The bias-variance dilemma of the Monte Carlo method

Mark Zlochin and Yoram Baram*

January 7, 2001

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 schemes with a lower approximation error than a related unbiased algorithms.

1 Introduction

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 is to approximate the desired target distribution by an empirical distribution of a sample generated by an ergodic source. The common practice is to impose unbiasedness using bias-correcting schemes such as importance weighting or the Metropolis algorithm. However, the fact that approximation accuracy is often lost when unbiasedness 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:

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

(1)

where $a(\theta)$ is some parameter-dependent quantity of interest and $p(\theta)$ is the parameter distribution (e.g. in Bayesian learning $a(\theta)$ can be the vector of output values for the test cases and $p(\theta)$ is the posterior distribution given the data). Since an exact calculation is often infeasible, an approximation is
made. The basic Monte Carlo estimate of $\hat{a}$ is:

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

(2)

where $\theta_i$ are independent and distributed according to $Q(\theta_i)$. In the case where sampling from $Q(\theta)$ is impossible, an importance sampling [Gilks et al., 1996] can be used:

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

(3)

where $\theta_i$ are distributed according to some $P_0(\theta_i)$ and $w(\theta_i) = \frac{Q(\theta_i)}{P_0(\theta_i)}$. Note that the basic method (2) is a particular case of importance sampling with $Q(\theta_i) = P_0(\theta_i)$.

In many problems, especially high dimensional ones, independent sampling cannot provide accurate estimates in reasonable time. The Markov Chain Monte Carlo method described next can be more appropriate.

### 2.2 Markov Chain Monte Carlo

A Markov chain is a random process defined by an initial distribution for the first state of the chain, $P(\theta^{(1)})$, and a set of transition probabilities $T(\theta^{(i+1)} | \theta^{(i)})$ for a new state, $\theta^{(i+1)}$, to follow the current state, $\theta^{(i)}$. An invariant distribution, $Q$, is one that persists once it is established. This invariance property can be written as follows:

$$Q(\theta') = \int T(\theta' | \theta)Q(\theta)d\theta$$

(4)

Invariance with respect to $Q$ is implied by a stronger condition of detailed balance:

$$T(\theta' | \theta)Q(\theta) = T(\theta | \theta')Q(\theta')$$

(5)
A chain satisfying the detailed balance condition is said to be reversible.

A Markov chain is called ergodic if it has a unique invariant distribution, the equilibrium distribution, to which it converges from any initial distribution [Neal, 1996].

In the Markov Chain Monte Carlo (MCMC) approach, the same estimate as in (2) is used, but the sampled values $\theta^{(t)}$ are not drawn independently. Instead, they are generated by an ergodic Markov Chain with invariant distribution $Q$.

The classical Monte Carlo approach is a particular instance of MCMC where $P(\theta^{(t+1)}) = T(\theta^{(t+1)} | \theta^{(t)}) = Q(\theta^{(t+1)})$.

3 The bias-variance dilemma

3.1 Approximation error and the bias-variance decomposition

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 $T_t(\theta^{(t+1)} | \theta^{(t)})$ depending on $t$.

Since $\bar{a}_n$ is a random variable, its bias, equilibrium bias and variance are defined as follows:
$$Bias_n = E\{\bar{a}_n\} - \hat{a}$$
$$Bias^{eq} = \lim_{n \to \infty} Bias_n$$
$$Var_n = E\{\bar{a}_n - E\{\bar{a}_n\}\}^2$$

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

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

The common practice is to try to construct a homogeneous (i.e. $T(\cdot | \cdot)$ independent of time) reversible Markov Chain with $p(\theta)$ 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:

$$Err_n = E[\bar{a}_n - \hat{a}]^2$$

(6)

The average estimation error can be easily shown to be:

$$Err_n = Bias_n^2 + Var_n = (Bias^{eq} + Bias_{\text{init}}^n)^2 + Var_n$$

(7)

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.
3.2 Batch versus online estimation

A similar trade-off between the components of the error appears in statistical estimation, where it is known as “the bias-variance dilemma”. The analogy to the statistical estimation 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 applications, 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 often 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).

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.

