Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget

Anoop Korattikara* Yutian Chen† Max Welling‡*

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 twice in order to reach a single binary decision is computationally inefficient. We introduce an approximate Metropolis-Hastings rule based on a sequential hypothesis test which 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 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.

Key Words: MCMC, Metropolis-Hastings, Sequential testing, Big Data

1. Introduction

Markov chain Monte Carlo (MCMC) sampling has been the main workhorse of Bayesian computation ever since its development in the 1950's. 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]. The MH rule needs to examine the likelihood of all the data-items twice, once for the current state and a second time for the proposed state. When the number of data-cases is large this is an awful lot of computation for one bit of information, namely to accept or reject a proposal.

In today's Big Data world, we need to rethink our Bayesian inference algorithms or risk becoming obsolete. 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 form 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 roughly 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 thus 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. We argue that many accept/reject decisions

*Donald Bren School of Information & Computer Sciences, University of California, Irvine, USA
†Department of Engineering, University of Cambridge, Cambridge, UK
‡Institute of Informatics, University of Amsterdam, Amsterdam, The Netherlands

can be taken by examining only a fraction of the full dataset. 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 similar method was developed 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 terms appearing 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 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 and 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 issue, 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 in section 4 for Bayesian posterior sampling. Finally, we show our experimental results in section 5 and conclude in section 6.

2. The Metropolis-Hastings algorithm

MCMC methods generate samples from a distribution $\mathcal{S}_0(\theta)$ by forward 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], 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:

$$P_a = \min\left[1, \frac{\mathcal{S}_0(\theta')q(\theta_t|\theta')}{\mathcal{S}_0(\theta_t)q(\theta'|\theta_t)}\right] \tag{1}$$

3. Draw $u \sim \text{Uniform}[0,1]$. 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.

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 and 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 a lot of 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 given 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 Trade-off

Ironically, the reason why MCMC methods are so slow is that they are designed to be unbiased. In fact, if we were to allow a slight bias in the stationary distribution, it is possible to design a Markov chain that can be simulated cheaply [Welling and Teh, 2011, Ahn et al., 2012]. That is, to estimate $I = \langle f \rangle_{\mathcal{S}_0}$, we can construct 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 the distribution of interest $\mathcal{S}_0$ 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.

Risk is defined as the mean squared error in $\hat{I}$, i.e. $R = \mathbb{E}[(\langle f \rangle_{\mathcal{S}_0} - \frac{1}{T}\sum_{t=1}^{T} f(\theta_t))^2]$, 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 and 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 $\epsilon \to 0$, the acceptance probability $P_a \to 1$ and the bias from not conducting MH tests disappears. However, when $\epsilon \to 0$ the chain mixes very slowly and the variance increases because the auto-correlation time $\tau \to \infty$. 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 X_N consisting of N independent observations $\{x_1, ...x_N\}$ which we model using a distribution $p(x;\theta)$ parameterized by $\theta \in \mathbb{R}^D$, and a prior distribution $\rho(\theta)$, the task is to generate samples from the posterior distribution $\mathcal{S}_0(\theta) \propto \rho(\theta)\prod_{i=1}^{N} p(x_i;\theta)$ defined on a space Θ with measure Ω.

If the dataset has a billion datapoints, it becomes excruciatingly painful to compute $\mathcal{S}_0(.)$ in the MH test, which has to be conducted for each posterior sample that 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 such accept/reject tests using only sub-samples of data, that satisfy detailed balance exactly with respect to the posterior distribution, we will quickly see the no free lunch theorem kicking in. For example, the pseudo marginal MCMC method [Andrieu and 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 or the Kennedy-Bhanot estimator [Lin et al., 2000] have incredibly 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.

Therefore, 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.

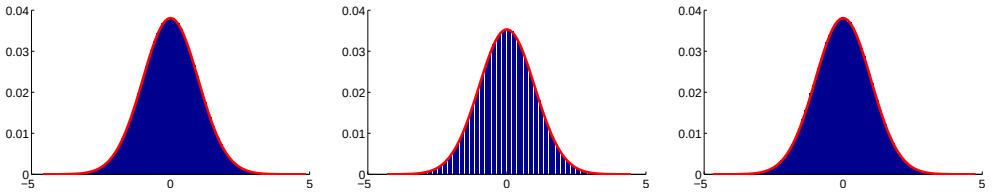
We will now show how to develop such approximate tests by reformulating the MH test as a statistical decision problem. It is easy to see that the original MH test (Equation 1) is equivalent to the following procedure: Draw $u \sim \text{Uniform}[0,1]$ and accept the proposal θ' if the average difference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

$$\mu_0 = \frac{1}{N}\log\left[u\frac{\rho(\theta_t)q(\theta'|\theta_t)}{\rho(\theta')q(\theta_t|\theta')}\right], \text{ and}$$
$$\mu = \frac{1}{N}\sum_{i=1}^{N} l_i \text{ where } l_i = \log p(x_i;\theta') - \log p(x_i;\theta_t) \tag{2}$$

