Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget

Anoop Korattikara AKORATTI@UCI.EDU
School of Information & Computer Sciences, University of California, Irvine, CA 92617, USA

Yutian Chen YUTIAN.CHEN@ENG.CAM.EDU
Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK

Max Welling WELLING@ICS.UCI.EDU
Informatics Institute, University of Amsterdam, Science Park 904 1098 XH, Amsterdam, Netherlands

Abstract

Can we make Bayesian posterior MCMC sampling more efficient when faced with very large datasets? We argue that computing the likelihood for N datapoints in the Metropolis-Hastings (MH) test to reach a single binary decision is computationally inefficient. We introduce an approximate MH rule based on a sequential hypothesis test that allows us to accept or reject samples with high confidence using only a fraction of the data required for the exact MH rule. While this method introduces an asymptotic bias, we show that this bias can be controlled and is more than offset by a decrease in variance due to our ability to draw more samples per unit of time.

1. Introduction

Markov chain Monte Carlo (MCMC) sampling has been the main workhorse of Bayesian computation since the 1990s. A canonical MCMC algorithm proposes samples from a distribution q and then accepts or rejects these proposals with a certain probability given by the Metropolis-Hastings (MH) formula (Metropolis et al., 1953; Hastings, 1970). For each proposed sample, the MH rule needs to examine the likelihood of all data-items. When the number of data-cases is large this is an awful lot of computation for one bit of information, namely whether to accept or reject a proposal.

In today's Big Data world, we need to rethink our Bayesian inference algorithms. Standard MCMC methods do not meet the Big Data challenge for the reason described above. Researchers have made some progress in terms of making MCMC more efficient, mostly by focusing on parallelization. Very few question the algorithm itself: is the standard MCMC paradigm really optimally efficient in achieving its goals? We claim it is not.

Any method that includes computation as an essential ingredient should acknowledge that there is a finite amount of time, T, to finish a calculation. An efficient MCMC algorithm should therefore decrease the "error" (properly defined) maximally in the given time T. For MCMC algorithms, there are two contributions to this error: bias and variance. Bias occurs because the chain needs to burn in during which it is sampling from the wrong distribution. Bias usually decreases fast, as evidenced by the fact that practitioners are willing to wait until the bias has (almost) completely vanished after which they discard these "burn-in samples". The second cause of error is sampling variance, which occurs because of the random nature of the sampling process. The retained samples after burn-in will reduce the variance as $O(1/T)$.

However, given a finite amount of computational time, it is not at all clear whether the strategy of retaining few unbiased samples and accepting an error dominated by variance is optimal. Perhaps, by decreasing the bias more slowly we could sample faster and thus reduce variance faster? In this paper we illustrate this effect by cutting the computational budget of the MH accept/reject step. To achieve that, we conduct sequential hypothesis tests to decide whether to accept or reject a given sample and find that the majority of these decisions can be made based on a small fraction of the data with high confidence. A related method was used in Singh et al. (2012), where the factors of a graphical model are sub-sampled to compute fixed-width confidence intervals for the log-likelihood in the MH test.

Our "philosophy" runs deeper than the algorithm proposed here. We advocate MCMC algorithms with a "bias-knob", allowing one to dial down the bias at a rate that optimally

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

balances error due to bias and variance. We only know of one algorithm that would also adhere to this strategy: stochastic gradient Langevin dynamics (Welling & Teh, 2011) and its successor stochastic gradient Fisher scoring (Ahn et al., 2012). In their case the bias-knob was the stepsize. These algorithms do not have an MH step which resulted in occasional samples with extremely low probability. We show that our approximate MH step largely resolves this, still avoiding $O(N)$ computations per iteration.

In the next section we introduce the MH algorithm and discuss its drawbacks. Then in Section 3, we introduce the idea of approximate MCMC methods and the bias variance trade-off involved. We develop approximate MH tests for Bayesian posterior sampling in Section 4 and present a theoretical analysis in Section 5. Finally, we show our experimental results in Section 6 and conclude in Section 7.

2. The Metropolis-Hastings algorithm

MCMC methods generate samples from a distribution $\mathcal{S}_0(\theta)$ by simulating a Markov chain designed to have stationary distribution $\mathcal{S}_0(\theta)$. A Markov chain with a given stationary distribution can be constructed using the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970), which uses the following rule for transitioning from the current state θ_t to the next state θ_{t+1}:

1. Draw a candidate state θ' from a proposal distribution $q(\theta'|\theta_t)$
2. Compute the acceptance probability:
$$P_a = \min\left[1, \frac{\mathcal{S}_0(\theta')q(\theta_t|\theta')}{\mathcal{S}_0(\theta_t)q(\theta'|\theta_t)}\right] \qquad (1)$$
3. Draw $u \sim$ Uniform$[0, 1]$. If $u < P_a$ set $\theta_{t+1} \leftarrow \theta'$, otherwise set $\theta_{t+1} \leftarrow \theta_t$.

Following this transition rule ensures that the stationary distribution of the Markov chain is $\mathcal{S}_0(\theta)$. The samples from the Markov chain are usually used to estimate the expectation of a function $f(\theta)$ with respect to $\mathcal{S}_0(\theta)$. To do this we collect T samples and approximate the expectation $I = \langle f\rangle_{\mathcal{S}_0}$ as $\hat{I} = \frac{1}{T}\sum_{t=1}^{T} f(\theta_t)$. Since the stationary distribution of the Markov chain is $\mathcal{S}_0$, $\hat{I}$ is an unbiased estimator of I (if we ignore burn-in).