4 Examples

4.1 Independent sampling

4.1.1 Importance sampling versus sample mean

Let us start with a simple one-dimensional example. Consider $x \in \mathbb{R}$ whose distribution $Q(x)$ is $N(0.1, 11^2)$, i.e., Gaussian with mean 0.1 and standard deviation 11. Assume that direct sampling from $Q(x)$ is impossible, but we have an approximation $P = N(0, 10^2)$. An unbiased estimate of $E_Q(x)$ can be obtained using importance sampling:

$$\bar{x}_n = \frac{1}{n} \sum_{i=1}^{n} \frac{x^{(i)} Q(x^{(i)})}{P(x^{(i)})}$$

(8)

where $\{x^{(i)}\}_{i=1}^{n}$ are generated independently from $P(x)$. The sample mean was also calculated for comparison.

The two methods were compared by simulating 1000 independent trials. The second method is biased, but, as can be seen in table 1, it has a lower variance. As a result, it also has lower error for a wide range of sample sizes.
Table 1: Mean squared error and variance of Monte Carlo estimates for $x$ (for $\text{dim} = 1$) and $x_1$ (for $\text{dim} = 10$) using importance sampling and sample mean from the “approximate” distribution for different samples sizes $n$. The results are based on 1000 independent trials.

Only when the sample size is really high, does the influence of the bias tip the balance in favor of importance sampling.

We have also considered a multidimensional case, where $\bar{x} \in \mathbb{R}^{10}$, and its components $x_1, \ldots, x_{10}$ are independent and distributed according to $\mathcal{N}(0.1, 11^2)$. The approximation distribution was $\mathcal{N}(0, 10^2 I)$. The results for the multidimensional case, shown in table 1, are even more in favor of the biased method.

Naturally, the sample mean approximation will fail in case the approximating distribution significantly differs from the target distribution. This leads us to two fundamental questions:

1. How can a good approximating distribution be obtained?
2. How can an estimate with a low variance and a small bias be obtained using the sample generated from the approximation distribution?

Several answers to the first question have been proposed (e.g., [Schuermans, 2000]; [Neal, 1998]). The Annealed Importance Sampling [Neal, 1998], described next, seems to be the most general method proposed so far.

### 4.1.2 Annealed importance sampling

In the annealed importance sampling algorithm, the sample points are generated using some variant of simulated annealing [Kirkpatrick et al., 1983] and then the importance weights are calculated in a way that insures asymptotic unbiasedness of the weighted average.

Specifically, a sequence of distributions is used, with probability densities given by \( Q(x) = p_0(x), p_1(x), \ldots, p_n(x) \). This sequence interpolates between some simple distribution \( p_n(x) \), from which the sampling is easy (e.g. standard Gaussian), and the target distribution \( Q \). It is assumed that, for each distribution, \( p_j \), it is possible to compute some function \( f_j(x) \propto p_j(x) \). In [Neal, 1998] the sequence is defined by:

\[
  f_j(x) = f_0(x)^{\beta_j} f_n(x)^{1-\beta_j}
\]

where \( 1 = \beta_0 > \beta_1 > \ldots > \beta_n = 0 \).

In addition, for each distribution, \( p_j \), there exists a transformation \( T_j \) that leaves \( p_j \) invariant. This transformations may be constructed using methods used in Markov Chains modeling, such as, for example, the Metropolis algorithm [Metropolis et al., 1953] and Gibbs sampling [Geman and Geman, 1984].
An annealing run begins with an initial state, $x_{n-1}$, generated from $p_n(x)$ and proceeds by applying transformations $T_{n-1}, \ldots, T_1$. As a result, a sequence $x_{n-1}, x_{n-2}, \ldots, x_1, x_0$ is generated, where $x_{j-1}$ is the outcome of $T_j$ applied to $x_j$.

The output of the $i$-th annealing run is $x^{(i)} = x_0$ and the weight $w^{(i)}$ defined as:

$$w^{(i)} = \frac{f_{n-1}(x_{n-1}) f_{n-2}(x_{n-2}) \cdots f_1(x_1) f_0(x_0)}{f_n(x_{n-1}) f_{n-1}(x_{n-2}) \cdots f_2(x_1) f_1(x_0)}$$  \hspace{1cm} (10)

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

$$\bar{a}_n = \frac{\sum_{i=1}^{n} a(x^{(i)}) w^{(i)}}{\sum_{i=1}^{n} w^{(i)}}$$  \hspace{1cm} (11)

4.1.3 Smoothed importance sampling

While implicitly concentrating on the regions of high probability and dealing quiet successfully with mutimodality, the annealed importance sampling algorithm still has two major drawbacks:

1. When several modes are present, the algorithm may systematically “undersample” certain narrow modes, i.e. the proportion of points falling under this mode is very low. Since this “undersampling” is compensated by large importance weights, the resulting estimate has a high variance.