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 $\{l_{i_1}, \ldots, l_{i_n}\}$ drawn without replacement from the population $\{l_1, \ldots, l_N\}$, 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) till we have enough evidence to make a decision.

More formally, we test the null hypothesis $H_0 : \mu = \mu_0$ vs the alternate $H_1 : \mu \neq \mu_0$. To do this, we proceed as follows: We compute the sample mean $\bar{l}$ and the sample standard deviation $s_l = \sqrt{\sum_{i=1}^{n}(l_i - \bar{l})^2/(n-1)}$. Then the standard deviation of $\bar{l}$ can be estimated as

$$s = \frac{s_l}{\sqrt{n}}\sqrt{1 - \frac{n}{N}} \tag{3}$$



(a) n=500 (b) n=5000 (c) n=10000

Figure 1: Theoretical (red line) and empirical distribution (blue bars) of the t-statistic under resampling n datapoints without replacement from a dataset composed of digits 7 and 9 from the MNIST dataset (total $N = 12214$ points).

where $\sqrt{1 - \frac{n}{N}}$, 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:

$$t = \frac{\bar{l} - \mu_0}{s} \tag{4}$$

If n is large enough for the central limit theorem to hold, the test statistic t follows a standard Student-t distribution with $n - 1$ degrees of freedom, if the null hypothesis is true (see Figure 4 for empirical verification). Then, we compute the p-value $\delta = 1 - \phi_{n-1}(|t|)$ where $\phi_{n-1}(.)$ is the cdf of the standard Student-t distribution with $n-1$ degrees of freedom. If $\delta < \epsilon$ we can reject the null hypothesis, i.e. we have enough precision in the sample to make a decision. In this case, if $\bar{l} > \mu_0$, we accept the proposal, otherwise we reject it. If we fail to reject the null hypothesis, we draw more data to reduce the uncertainty, s, in the sample mean $\bar{l}$. We keep drawing more data until we have enough precision to make a decision. 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, and the sample mean $\bar{l} = \mu$. So, in this case 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 have assumed that the total number of datapoints is divisible by the size of the mini-batch for simplicity). This algorithm is similar to the Pocock sequential design [Pocock, 1977] which allows us to set a value of ϵ given a particular value of Type I error.

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.

We will now prove an upper bound in the error of the stationary distribution of our approximate algorithm. Denote by $\mathcal{T}_0$ the transition kernel of the exact Metropolis-Hastings algorithm, and by $d_v(P, Q) \stackrel{\text{def}}{=} \frac{1}{2} \int_{\theta \in \Theta} |f_P(\theta) - f_Q(\theta)| d\Omega(\theta)$ the total variation distance between two distributions, P and Q, that are absolutely continuous w.r.t. Ω with their Radon-Nikodym derivatives f_P and f_Q respectively. Let $P_{a,\epsilon}(\theta, \theta')$ be the actual acceptance probability of our approximate Metropolis-Hastings test. Given an upper bound on the error of this probability: $\Delta_{\max} = \sup_{\theta,\theta'} |P_{a,\epsilon}(\theta, \theta') - P_a(\theta, \theta')|$, and if $\mathcal{T}_0$ satisfies the contraction condition $d_v(P\mathcal{T}_0, \mathcal{S}_0) \leq \eta d_v(P, \mathcal{S}_0)$ for all probability distributions P and a constant $\eta \in [0, 1)$, we can prove the following upper bound on the discrepancy between the posterior distribution and the stationary distribution of our approximate Markov chain:

