T_t^2 with probability p_t. If the sequence $\{p_t\}$ starts with $p_1 = 1$ and decreases to zero, then for small t the resulting Markov chain behaves similarly to M_2, hence producing better estimates than M_1, but as t grows, its behavior approaches that of M_1 (in particular, the resulting estimate is asymptotically unbiased).

Clearly, if both Markov Chains are degenerate (i.e. produce IID states), it can be easily seen that the bias behaves asymptotically as

$$Bias_n = O(\frac{1}{n}\sum_{i=1}^{n} p_i) \tag{5}$$

and it can be shown that (5) also holds for any *uniformly ergodic* Markov Chains.

In order to balance between the variance, which typically decreases as $O(\frac{1}{n})$, and the bias, the sequence $\{p_i\}$ should be chosen so that $Bias_n = O(\frac{1}{\sqrt{n}})$, i.e. $\sum_{i=1}^{n} p_i = O(\sqrt{n})$.

This mixture Markov Chain approach allows to design efficient sampling algorithms both in online and in batch settings (in the latter case the sequence $\{p_i\}$ may depend on the sample size, n).

3 Examples

In this section we present two approaches for designing approximation algorithms with a small controlled bias. Section 3.1 describes a post-processing scheme based on the smoothing techniques, which reduces the variability of the importance weights in the Annealed Importance sampling algorithm [6]. Section 3.2 compares the well-known Hybrid Monte Carlo algorithm with a simple biased modification and a mixture Markov Chain. In both cases, the empirical comparison shows superiority of the biased algorithms.

3.1 Independent Sampling

Annealed Importance Sampling. In the Annealed Importance sampling algorithm [6], the sample points are generated using some variant of simulated annealing and then the importance weights, $w^{(i)}$, are calculated in a way that insures asymptotic unbiasedness of the weighted average.

The annealed importance sampling (AIS) estimate of $E_Q[a]$ is:

$$\bar{a}_n = \sum_{i=1}^{n} a(x^{(i)}) w^{(i)} \ / \ \sum_{i=1}^{n} w^{(i)} \tag{6}$$

Smoothed Importance Sampling. While implicitly concentrating on the regions of high probability, AIS estimate can still have a rather high variance, since its importance weights are random and their dispersion can be large.

Let us observe that $E_Q[a]$ may be written as

$$E_Q[a] = \frac{E_P[aw]}{E_P[w]} = E_{P(a)} \left\{ a \frac{E_{P(w|a)}[w]}{E_P[w]} \right\} \tag{7}$$

where $a = a(x)$ and P is the distribution defined by the annealing algorithm. The estimate (6) is simply a finite sample approximation to the first part of (7).

Next, the second part of (7) suggests using an estimate:

$$\hat{a}_n = \sum_{i=1}^{n} a(x^{(i)}) \hat{w}(a(x^{(i)})) \tag{8}$$

where $\hat{w}(a)$ is an estimate of $\frac{E_{P(w|a)}[w]}{E_P[w]}$. on the value of a). It can be shown that if $\hat{w}(a)$ was known exactly, than (8) would have a lower variance than (6). In

Table 1. Average estimation error of posterior output prediction for different sample sizes. The results for sample size n are based on $10^4/n$ trials.

n	Annealed IS	Smoothed IS
10	3.04	1.09
100	0.694	0.313
1000	0.0923	0.0594

practice, a good estimate of $\hat{w}(a)$ can be obtained using some data smoothing algorithm such as kernel smoothing [3]. The resulting method henceforth is referred to as *Smoothed Importance sampling*. The usage of smoothing introduces bias (which can be made arbitrary small by decreasing the smoothing kernel width to zero as $n \to \infty$), but in return can lead to a significant reduction of the variance, hence, a smaller estimation error than (6), as demonstrated by the following numerical experiment.

A Bayesian Linear Regression Problem. The comparison was carried out using a simple Bayesian learning problem from [6]. The data for this problem consisted of 100 independent cases, each having 10 real-valued predictor variables, $x_1, \ldots, x_{10}$, and a real-valued response variable, y, which is modeled by $y = \sum_{k=1}^{10} \beta_k x_k + \epsilon$, where ϵ is zero-mean Gaussian noise with unknown variance σ^2. The detailed description of the data-generation model can be found in [6].

The two methods were used to estimate the posterior prediction for a test set of size 100. The "correct" posterior prediction were estimated using Annealed Importance sampling with sample size 10000.

As can be seen in Table 1, the average estimation error of Smoothed Importance sampling was uniformly lower than that of Annealed Importance sampling. It should also be noted that the modeling error, i.e. the squared distance between the Bayesian prediction and the correct test values, was 5.08, which is of the same order magnitude as the estimation error for $n = 10$. This confirms our claim that, in the context of Bayesian learning, high accuracy approximations are not needed.