2. The weights are random and their dispersion (even within the same mode) can be large, leading again to a high variance of the estimate.
A solution for both these problems is proposed in a companion paper [Zlochin and Baram, 2000]. In the present paper, however, we confine our attention to the second problem, answering the question:

*How can an estimate with a low variance and a small bias be obtained using the sample generated from the approximation distribution?*

First, let us observe that $E_Q[a]$ may be written as

$$
E_Q[a] = \frac{E_P[aw]}{E_P[w]} = E_{P|x}\left\{ a \frac{E_{P[w|x]}[w]}{E_P[w]} \right\}
$$

(12)

where $a = a(x)$ and $P$ is the distribution defined by the annealing algorithm. The estimate (11) is simply a finite sample approximation to the first part of (12).

Next, the second part of (12) suggests using an estimate:

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

(13)

where $\hat{w}(a)$ is an estimate of $\frac{E_{P[w|x]}[w]}{E_P[w]}$ (i.e. the normalized expectation of the importance weight conditional on the value of $a$). It can be shown that if $\hat{w}(a)$ was known exactly, than (13) would have a lower variance than (11). A proof is given in [Zlochin and Baram, 2000]. In practice, a good estimate of $\hat{w}(a)$ can be obtained using some data smoothing algorithm such as kernel smoothing, running means etc. [Hardle, 1990]. The resulting method is called, for obvious reasons, *smoothed importance sampling*. While introducing a small bias, this method can lead to a significant reduction of the variance, hence, a smaller estimation error than (11), as demonstrated by the following numerical experiments.
4.1.4 Two simple distributions

We have compared Annealed Importance sampling and Smoothed Importance sampling using two simple distributions over $\mathbb{R}^6$, employed in [Neal, 1998]. The first, unimodal, distribution is Gaussian with the components, $x_1$ to $x_6$, being independent with mean 1 and variance 0.01. The second distribution is a weighted mixture of two Gaussians, with independent components having the same means and standard deviations. One of the Gaussians, with a weight 1/3, had a mean 1 and a standard deviation 0.1 for each of the components. The second Gaussian, with a weight 2/3, had a mean 1 and a standard deviation 0.05 for each of the components. In both experiments the statistic of interest was $a(x) = x_1$.

The implementation of Annealed Importance sampling in all the experiments was exactly as in [Neal, 1998], therefore, we give only a brief overview here.

The sequence of 200 interpolating distributions was defined by (9), with $p_n$ being a Gaussian $N(0, I)$. The transformations $T_j$ consisted of 10 repetitions of a sequence of three Metropolis updates with Gaussian proposal distributions centered in the current state and having covariances $0.05^2 I$, $0.15^2 I$, $0.5^2 I$.

In the Smoothed Importance sampling the estimate of the function $\hat{w}(a)$ was obtained using a kernel smoother with Epanechnikov kernel [Hardle, 1990] of width 1, i.e. $\hat{w}(a)$ was estimated to be a weighted average of the weights produced by Annealed Importance sampling:

$$\hat{w}(a) = \frac{\sum_{a'} w(a') k(a, a')}{\sum_{a'} w(a')}$$

(14)

where $k(a, a') = 1 - (a - a')^2$, for $|a - a'| < 1$, and zero otherwise.
A typical result of such smoothing is shown in Figure 1. As can be seen from the graph, the smoothing eliminates random variations while preserving the correct weighting of the different modes.

The average estimation error of Annealed Importance Sampling and Smoother Importance sampling for both problems is reported in table 2.