Algorithm 1 Approximate MH test

Input: θ_t, θ', ϵ, m, X_N
Output: $accept$

Initialize estimated means $\bar{l} \leftarrow 0$ and $\bar{l^2} \leftarrow 0$
Initialize $n \leftarrow 0$
Draw $u \sim$ Uniform[0,1]
Compute $\mu_0 \leftarrow \frac{1}{N} \log \left[u \frac{\rho(\theta_t) q(\theta'|\theta_t)}{\rho(\theta') q(\theta_t|\theta')} \right]$,
$done \leftarrow$ **false**
while not done **do**
  Draw mini-batch $\mathcal{X}_m$ of size m without replacement from X_N
  Set $X_N \leftarrow X_N \setminus \mathcal{X}_m$
  Update $\bar{l}$ and $\bar{l^2}$ using $\mathcal{X}_m$.
  $n \leftarrow n + m$
  Compute estimated standard deviation $s \leftarrow \sqrt{1 - \frac{n}{N}} \sqrt{\frac{\bar{l^2} - (\bar{l})^2}{n-1}}$
  Compute student-t statistic $t \leftarrow \frac{\bar{l} - \mu_0}{s}$
  Compute p-value $\delta \leftarrow 1 - \phi_{n-1}(|t|)$
  if $\delta < \epsilon$ **then**
    $accept \leftarrow$ **true** if $\bar{l} > \mu_0$ and **false** otherwise
    $done \leftarrow$ **true**
  end if
end while

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*

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

The proof is given in the appendix. The theorem applies to both continuous and discrete state spaces.

5. Experiments

5.1 Random Walk - Logistic Regression

We first test our method using a random walk proposal $q(\theta'|\theta_t) = \mathcal{N}(\theta_t, \sigma^2_{RW})$. Although the random walk proposal is not efficient, it is very useful for illustrating our algorithm because the proposal contains no information about the target distribution, unlike Langevin/Hamiltonian methods. So, the responsibility of converging to the correct distribution lies solely with the MH test. Also note that since q is symmetric i.e. $q(\theta_t|\theta') = q(\theta'|\theta_t)$, it does not appear in the MH test at all. Therefore we can conduct the MH test with:

$$\mu_0 = \frac{1}{N} \log \left[u \frac{\rho(\theta_t)}{\rho(\theta')} \right], \tag{5}$$

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 Gaussian prior on the parameters.

In figure 2, we show the average percentage of data used per test (blue line) and the percentage of wrong decisions (red line) as a function of ϵ. The amount of data required drops off very fast as ϵ is increased even though the error does not increase much. Note that the setting $\epsilon = 0$ corresponds to the exact MH algorithm.

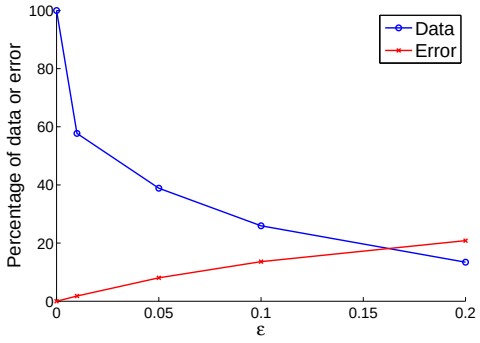


Figure 2: Percentage of data (blue line) used and percentage of wrong accept-reject decisions (red line) as a function of ϵ.

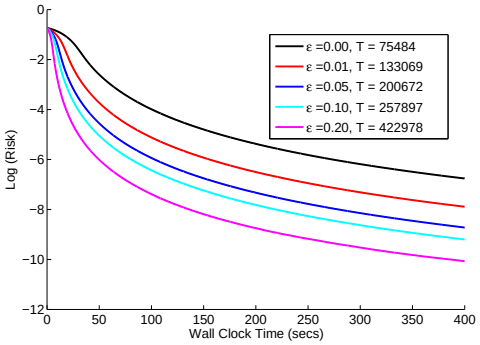
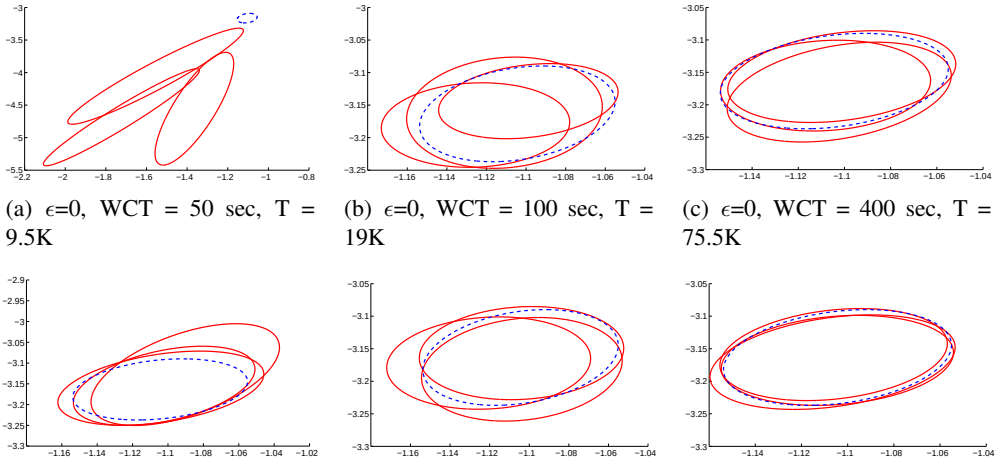