The variance of $\hat{I}$ is $V = \mathbb{E}[(\langle f\rangle_{\mathcal{S}_0} - \frac{1}{T}\sum_{t=1}^{T} f(\theta_t))^2]$, where the expectation is over multiple simulations of the Markov chain. It is well known that $V \approx \sigma^2_{f,\mathcal{S}_0}\tau/T$, where $\sigma^2_{f,\mathcal{S}_0}$ is the variance of f with respect to S_0 and τ is the integrated auto-correlation time, which is a measure of the interval between independent samples (Gamerman & Lopes, 2006). Usually, it is quite difficult to design a chain that mixes fast and therefore, the auto-correlation time will be quite high. Also, for many important problems, evaluating $\mathcal{S}_0(\theta)$ to compute the acceptance probability P_a in every step is so expensive that we can collect only a very small number of samples (T) in a realistic amount of computational time. Thus the variance of $\hat{I}$ can be prohibitively high, even though it is unbiased.

3. Approximate MCMC and the Bias-Variance Tradeoff

Ironically, the reason MCMC methods are so slow is that they are designed to be unbiased. If we were to allow a small bias in the stationary distribution, it is possible to design a Markov chain that can be simulated cheaply (Welling & Teh, 2011; Ahn et al., 2012). That is, to estimate $I = \langle f\rangle_{\mathcal{S}_0}$, we can use a Markov chain with stationary distribution $\mathcal{S}_\epsilon$ where ϵ is a parameter that can be used to control the bias in the algorithm. Then I can be estimated as $\hat{I} = \frac{1}{T}\sum_{t=1}^{T} f(\theta_t)$, computed using samples from $\mathcal{S}_\epsilon$ instead of $\mathcal{S}_0$.

As $\epsilon \to 0$, $\mathcal{S}_\epsilon$ approaches $\mathcal{S}_0$ (the distribution of interest) but it becomes expensive to simulate the Markov chain. Therefore, the bias in $\hat{I}$ is low, but the variance is high because we can collect only a small number of samples in a given amount of computational time. As ϵ moves away from 0, it becomes cheap to simulate the Markov chain but the difference between $\mathcal{S}_\epsilon$ and $\mathcal{S}_0$ grows. Therefore, $\hat{I}$ will have higher bias, but lower variance because we can collect a larger number of samples in the same amount of computational time. This is a classical bias-variance trade-off and can be studied using the risk of the estimator.

The risk can be defined as the mean squared error in $\hat{I}$, i.e. $R = \mathbb{E}[(I - \hat{I})^2]$, where the expectation is taken over multiple simulations of the Markov chain. It is easy to show that the risk can be decomposed as $R = B^2 + V$, where B is the bias and V is the variance. If we ignore burn-in, it can be shown that $B = \langle f\rangle_{\mathcal{S}_\epsilon} - \langle f\rangle_{\mathcal{S}_0}$ and $V = \mathbb{E}[(\langle f\rangle_{\mathcal{S}_\epsilon} - \frac{1}{T}f(\theta_t))^2] \approx \sigma^2_{f,\mathcal{S}_\epsilon}\tau/T$.

The optimal setting of ϵ that minimizes the risk depends on the amount of computational time available. If we have an infinite amount of computational time, we should set ϵ to 0. Then there is no bias, and the variance can be brought down to 0 by drawing an infinite number of samples. This is the traditional MCMC setting. However, given a finite amount of computational time, this setting may not be optimal. It might be better to tolerate a small amount of bias in the stationary distribution if it allows us to reduce the variance quickly, either by making it cheaper to collect a large number of samples or by mixing faster.

It is interesting to note that two recently proposed algorithms follow this paradigm: Stochastic Gradient Langevin

Dynamics (SGLD) (Welling & Teh, 2011) and Stochastic Gradient Fisher Scoring (SGFS) (Ahn et al., 2012). These algorithms are biased because they omit the required Metropolis-Hastings tests. However, in both cases, a knob ϵ (the step-size of the proposal distribution) is available to control the bias. As $\epsilon \rightarrow 0$, the acceptance probability $P_a \rightarrow 1$ and the bias from not conducting MH tests disappears. However, when $\epsilon \rightarrow 0$ the chain mixes very slowly and the variance increases because the auto-correlation time $\tau \rightarrow \infty$. As ϵ is increased from 0, the auto-correlation, and therefore the variance, reduces. But, at the same time, the acceptance probability reduces and the bias from not conducting MH tests increases as well.

In the next section, we will develop another class of approximate MCMC algorithms for the case where the target $\mathcal{S}_0$ is a Bayesian posterior distribution given a very large dataset. We achieve this by developing an approximate Metropolis-Hastings test, equipped with a knob for controlling the bias. Moreover, our algorithm has the advantage that it can be used with any proposal distribution. For example, our method allows approximate MCMC methods to be applied to problems where it is impossible to compute gradients (which is necessary to apply SGLD/SGFS). Or, we can even combine our method with SGLD/SGFS, to obtain the best of both worlds.

4. Approximate Metropolis-Hastings Test for Bayesian Posterior Sampling

An important method in the toolbox of Bayesian inference is posterior sampling. Given a dataset of N independent observations $X_N = \{x_1, \dots, x_N\}$, which we model using a distribution $p(x;\theta)$ parameterized by θ, defined on a space Θ with measure Ω, and a prior distribution $\rho(\theta)$, the task is to sample from the posterior distribution $\mathcal{S}_0(\theta) \propto \rho(\theta) \prod_{i=1}^N p(x_i;\theta)$.

If the dataset has a billion datapoints, it becomes very painful to compute $\mathcal{S}_0(.)$ in the MH test, which has to be done for each posterior sample we generate. Spending $O(N)$ computation to get just 1 bit of information, i.e. whether to accept or reject a sample, is likely not the best use of computational resources.