In the unimodal case Smoother Importance sampling produced uniformly better (by factor 2) results than Annealed Importance sampling. In the bimodal case the results are statistically indistinguishable. This is due to the fact, that the main source of variability in the second problem is not the random dispersion of weights, but “undersampling” of the left mode - only about 3% of the sample points fell inside it. This “undersampling” is compensated by large weights which are preserved under smoothing. The only way to reduce the estimations error in this case is to modify the initial distribution $p_n$, as discussed in [Zlochin and Baram, 2000].
Table 2: Mean squared error of Monte Carlo estimates for $x_1$ using Annealed Importance sampling and Smoothed Importance sampling for different sample sizes $n$. The results (normalized by $\frac{1}{n}$) for sample size $n$ are based on 100000/n independent trials.

4.1.5 A Bayesian linear regression problem

An additional comparison was carried out using a simple Bayesian learning problem from [Neal, 1998]. 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

$$y = \sum_{k=1}^{10} \beta_k x_k + \epsilon$$

(15)

where $\epsilon$ is zero-mean Gaussian noise with unknown variance $\sigma^2$. The 100 cases were generated from this model with $\sigma^2 = 1$ and with $\beta_1 = 1$, $\beta_2 = 0.5$, $\beta_3 = -0.5$, and $\beta_k = 0$, $4 \leq k \leq 10$. The predictor variables were generated from a multivariate Gaussian with unit variance for each $x_k$ and with correlations 0.9 between each pair of $x_k$. The inverse variance $1/\sigma^2$ was given a gamma prior with mean $1/0.1^2$ and shape parameter 0.5. The prior

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c||c|c|}
\hline
 & Unimodal problem & & Bimodal problem & \\
\hline
n & Annealed IS & Smoothed IS & Annealed IS & Smoothed IS \\
\hline
10 & 0.0173 & 0.0101 & 14.0 & 14.0 \\
100 & 0.0207 & 0.0105 & 26.9 & 26.9 \\
1000 & 0.0257 & 0.0116 & 17.4 & 17.6 \\
10000 & 0.0362 & 0.0205 & 7.07 & 7.08 \\
\hline
\end{tabular}
\end{table}
distribution of $\beta_k$ was Cauchy with hyperparameter $\nu$, whose reciprocal was given a gamma prior with mean $1/0.05^2$ and shape parameter 0.25.

An interpolating sequence of 100 distributions based on (9) was used. The transformations $T_j$ were a combination of Gibbs sampling for $\sigma^2$ and Hybrid Monte Carlo [Duane et al., 1987], described in the next section, for the $\beta_k$.

Two test statistics were used for comparison: average prediction for a test set of size 100 and the average posterior log-probability. The “correct” values of both statistics were estimated using Annealed Importance sampling with sample size 10000.

The smoothing window width for the Smoothed importance sampling with sample size 10 was taken to be the span of values of the statistic $a$ and scaled by a factor of $n^{-1/5}$ for larger sample sizes $n$. Such a scaling is asymptotically optimal in a sense of mean squared error [Hardle, 1990]. In particular, it insures asymptotic unbiasedness of the estimates. The results of the comparison are shown in Table 3.

For the mean output prediction, the 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. For the mean log-posterior, Smoothed Importance sampling produced better estimates for $n = 10$ and comparable results for $n = 100$. With $n = 1000$, the bias, introduced by the smooth-
<table>
<thead>
<tr>
<th>n</th>
<th>Output prediction</th>
<th>Log-posterior</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Annealed IS</td>
<td>Smoothed IS</td>
</tr>
<tr>
<td>10</td>
<td>3.04</td>
<td>1.09</td>
</tr>
<tr>
<td>100</td>
<td>0.694</td>
<td>0.313</td>
</tr>
<tr>
<td>1000</td>
<td>0.0923</td>
<td>0.0594</td>
</tr>
</tbody>
</table>

Table 3: Average estimation error of mean output prediction and mean log-posterior for different sample sizes. The results for sample size $n$ are based on $10^4/n$ trials.

The annealed importance sampling algorithm overweighted the variance, leading to larger error. However, a more careful choice of smoothing window width leads to smaller errors, as discussed in [Zlochin and Baram, 2000].

### 4.2 Dependent sampling

#### 4.2.1 HMC algorithm

The HMC algorithm [Duane et al., 1987] 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 $E(q)$:

$$P(q) \propto \exp(-E(q))$$

Note that any probability density that is nowhere zero can be put in this form, by simply defining $E(q) = -\log P(q) - \log Z$, for any convenient $Z$. 