Figure 3: Average Risk in predictive mean of 2037 test points vs wall clock time for different values of ϵ. T denotes the total number of samples collected in 400 secs.



(a) ϵ=0, WCT = 50 sec, T = 9.5K

(b) ϵ=0, WCT = 100 sec, T = 19K

(c) ϵ=0, WCT = 400 sec, T = 75.5K

(d) ϵ=0.01, WCT = 50 sec, T = 16.5K

(e) ϵ=0.01, WCT = 100 sec, T = 33K

(f) ϵ=0.01, WCT = 400 sec, T = 133K

Figure 4: Marginals θ_1 vs θ_t for $\epsilon = 0$ and $\epsilon = 0.01$ at different values of wall clock time. The actual number of samples T is shown as well. The blue curve shows the true marginal obtained using a long run of Hybrid Monte Carlo. The red curve shows marginals obtained by running the random walk algorithm 3 times.

In figure 4, we show marginals of θ_1 vs θ_2 for $\epsilon = 0$ and $\epsilon = 0.01$ at different values of wall clock time. The red curves are marginals for 3 different runs of the random walk algorithm whereas the blue curve shows the true marginal obtained from a long run of Hybrid Monte Carlo. At T = 50 sec, the exact MH algorithm $\epsilon = 0$ has still not completed burn-in. Our algorithm with $\epsilon = 0.01$ is able to accelerate burn-in because it can collect more samples in a given amount of time. Theoretically, as more computational time becomes available the exact MH algorithm will catch up with (and eventually outperform) our algorithm, because the exact MH algorithm is unbiased unlike ours. But the bias is

hardly noticeable in this example and the error is still dominated by the variance even after collecting around 100K samples.

In figure 3, we show how the logarithm of the risk in the predictive mean of a test point decreases as a function of wall clock time. The risk is computed by measuring the squared error in the estimate (of predictive mean), and averaging it over 8 different simulations of the Markov chain. 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 $\epsilon > 0$.

5.2 Random Walk on Stiefel Manifold - Independent Component Analysis

Next, we use our algorithm to sample from the posterior distribution of the unmixing matrix in Independent Component Analysis (ICA) [Hyvärinen and 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]. To propose a new value of W, we first randomly discard a row w_k. Then, we randomly pick a row w_j, scale it by a constant ψ, add standard Gaussian noise, project it onto the subspace orthogonal to the other $D-2$ vectors and then renormalize it to unit length. Then we set w_k to the unique unit vector orthonormal to the other $D-1$ vectors. For a more detailed description, see Ouyang [2008]. The constant ψ determines the peakedness of the proposal distribution. This is a symmetric proposal distribution and does not appear in the MH ratio. Therefore, we can use:

$$\mu_0 = \frac{1}{N} \log [u] , \tag{6}$$

In order to perform a large scale experiment, we created a synthetic dataset by mixing 4 sources: a) a Classical music recording b) street / traffic noise c) standard Gaussian noise d) another independent standard Gaussian noise source. We used a 43 second clip of each source sampled at 44.1 kHz resulting in about 1.95 million samples.

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 inverse of the true mixing 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 around 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 figure 5. 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 with $\epsilon = 0$. However, if we were to run for an infinitely long time, we would expect the exact algorithm to outperform all the approximate algorithms.

5.3 Reversible Jump - 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 very similar to the Bayesian LASSO model for linear regression described in Chen et al. [2011].