But, if we try to develop accept/reject tests that satisfy detailed balance exactly with respect to the posterior distribution using only sub-samples of data, we will quickly see the no free lunch theorem kicking in. For example, the pseudo marginal MCMC method (Andrieu & Roberts, 2009) and the method developed by Lin et al. (2000) provide a way to conduct exact accept/reject tests using unbiased estimators of the likelihood. However, unbiased estimators of the likelihood that can be computed from mini-batches of data, such as the Poisson estimator (Fearnhead et al., 2008) or the Kennedy-Bhanot estimator (Lin et al., 2000) have very high variance for large datasets. Because of this, once we get a very high estimate of the likelihood, almost all proposed moves are rejected and the algorithm gets stuck.

Thus, we should be willing to tolerate some error in the stationary distribution if we want faster accept/reject tests. If we can offset this small bias by drawing a large number of samples cheaply and reducing the variance faster, we can establish a potentially large reduction in the risk.

We will now show how to develop such approximate tests by reformulating the MH test as a statistical decision problem. It is easy to see that the original MH test (Eqn. 1) is equivalent to the following procedure: Draw $u \sim \text{Uniform}[0,1]$ and accept the proposal θ' if the average difference μ in the log-likelihoods of θ' and θ_t is greater than a threshold μ_0, i.e. compute

$$\mu_0 = \frac{1}{N} \log \left[u \frac{\rho(\theta_t) q(\theta'|\theta_t)}{\rho(\theta') q(\theta_t|\theta')} \right], \text{ and} \tag{2}$$

$$\mu = \frac{1}{N} \sum_{i=1}^N l_i \text{ where } l_i = \log p(x_i;\theta') - \log p(x_i;\theta_t) \tag{3}$$

Then if $\mu > \mu_0$, accept the proposal and set $\theta_{t+1} \leftarrow \theta'$. If $\mu \leq \mu_0$, reject the proposal and set $\theta_{t+1} \leftarrow \theta_t$. This reformulation of the MH test makes it very easy to frame it as a statistical hypothesis test. Given μ_0 and a random sample $\{l_{i_1}, \dots, l_{i_n}\}$ drawn without replacement from the population $\{l_1, \dots, l_N\}$, can we decide whether the population mean μ is greater than or less than the threshold μ_0? The answer to this depends on the precision in the random sample. If the difference between the sample mean $\bar{l}$ and μ_0 is significantly greater than the standard deviation s of $\bar{l}$, we can make the decision to accept or reject the proposal confidently. If not, we should draw more data to increase the precision of $\bar{l}$ (reduce s) until we have enough evidence to make a decision.

More formally, we test the hypotheses $H_1 : \mu > \mu_0$ vs $H_2 : \mu < \mu_0$. To do this, we proceed as follows: We compute the sample mean $\bar{l}$ and the sample standard deviation $s_l = \sqrt{(\overline{l^2} - (\bar{l})^2)\frac{n}{n-1}}$. Then the standard deviation of $\bar{l}$ can be estimated as:

$$s = \frac{s_l}{\sqrt{n}} \sqrt{1 - \frac{n-1}{N-1}} \tag{4}$$

where $\sqrt{1 - \frac{n-1}{N-1}}$, the finite population correction term, is applied because we are drawing the subsample without replacement from a finite-sized population. Then, we compute the test statistic:

$$t = \frac{\bar{l} - \mu_0}{s} \tag{5}$$

Algorithm 1 Approximate MH test

```
Require: $\theta_t, \theta', \epsilon, \mu_0, X_N, m$
Ensure: $accept$
Initialize estimated means $\bar{l} \leftarrow 0$ and $\overline{l^2} \leftarrow 0$
Initialize $n \leftarrow 0$, $done \leftarrow$ false
Draw $u \sim$ Uniform[0,1]
while not $done$ do
    Draw mini-batch $\mathcal{X}$ of size $\min(m, N - n)$ without replacement from $X_N$ and set $X_N \leftarrow X_N \setminus \mathcal{X}$
    Update $\bar{l}$ and $\overline{l^2}$ using $\mathcal{X}$, and $n \leftarrow n + |\mathcal{X}|$
    Estimate std $s$ using Eqn. 4
    Compute $\delta \leftarrow 1 - \phi_{n-1}\left(\left|\frac{\bar{l} - \mu_0}{s}\right|\right)$
    if $\delta < \epsilon$ then
        $accept \leftarrow$ true if $\bar{l} > \mu_0$ and false otherwise
        $done \leftarrow$ true
    end if
end while
```

If n is large enough for the central limit theorem (CLT) to hold, the test statistic t follows a standard Student-t distribution with $n - 1$ degrees of freedom, when $\mu = \mu_0$ (see Fig. 7 in supplementary for an empirical verification). Then, we compute $\delta = 1 - \phi_{n-1}(|t|)$ where $\phi_{n-1}(.)$ is the cdf of the standard Student-t distribution with $n-1$ degrees of freedom. If $\delta < \epsilon$ (a fixed threshold) we can confidently say that μ is significantly different from μ_0. In this case, if $\bar{l} > \mu_0$, we decide $\mu > \mu_0$, otherwise we decide $\mu < \mu_0$. If $\delta \geq \epsilon$, we do not have enough evidence to make a decision. In this case, we draw more data to reduce the uncertainty, s, in the sample mean $\bar{l}$. We keep drawing more data until we have the required confidence (i.e. until $\delta < \epsilon$). Note, that this procedure will terminate because when we have used all the available data, i.e. $n = N$, the standard deviation s is 0, the sample mean $\bar{l} = \mu$ and $\delta = 0 < \epsilon$. So, we will make the same decision as the original MH test would make. Pseudo-code for our test is shown in Algorithm 1. Here, we start with a mini-batch of size m for the first test and increase it by m datapoints when required.