3.2 MCMC Sampling

HMC Algorithm. The Hybrid Monte Carlo (HMC) algorithm [1] is one of the state-of-the-art asymptotically unbiased MCMC algorithms for sampling from complex distributions. The algorithm is expressed in terms of sampling from a *canonical* distribution defined in terms of the energy function $E(q)$:

$$P(q) \propto \exp(-E(q)) \tag{9}$$

To allow the use of dynamical methods, a "momentum" variable, p, is introduced, with the same dimensionality as q. The canonical distribution over the "phase space" is defined to be:

$$P(q,p) \propto \exp(-H(q,p)) = \exp(-E(q) - \sum_{i=1}^{n} \frac{p_i^2}{2}) \tag{10}$$

where $p_i, i = 1, \ldots, n$ are the momentum components.

Sampling from the canonical distribution can be done using the *stochastic dynamics* method, in which simulation of the *Hamiltonian dynamics* of the system using *leapfrog* discretization is alternated with Gibbs sampling of the momentum.

In the hybrid Monte Carlo method, the bias, introduced by the discretization, is eliminated by applying the Metropolis algorithm to the candidate states, generated by the stochastic dynamic transitions. The candidate state is accepted with probability $\max(1, \exp(\Delta H))$, where ΔH is the difference between the Hamiltonian in the beginning and and the end of the trajectory [5].

A modification of the Gibbs sampling of the momentum, proposed in [4], is to replace p each time by $p \cdot \cos(\theta) + \zeta \cdot \sin(\theta)$, where θ is a small angle and ζ is distributed according to $N(0, I)$. While keeping the canonical distribution invariant, this scheme, called *momentum persistence*, allows the use of shorter (hence cheaper) trajectories. However, in order to insure invariance of the target distribution, the momentum has to be reversed in case the candidate state has been rejected by the Metropolis step. This causes occasional backtracking and slows the sampling down.

Stabilized Stochastic Dynamics. As an alternative to the HMC algorithm we consider stochastic dynamics with momentum persistence, but without the Metropolis step. In order to insure stability of the algorithm, if the momentum size grows beyond a certain percentile of its distribution (say, 99%), it is replaced using Gibbs sampling.

The resulting Stabilized Stochastic Dynamics (SSD) algorithm is biased. However, since it avoids backtracking, it can produce a lower estimation error, as found in experiments described next.

Bayesian Neural Networks. We compared the performances of SSD, HMC and mixture Markov Chain with

$$p_t^n = \begin{cases} 1, & \text{if } t \leq \sqrt{200n} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

on the following Bayesian learning problem. The data consisted of 30 input-output pairs, generated by the following model:

$$y_i = \frac{\sin(2.5|x_i|)}{2.5|x_i|} + \epsilon_i, i = 1, \ldots, 20 \tag{12}$$

where x_i are independently and uniformly generated from $[-\pi, \pi]$ and ϵ_i are Gaussian zero-mean noise variables with standard deviation 0.1.

We have used multilayer perceptron with one input node, one hidden layer containing 5 tanh nodes and one linear output node. The prior for each weight was taken to be zero-mean Gaussian with inverse variance 0.1.

The comparison was carried out using two test statistics: average log-posterior and average prediction for 100 equidistant points in $[-\pi, \pi]$. Since the correct values of those statistics are not known, they were estimated using a sample of

Table 2. Average estimation error of mean log-posterior and mean output prediction for different sample sizes.

	Log-posterior			Output prediction		
n	SSD	HMC	MIX	SSD	HMC	MIX
1000	3.55	8.52	5.51	0.053	0.102	0.061
3000	0.75	1.34	0.85	0.015	0.031	0.024
10000	0.23	0.43	0.37	0.0049	0.0089	0073

size 10^6 generated by HMC. For each algorithm we performed 100 runs with initial state generated from the prior.

As can be seen from the results in Table 2, both SSD and mixture Markov Chain produced results which are uniformly better than those of HMC. It may be hypothesized that, for very large sample sizes, HMC will become superior to SSD, because of the influence of the bias. However, the mixture Markov Chain for these large sample sizes is expected to produce results which are very similar to HMC. More importantly, the large sample behavior of the algorithms is not very relevant, as the modeling error, i.e. the squared distance between the Bayesian prediction and the correct test values, in this experiment was 0.187, which is larger than the estimation error for $n = 1000$, meaning that, once again, there is no use in making a high accuracy large sample approximation.

4 Conclusion

We have shown that in Monte Carlo estimation there is a "bias-variance dilemma", which has been largely overlooked. By means of numerical examples, we have demonstrated that bias correcting procedures can, in fact, increase the estimation error. As an alternative, approximate (possibly asymptotically biased) estimation algorithms, with lower variance and convergence bias should be considered. Once the unbiasedness requirement is removed, a whole range of new possibilities for designing sampling algorithms opens up, such as automatic discretization step selection, different annealing schedules, alternative discretization schemes etc.

References

1. S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195:216–222, 1987.
2. W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman&Hall, 1996.
3. W. Hardle. *Applied nonparametric regression*. Cambridge University Press, 1990.
4. A. M. Horowitz. A generalized guided Monte Carlo algorithm. *Physics Letters B*, 268:247–252, 1991.
5. R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996.
6. R. M. Neal. Annealed importance sampling. Technical Report No. 9805, Dept. of Statistics, University of Toronto, 1998.