16
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:

$$P(q, p) \propto \exp(-H(q, p))$$  \hspace{1cm} (17)

where $H(q, p) = E(q) + K(p)$ is the “Hamiltonian”, which represents the total energy. $K(p)$ is the “kinetic energy” due to momentum, defined as

$$K(p) = \sum_{i=1}^{n} \frac{p_i^2}{2m_i}$$  \hspace{1cm} (18)

where $p_i, i = 1, \ldots, n$ are the momentum components and $m_i$ is the “mass” associated with $i$’th component, so that different components can be given different weight.

In this canonical distribution $p$ and $q$ are independent, $q$ having the marginal distribution $\exp(-E(q))$, and $p$ - the Gaussian distribution. Now we can proceed by defining a Markov chain that converges to the canonical distribution for $q$ and $p$, and then simply ignore the $p$ values [Neal, 1996].

Sampling from the canonical distribution can be done using the stochastic dynamics method, in which the task is split into two - sampling uniformly from values of $q$ and $p$ with a fixed total energy, $H(q, p)$, and sampling states with different values of $H$. The first task is done by simulating the Hamiltonian dynamics of the system:

$$\frac{dq_i}{d\tau} = + \frac{\partial H}{\partial p_i} = \frac{p_i}{m_i}$$

$$\frac{dp_i}{d\tau} = - \frac{\partial H}{\partial q_i} = - \frac{\partial E}{\partial q_i}$$
In practice, the Hamiltonian dynamics cannot be simulated exactly, but can be approximated by some discretization using finite time steps. One common approximation is \textit{leapfrog} discretization [Neal, 1996].

Different energy levels are obtained by occasional Gibbs sampling [Geman and Geman, 1984] of the momentum. In the case of (18), this can be easily done, since the distribution of \( p \), given \( q \), is the Gaussian \( N(0, I) \).

The finite time step discretization introduces bias, because the Hamiltonian is no longer preserved by the dynamic transitions. In the hybrid Monte Carlo method, the bias is eliminated by applying the Metropolis algorithm [Metropolis et al., 1953] to the candidate states, generated by the stochastic dynamic transitions. The candidate state is accepted with probability \( \max(1, \exp(\Delta H)) \), where \( \Delta H \) is the difference between the Hamiltonian in the beginning and and the end of the trajectory.

A modification of the Gibbs sampling of the momentum, proposed in [Horowitz, 1991], is to replace \( p \) each time by \( p \cdot \cos(\theta) + \zeta \cdot \sin(\theta) \), where \( \theta \) is a small angle and \( \zeta \) is distributed according to \( N(0, I) \). While keeping the canonical distribution invariant, this scheme, called \textit{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.
4.2.2 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, the following modifications are introduced:

1. If the momentum size grows beyond a certain percentile of its distribution (say, 95%), it is replaced using Gibbs sampling.

2. If the dynamic transition produces a candidate state for which $\Delta H$ is above a certain threshold (in our experiments we used 0.3), the candidate state is rejected and the discretization step size for consequent transitions is decreased. After a few steps, if $\Delta H$ remains small, the step size is restored.

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.

4.2.3 Sampling from an anisotropic Gaussian

In many Bayesian learning problems, the parameter distribution is anisotropic. For example, in the experiment described in the next section, the condition number (i.e. the ratio of the largest eigenvalue to the smallest one) of the Hessian of the log-posterior was as high as $10^7$.

We compared the performance of HMC and SSD on two-dimensional Gaussian distributions with condition number ranging from $10^2$ to $10^5$. Specif-
ically, the weighted squared error:

\[ Err(x) = Err(x_1, x_2) = ((x_1 - 10)^2 + 10^k x_2^2)/2, \quad k = 2, 3, 4, 5 \]  \hspace{1cm} (19)

was used as an energy function.

The initial distribution for the first state of the chain was an isotropic Gaussian \( N(0, I) \). Both algorithms used momentum persistence with \( \cos(\theta) = 0.95 \) and dynamic transitions consisting of 10 leapfrog steps. Following [Rasmussen, 1996], the discretization timestep was chosen so as to keep the rejection rate in HMC below 5%.