The advantage of our method is that often we can make confident decisions with $n < N$ datapoints and save on computation, although we introduce a small bias in the stationary distribution. But, we can use the computational time we save to draw more samples and reduce the variance. The bias-variance trade-off can be controlled by adjusting the knob ϵ. When ϵ is high, we make decisions without sufficient evidence and introduce a high bias. As $\epsilon \to 0$, we make more accurate decisions but are forced to examine more data which results in high variance.

Our algorithm will behave erratically if the CLT does not hold, e.g. with very sparse datasets or datasets with extreme outliers. The CLT assumption can be easily tested empirically before running the algorithm to avoid such pathological situations. The sequential hypothesis testing method can also be used to speed-up Gibbs sampling in densely connected Markov Random Fields. We explore this idea briefly in Section F of the supplementary.

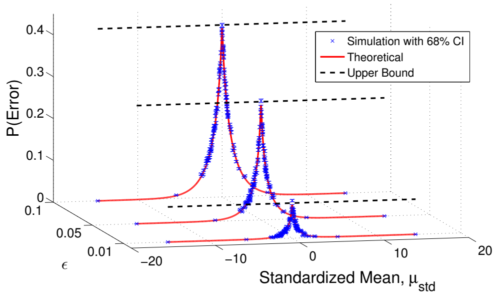


Figure 1. Error $\mathcal{E}$ estimated using simulation (blue cross with 1 σ error bar) and dynamic programming (red line). An upper bound (black dashed line) is also shown.

5. Error Analysis and Test Design

In 5.1, we study the relation between the parameter ϵ, the error $\mathcal{E}$ of the complete sequential test, the error Δ in the acceptance probability and the error in the stationary distribution. In 5.2, we describe how to design an optimal test that minimizes data usage given a bound on the error.

5.1. Error Analysis and Estimation

The parameter ϵ is an upper-bound on the error of a single test and not the error of the complete sequential test. To compute this error, we assume a) n is large enough that the t statistics can be approximated with z statistics, and b) the joint distribution of the $\bar{l}$'s corresponding to different mini-batches used in the test is multivariate normal. Under these assumptions, we can show that the test statistic at different stages of the sequential test follows a Gaussian Random Walk process. This allows us to compute the error of the sequential test $\mathcal{E}(\mu_{\mathrm{std}}, m, \epsilon)$, and the expected proportion of the data required to reach a decision $\bar{\pi}(\mu_{\mathrm{std}}, m, \epsilon)$, using an efficient dynamic programming algorithm. Note that $\mathcal{E}$ and $\bar{\pi}$ depend on θ, θ' and u only through the 'standardized mean' defined as $\mu_{\mathrm{std}}(u, \theta, \theta') \overset{\text{def}}{=} \frac{(\mu(\theta, \theta') - \mu_0(\theta, \theta', u))\sqrt{N-1}}{\sigma_l(\theta, \theta')}$ where σ_l is the true standard deviation of the l_i's. See Section A of the supplementary for a detailed derivation and an empirical validation of the assumptions.

Fig. 1 shows the theoretical and actual error of 1000 sequential tests for the logistic regression model described in Section 6.1. The error $\mathcal{E}(\mu_{\mathrm{std}}, m, \epsilon)$ is highest in the worst case when $\mu = \mu_0$. Therefore, $\mathcal{E}(0, m, \epsilon)$ is an upper-bound on $\mathcal{E}$. Since the error decreases sharply as μ moves away from μ_0, we can get a more useful estimate of $\mathcal{E}$ if we have some knowledge about the distribution of μ_{std}'s that will be encountered during the Markov chain simulation.

Now, let $P_{a,\epsilon}(\theta, \theta')$ be the actual acceptance probability of our algorithm and let $\Delta(\theta, \theta') \overset{\text{def}}{=} P_{a,\epsilon}(\theta, \theta') - P_a(\theta, \theta')$ be the error in $P_{a,\epsilon}$. In Section B of the supplementary, we show that for any (θ, θ'):

$$\Delta = \int_{P_a}^{1} \mathcal{E}(\mu_{\text{std}}(u))\mathrm{d}u - \int_{0}^{P_a} \mathcal{E}(\mu_{\text{std}}(u))\mathrm{d}u \qquad (6)$$

Thus, the errors corresponding to different u's partly cancel each other. As a result, although $|\Delta(\theta, \theta')|$ is upper-bounded by the worst-case error $\mathcal{E}(0, m, \epsilon)$ of the sequential test, the actual error is usually much smaller. For any given (θ, θ'), Δ can be computed easily using 1-dimensional quadrature.

Finally, we show that the error in the stationary distribution is bounded linearly by $\Delta_{\text{max}} = \sup_{\theta,\theta'} |\Delta(\theta, \theta')|$. As noted above, $\Delta_{\text{max}} \le \mathcal{E}(0, m, \epsilon)$ but is usually much smaller. Let $d_v(P, Q)$ denote the total variation distance[1] between two distributions, P and Q. If the transition kernel $\mathcal{T}_0$ of the exact Markov chain satisfies the contraction condition $d_v(P\mathcal{T}_0, \mathcal{S}_0) \le \eta d_v(P, \mathcal{S}_0)$ for all probability distributions P with a constant $\eta \in [0, 1)$, we can prove (see supplementary Section C) the following upper bound on the error in the stationary distribution:

Theorem 1. *The distance between the posterior distribution $\mathcal{S}_0$ and the stationary distribution of our approximate Markov chain $\mathcal{S}_\epsilon$ is upper bounded as:*

$$d_v(\mathcal{S}_0, \mathcal{S}_\epsilon) \le \frac{\Delta_{max}}{1 - \eta}$$

5.2. Optimal Sequential Test Design

We now briefly describe how to choose the parameters of the algorithm: ϵ, the error of a single test and m, the mini-batch size. A very simple strategy we recommend is to choose $m \approx 500$ so that the Central Limit Theorem holds and keep ϵ as small as possible while maintaining a low average data usage. This rule works well in practice and is used in Experiments 6.1 - 6.4.

The more discerning practitioner can design an optimal test that minimizes the data used while keeping the error below a given tolerance. Ideally, we want to do this based on a tolerance on the error in the stationary distribution $\mathcal{S}_\epsilon$. Unfortunately, this error depends on the contraction parameter, η, of the exact transition kernel, which is difficult to compute. A more practical choice is a bound on the error Δ in the acceptance probability, since the error in $\mathcal{S}_\epsilon$ increases linearly with Δ. Since Δ is a function of (θ, θ'), we can try to control the average value of Δ over the empirical distribution of (θ, θ') that would be encountered while simulating the Markov chain. Given a tolerance Δ^* on this average error, we can find the optimal m and ϵ by solving the following optimization problem (e.g. using grid search) to minimize the average data usage :

$$\begin{aligned} &\min_{m,\epsilon} \mathbb{E}_{\theta,\theta'} \left[\mathbb{E}_u \bar{\pi}(\mu_{\text{std}}(u, \theta, \theta'), m, \epsilon)\right] \\ &\text{s.t. } \mathbb{E}_{\theta,\theta'} |\Delta(m, \epsilon, \theta, \theta')| \le \Delta^* \end{aligned} \qquad (7)$$

In the above equation, we estimate the average data usage, $\mathbb{E}_u[\bar{\pi}]$, and the error in the acceptance probability, Δ, using dynamic programming with one dimensional numerical quadrature on u. The empirical distribution for computing the expectation with respect to (θ, θ') can be obtained using a trial run of the Markov chain. Without a trial run the best we can do is to control the worst case error $\mathcal{E}(0, m, \epsilon)$ (which is also an upper-bound on Δ) in each sequential test by solving the following minimization problem:

$$\min_{m,\epsilon} \bar{\pi}(0, m, \epsilon) \;\text{ s.t. }\; \mathcal{E}(0, m, \epsilon) \le \Delta^* \qquad (8)$$

But this leads to a very conservative design as the worst case error is usually much higher than the average case error. We illustrate the sequential design in Experiment 6.5. More details and a generalization of this method is given in supplementary Section D.

6. Experiments

6.1. Random Walk - Logistic Regression

We first test our method using a random walk proposal $q(\theta'|\theta_t) = \mathcal{N}(\theta_t, \sigma_{RW}^2)$. Although the random walk proposal is not efficient, it is very useful for illustrating our algorithm because the proposal does not contain any information about the target distribution, unlike Langevin or Hamiltonian methods. So, the responsibility of converging to the correct distribution lies solely with the MH test. Also since q is symmetric, it does not appear in the MH test and we can use $\mu_0 = \frac{1}{N} \log\left[u\rho(\theta_t)/\rho(\theta')\right]$.

The target distribution in this experiment was the posterior for a logistic regression model trained on the MNIST dataset for classifying digits 7 vs 9. The dataset consisted of 12214 datapoints and we reduced the dimensionality from 784 to 50 using PCA. We chose a zero mean spherical Gaussian prior with precision = 10, and set $\sigma_{RW} = 0.01$.

In Fig. 2, we show how the logarithm of the risk in estimating the predictive mean, decreases as a function of wall

[1] The total variation distance between two distributions P and Q, that are absolutely continuous w.r.t. measure Ω, is defined as $d_v(P, Q) \overset{\text{def}}{=} \frac{1}{2}\int_{\theta\in\Theta} |f_P(\theta) - f_Q(\theta)| d\Omega(\theta)$ where f_P and f_Q are their respective densities (or Radon-Nikodym derivatives to be more precise).

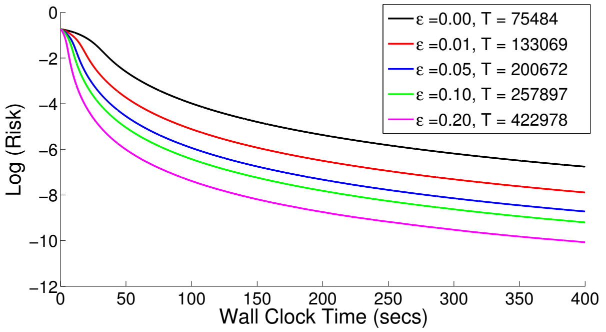


Figure 2. Logistic Regression: Risk in predictive mean.

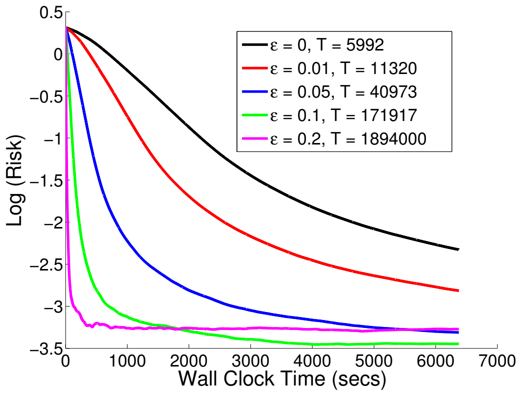


Figure 3. ICA: Risk in mean of Amari distance

clock time. The predictive mean of a test point x^* is defined as $\mathbb{E}_{p(\theta|X_N)}[p(x^*|\theta)]$. To calculate the risk, we first estimate the true predictive mean using a long run of Hybrid Monte Carlo. Then, we compute multiple estimates of the predictive mean from our approximate algorithm and obtain the risk as the mean squared error in these estimates. We plot the average risk of 2037 datapoints in the test set. Since the risk $R = B^2 + V = B^2 + \frac{\sigma^2 f}{T}$, we expect it to decrease as a function of time until the bias dominates the variance. The figure shows that even after collecting a lot of samples, the risk is still dominated by the variance and the minimum risk is obtained with $\epsilon > 0$.