For each combination of \( k \) and \( n \) (sample size), the simulation was performed 100 times (using the same random starting value for both algorithms). The first 10% of each sample were discarded and the averages \( \bar{x} \) and \( \overline{Err(x)} \) were calculated using the remaining 90%. The estimation error of \( \bar{x} \) was calculated using (19), which gives a weighted squared distance to the true mean. The correct value of \( E_Q \{Err(x)\} \) can be calculated analytically in this case - it is equal to 1 - hence, the squared estimation error of \( \overline{Err(x)} \) could be computed as well.

The results, normalized by \( \frac{1}{n} \), are reported in tables 4 and 5. The estimation error of \( E_Q \{x\} \) is uniformly smaller for SSD than for HMC. This is apparently caused by the fact that the asymptotic bias in \( \bar{x} \), introduced by SSD, is negligible. As to \( \overline{Err(x)} \), SSD produces better estimates when the sample size is relatively small. The sample size range for which SSD is better grows with the condition number - for \( k = 4 \) and \( k = 5 \), SSD produced better results for all the sample sizes. It should also be noted that for \( k = 5, n = 1000 \) both algorithms failed to converge, but SSD still produced
Table 4: Average estimation error of the $E_Q\{x\}$ for different condition numbers, $k$, and sample sizes, $n$. The results are normalized by $\frac{1}{n}$.

Table 5: Average estimation error of the $E_Q\{Err(x)\}$ for different condition numbers, $k$, and sample sizes, $n$. The results are normalized by $\frac{1}{n}$.

ten-fold better results than HMC.

4.2.4 Bayesian neural networks

Finally, we compared the performances of SSD and HMC on the following Bayesian learning problem. The data consisted of 30 input-output pairs, generated by the following model:

$$y_i = \frac{\sin(2.5|x_i|)}{2.5|x_i|} + \epsilon_i, i = 1, \ldots, 20$$  \hspace{1cm} (20)
where \( x_i \) are independently and uniformly generated from \([-\pi, \pi]\) and \( \epsilon_i \) 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 (with respect to posterior) energy 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 size \( 5 \cdot 10^5 \) generated by HMC. For each algorithm we performed 10 runs with initial state generated from the prior.

The results are reported in table 6. SSD is clearly superior for a sample size 1000. For a sample size 10000 SSD has a lower energy estimation error, while the estimation error for the output prediction is comparable. It should be noted, however, that, since the “true” values were estimated using HMC, this may produce a slight bias in favor of HMC. Sample sizes larger than 10000 were not considered, because the mean squared distance between the true function and the Bayesian prediction was \( 7.5 \cdot 10^{-3} \), which is an order of magnitude larger than the estimation error for the sample size 1000.

4.2.5 Mixture Markov Chains

It is possible to combine the superior performance of SSD with the asymptotic unbiasedness of HMC. To this end we may use a mixture Markov chain with mixing probabilities \( p_t \). Specifically, the transition probabilities are taken to be a convex combination of the transition probabilities of SSD and HMC,
Table 6: Average (over 10 runs) estimation error of mean energy and mean output prediction for samples of size 1000 and 10000.

<table>
<thead>
<tr>
<th>n</th>
<th>Energy SSD</th>
<th>Energy HMC</th>
<th>Output prediction SSD</th>
<th>Output prediction HMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>1.66</td>
<td>4.40</td>
<td>2.3 \cdot 10^{-4}</td>
<td>7.0 \cdot 10^{-4}</td>
</tr>
<tr>
<td>10000</td>
<td>0.28</td>
<td>0.45</td>
<td>7.5 \cdot 10^{-5}</td>
<td>6.3 \cdot 10^{-5}</td>
</tr>
</tbody>
</table>

with the weights $p_t$ and $1 - p_t$ correspondingly. In practice this means that the SSD transition is made with probability $p_t$ and the HMC transition - with probability $1 - p_t$. The sequence $\{p_t\}$ should be decreasing to zero with $p_t=1$. The resulting Markov chain behaves similarly to SSD for a low $t$, but as $t$ grows its behavior approaches that of HMC, hence insuring asymptotic unbiasedness.

5 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, an 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 use of data smoothing techniques, automatic discretiza-
tion step selection, different annealing schedules, alternative discretization schemes etc.

References