6.2. Independent Component Analysis

Next, we use our algorithm to sample from the posterior distribution of the unmixing matrix in Independent Component Analysis (ICA) (Hyvärinen & Oja, 2000). When using prewhitened data, the unmixing matrix $W \in \mathbb{R}^{D\times D}$ is constrained to lie on the Stiefel manifold of orthonormal matrices. We choose a prior that is uniform over the manifold and zero elsewhere. We model the data as $p(x|W) = |\det(W)| \prod_{j=1}^{D} \left[4\cosh^2(\frac{1}{2}w_j^T x)\right]^{-1}$ where w_j are the rows of W. Since the prior is zero outside the manifold, the same is true for the posterior. Therefore we use a random walk on the Stiefel manifold as a proposal distribution (Ouyang, 2008). Since this is a symmetric proposal distribution, it does not appear in the MH test and we can use $\mu_0 = \frac{1}{N}\log[u]$.

To perform a large scale experiment, we created a synthetic dataset by mixing 1.95 million samples of 4 sources: (a) a Classical music recording (b) street / traffic noise (c) & (d) 2 independent Gaussian sources. To measure the correctness of the sampler, we measure the risk in estimating $I = \mathbb{E}_{p(W|X)}[d_A(W, W_0)]$ where the test function d_A is the Amari distance (Amari et al., 1996) and W_0 is the true unmixing matrix. We computed the ground truth using a long run (T = 100K samples) of the exact MH algorithm. Then we ran each algorithm 10 times, each time for ≈ 6400 secs. We calculated the risk by averaging the squared error in the estimate from each Markov chain, over the 10 chains. This is shown in Fig. 3. Note that even after 6400 secs the variance dominates the bias, as evidenced by the still decreasing risk, except for the most biased algorithm with $\epsilon = 0.2$. Also, the lowest risk at 6400 secs is obtained with $\epsilon = 0.1$ and not the exact MH algorithm ($\epsilon = 0$). But we expect the exact algorithm to outperform all the approximate algorithms if we were to run for an infinite time.

6.3. Variable selection in Logistic Regression

Now, we apply our MH test to variable selection in a logistic regression model using the reversible jump MCMC algorithm of Green (1995). We use a model that is similar to the Bayesian LASSO model for linear regression described in Chen et al. (2011). Specifically, given D input features, our parameter $\theta = \{\beta, \gamma\}$ where β is a vector of D regression coefficients and γ is a D dimensional binary vector that indicates whether a particular feature is included in the model or not. The prior we choose for β is $p(\beta_j|\gamma,\nu) = \frac{1}{2\nu}\exp\left\{-\frac{|\beta_j|}{\nu}\right\}$ if $\gamma_j = 1$. If $\gamma_j = 0$, β_j does not appear in the model. Here ν is a shrinkage parameter that pushes β_j towards 0, and we choose a prior $p(\nu) \propto 1/\nu$. We also place a right truncated Poisson prior $p(\gamma|\lambda) \propto \dfrac{\lambda^k}{\binom{D}{k}k!}$ on γ to control the size of the model, $k = \sum_{j=1}^{D}\gamma_j$ We set $\lambda = 10^{-10}$ in this experiment.

Denoting the likelihood of the data by $l_N(\beta,\gamma)$, the posterior distribution after integrating out ν is $p(\beta,\gamma|X_N, y_N, \lambda) \propto l_N(\beta,\gamma)\|\beta\|_1^{-k}\lambda^k B(k, D-k+1)$ where $B(.,.)$ is the beta function. Instead of integrating out λ, we use it as a parameter to control the size of the model. We use the same proposal distribution as in (Chen et al., 2011) which is a mixture of 3 type of moves that are picked randomly in each iteration: an update move, a birth move and a death move. A detailed description is given in

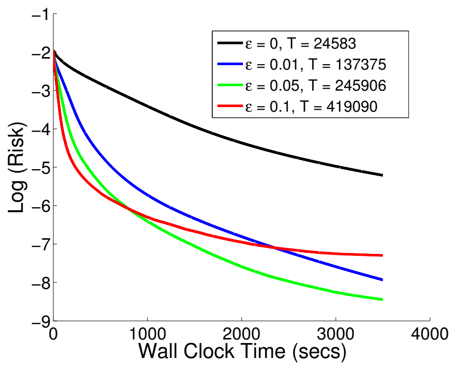


Figure 4. RJMCMC: Risk in predictive mean

Supplementary Section E.

We applied this to the MiniBooNE dataset from the UCI machine learning repository(Bache & Lichman, 2013). Here the task is to classify electron neutrinos (signal) from muon neutrinos (background). There are 130,065 datapoints (28% in +ve class) with 50 features to which we add a constant feature of 1's. We randomly split the data into a training (80%) and testing (20%) set. To compute ground truth, we collected T=400K samples using the exact reversible jump algorithm ($\epsilon = 0$). Then, we ran the approximate MH algorithm with different values of ϵ for around 3500 seconds. We plot the risk in predictive mean of test data (estimated from 10 Markov chains) in Fig. 4. Again we see that the lowest risk is obtained with $\epsilon > 0$.

The acceptance rates for the birth/death moves starts off at $\approx 20\%$ but dies down to $\approx 2\%$ once a good model is found. The acceptance rate for update moves is kept at $\approx 50\%$. The model also suffers from local minima. For the plot in Fig. 4, we started with only one variable and we ended up learning models with around 12 features, giving a classification error $\approx 15\%$. But, if we initialize the sampler with all features included and initialize β to the MAP value, we learn models with around 45 features, but with a lower classification error $\approx 10\%$. Both the exact reversible jump algorithm and our approximate version suffer from this problem. We should bear this in mind when interpreting "ground truth". However, we have observed that when initialized with the same values, we obtain similar results with the approximate algorithm and the exact algorithm (see e.g. Fig. 13 in supplementary).

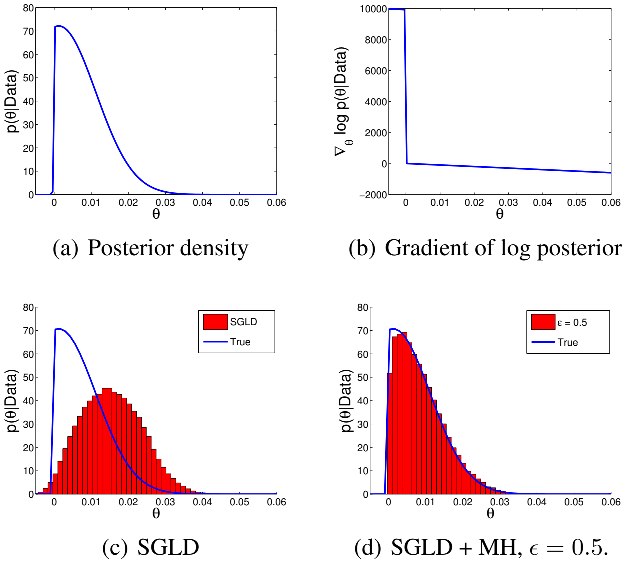
6.4. Stochastic Gradient Langevin Dynamics

Finally, we apply our method to Stochastic Gradient Langevin Dynamics(Welling & Teh, 2011). In each iteration, we randomly draw a mini-batch $\mathcal{X}_n$ of size n, and propose $\theta' \sim q(.|\theta, \mathcal{X}_n) =$

$$\mathcal{N}\left(\theta + \frac{\alpha}{2}\nabla_\theta\left\{\frac{N}{n}\sum_{x\in\mathcal{X}_n}\log p(x|\theta) + \log\rho(\theta)\right\}, \alpha\right).$$

The proposed state θ' is always accepted (without conducting any MH test). Since the acceptance probability approaches 1 as we reduce α, the bias from not conducting the MH test can be kept under control by using $\alpha \approx 0$. However, we have to use a reasonably large α to keep the mixing rate high. This can be problematic for some distributions, because SGLD relies solely on gradients of the log density and it can be easily thrown off track by large gradients in low density regions, unless $\alpha \approx 0$.

As an example, consider an L1-regularized linear regression model. Given a dataset $\{x_i, y_i\}_{i=1}^N$ where x_i are predictors and y_i are targets, we use a Gaussian error model $p(y|x,\theta) \propto \exp\left\{-\frac{\lambda}{2}(y - \theta^T x)^2\right\}$ and choose a Laplacian prior for the parameters $p(\theta) \propto \exp(-\lambda_0\|\theta\|_1)$. For pedagogical reasons, we will restrict ourselves to a toy version of the problem where θ and x are one dimensional. We use a synthetic dataset with $N = 10000$ datapoints generated as $y_i = 0.5x_i + \xi$ where $\xi \sim \mathcal{N}(0, 1/3)$. We choose $\lambda = 3$ and $\lambda_0 = 4950$, so that the prior is not washed out by the likelihood. The posterior density and the gradient of the log posterior are shown in figures 5(a) and 5(b) respectively.



(a) Posterior density

(b) Gradient of log posterior

(c) SGLD

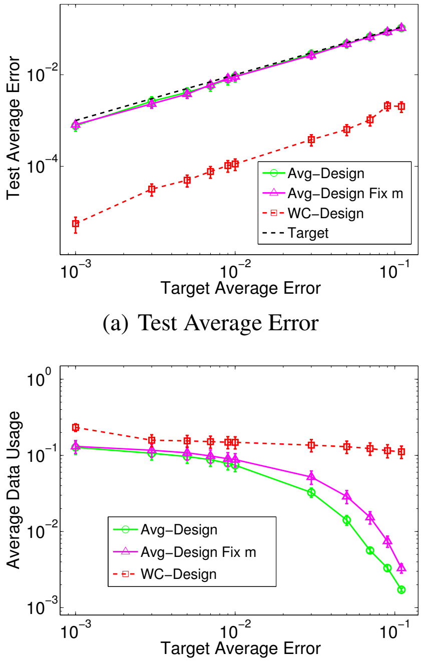
(d) SGLD + MH, $\epsilon = 0.5$.

Figure 5. Pitfalls of using uncorrected SGLD

An empirical histogram of samples obtained by running SGLD with $\alpha = 5 \times 10^{-6}$ is shown in Fig. 5(c). The effect of omitting the MH test is quite severe here. When the sampler reaches the mode of the distribution, the Langevin noise occasionally throws it into the valley to the left, where the gradient is very high. This propels the sampler far off to the right, after which it takes a long time to find its way back to the mode. However, if we had used an MH accept-reject test, most of these troublesome jumps into the valley would be rejected because the density in the valley is much lower than that at the mode.

To apply an MH test, note that the SGLD proposal $q(\theta'|\theta)$ can be considered a mixture of component kernels $q(\theta'|\theta, \mathcal{X}_n)$ corresponding to different mini-batches. The mixture kernel will satisfy detailed balance with respect to the posterior distribution if the MH test enforces detailed balance between the posterior and each of the component kernels $q(\theta'|\theta, \mathcal{X}_n)$. Thus, we can use an MH test with $\mu_0 = \frac{1}{N} \log \left[u \frac{\rho(\theta_t) q(\theta'|\theta_t, \mathcal{X}_n)}{\rho(\theta') q(\theta_t|\theta', \mathcal{X}_n)} \right]$.

The result of running SGLD (keeping $\alpha = 5 \times 10^{-6}$ as before) corrected using our approximate MH test, with $\epsilon = 0.5$, is shown in Fig. 5(d). As expected, the MH test rejects most troublesome jumps into the valley because the density in the valley is much lower than that at the mode. The stationary distribution is almost indistinguishable from the true posterior. Note that when $\epsilon = 0.5$, a decision is always made in the first step (using just $m = 500$ datapoints) without querying additional data sequentially.



(a) Test Average Error

(b) Average Data Usage

Figure 6. Test average error in P_a and data usage $\mathbb{E}_u[\bar{\pi}]$ for the ICA experiment using average design over both m and ϵ ($\bigcirc$), with fixed $m = 600$ ($\triangle$), and worst-case design ($\square$).

6.5. Optimal Design of Sequential Tests

We illustrate the advantages of the optimal test design proposed in Section 5.2 by applying it to the ICA experiment described in Section 6.2. We consider two design methods: the 'average design' (Eqn. 7) and the 'worst-case design' (Eqn. 8). For the average design, we collected 100 samples of the Markov chain to approximate the expectation of the error over (θ, θ'). We will call these samples the training set. The worst case design does not need the training set as it does not involve the distribution of (θ, θ'). We compute the optimal m and ϵ using grid search, for different values of the target training error, for both designs. We then collect a new set of 100 samples (θ, θ') and measure the average error and data usage on this test set (Fig. 6).

For the same target error on the training set, the worst-case design gives a conservative parameter setting that achieves a much smaller error on the test set. In contrast, the average design achieves a test error that is almost the same as the target error (Fig. 6(a)). Therefore, it uses much less data than the worst-case design (Fig. 6(b)).

We also analyze the performance in the case where we fix $m = 600$ and only change ϵ. This is a simple heuristic we recommended at the beginning of Section 5.2. Although this usually works well, using the optimal test design ensures the best possible performance. In this experiment, we see that when the error is large, the optimal design uses only half the data (Fig. 6(b)) used by the heuristic and is therefore twice as fast.

7. Conclusions and Future Work

We have taken a first step towards cutting the computational budget of the Metropolis-Hastings MCMC algorithm, which takes $O(N)$ likelihood evaluations to make the binary decision of accepting or rejecting a proposed sample. In our approach, we compute the probability that a new sample will be accepted based on a subset of the data. We increase the cardinality of the subset until a prescribed confidence level is reached. In the process we create a bias, which is more than compensated for by a reduction in variance due to the fact that we can draw more samples per unit time. Current MCMC procedures do not take these trade-offs into account. In this work we use a fixed decision threshold for accepting or rejecting a sample, but in theory a better algorithm can be obtained by adapting this threshold over time. An adaptive algorithm can tune bias and variance contributions in such a way that at every moment our risk (the sum of squared bias and variance) is as low as possible. We leave these extensions for future work.

Acknowledgments

We thank Alex Ihler, Daniel Gillen, Sungjin Ahn, Babak Shahbaba and the anonymous reviewers for their valuable suggestions. This material is based upon work supported by the National Science Foundation under Grant No. 1216045.

References

Ahn, S., Korattikara, A., and Welling, M. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *International Conference on Machine Learning*, 2012.

Amari, Shun-ichi, Cichocki, Andrzej, Yang, Howard Hua, et al. A new learning algorithm for blind signal separation. *Advances in neural information processing systems*, pp. 757–763, 1996.

Andrieu, Christophe and Roberts, Gareth O. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

Bache, K. and Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Brémaud, P. *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, volume 31. Springer, 1999.

Chen, Xiaohui, Jane Wang, Z, and McKeown, Martin J. A Bayesian Lasso via reversible-jump MCMC. *Signal Processing*, 91(8):1920–1932, 2011.

Fearnhead, Paul, Papaspiliopoulos, Omiros, and Roberts, Gareth O. Particle filters for partially observed diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):755–777, 2008.

Gamerman, Dani and Lopes, Hedibert F. *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*, volume 68. Chapman & Hall/CRC, 2006.

Green, Peter J. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Hastings, W Keith. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1): 97–109, 1970.

Hyvärinen, Aapo and Oja, Erkki. Independent component analysis: algorithms and applications. *Neural networks*, 13(4):411–430, 2000.

Lin, L, Liu, KF, and Sloan, J. A noisy Monte Carlo algorithm. *Physical Review D*, 61(7):074505, 2000.

Metropolis, Nicholas, Rosenbluth, Arianna W, Rosenbluth, Marshall N, Teller, Augusta H, and Teller, Edward. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21:1087, 1953.

O'Brien, Peter C and Fleming, Thomas R. A multiple testing procedure for clinical trials. *Biometrics*, pp. 549–556, 1979.

Ouyang, Zhi. *Bayesian Additive Regression Kernels*. PhD thesis, Duke University, 2008.

Pocock, Stuart J. Group sequential methods in the design and analysis of clinical trials. *Biometrika*, 64(2):191–199, 1977.

Singh, Sameer, Wick, Michael, and McCallum, Andrew. Monte Carlo MCMC: efficient inference by approximate sampling. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1104–1113. Association for Computational Linguistics, 2012.

Wang, Samuel K and Tsiatis, Anastasios A. Approximately optimal one-parameter boundaries for group sequential trials. *Biometrics*, pp. 193–199, 1987.

Welling, M. and Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pp. 681–688, 2011.