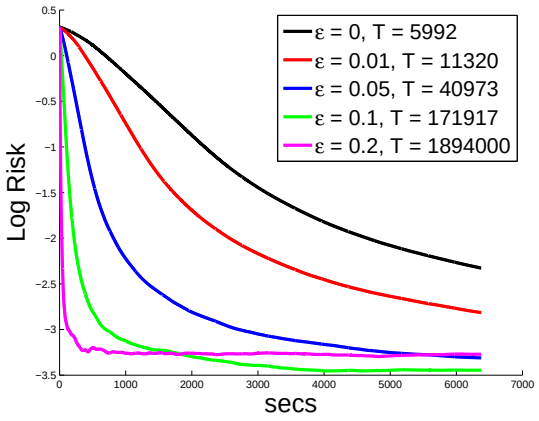


Figure 5: Risk in mean of Amari distance for the ICA model

Specifically, given D input features (covariates), our parameter $\theta = \{\beta, \gamma\}$ where β is a D dimensional vector of 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 on γ as in Chen et al. [2011] to control the size of the model, $k = \sum_{j=1}^{D} \gamma_j$:

$$p(\gamma|\lambda) \propto \frac{\lambda^k}{\binom{D}{k} k!} \tag{7}$$

The extra $\binom{D}{k}$ factor appears because there are $\binom{D}{k}$ different models of size k. The likelihood of the data is:

$$p(y_N|X_N, \beta, \gamma) = \prod_{i=1}^{N} \left[\text{sigm}(\beta^T x_i)^{y_i}(1 - \text{sigm}(\beta^T x_i)^{(1-y_i)}\right] \tag{8}$$

In the above, 'sigm' is the sigmoid function. We will denote this likelihood by $l_N(\beta, \gamma)$. Then, the posterior distribution after analytically integrating out ν is:

$$p(\beta, \gamma|X_N, y_N, \lambda) \propto l_N(\beta, \gamma)\|\beta\|_1^{-k}\lambda^k B(k, D - k + 1) \tag{9}$$

where $B(.,.)$ is the beta function. We do not analytically integrate out λ, but instead use it as a knob 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. The update move is the usual MCMC move which involves changing the parameter vector β without changing the model γ. Specifically, we randomly pick an active component $j : \gamma_j = 1$ and set $\beta_j = \beta_j + \eta$ where $\eta \sim \mathcal{N}(0, \sigma_{update})$. The birth move involves (for $k < D$) randomly picking an inactive component $j : \gamma_j = 0$ and setting $\gamma_j = 1$. We also propose a new value for $\beta_j \sim \mathcal{N}(0, \sigma_{birth})$. The birth move is paired with a corresponding death move (for $k > 1$) which involves randomly picking an active component $j : \gamma_j = 1$ and setting $\gamma_j = 0$. The corresponding β_j is discarded. The probabilities of picking these moves $p(\gamma \to \gamma')$ is the same as in Chen et al. [2011]. The value of μ_0 used in the MH test for different moves is

given below.
1. Update move:

$$\mu_0 = \frac{1}{N}\log\left[u\frac{\|\beta\|_1^{-k}}{\|\beta'\|_1^{-k}}\right] \tag{10}$$

2. Birth move:

$$\mu_0 = \frac{1}{N}\log\left[u\frac{\|\beta\|_1^{-k}p(\gamma\to\gamma')\mathcal{N}(\beta_j|0,\sigma_{birth})(D-k)}{\|\beta'\|_1^{-(k+1)}p(\gamma'\to\gamma)\lambda k}\right] \tag{11}$$

2. Death move:

$$\mu_0 = \frac{1}{N}\log\left[u\frac{\|\beta\|_1^{-k}p(\gamma\to\gamma')\lambda(k-1)}{\|\beta'\|_1^{-(k-1)}p(\gamma'\to\gamma)\mathcal{N}(\beta_j|0,\sigma_{birth})(D-k+1)}\right] \tag{12}$$

For more details see Chen et al. [2011]. We applied this to the MiniBooNE dataset from the UCI machine learning repository[Bache and Lichman, 2013]. Here the task is to classify electron neutrinos (signal) from muon neutrinos (background). There are 130,065 data-points with 50 features to which we add a constant feature of 1's. Around 28% of the examples are in the positive class. We randomly split the data into a training (80%) and testing (20%) set. First, to compute ground truth, we collected T=400K samples using the exact reversible jump algorithm. Then, we ran different algorithms with $\epsilon = 0$, $\epsilon = 0.01$, $\epsilon = 0.05$ and $\epsilon = 0.1$ for around 3500 seconds. We plot the risk in predictive mean of test data (estimated from 10 Markov chains) in figure 6 . Again we see that the lowest risk is obtained with $\epsilon > 0$. We also plot the marginal posterior probability of including a feature in the model, i.e. $p(\gamma_j = 1|X_N, y_N, \lambda)$ in figure 7.

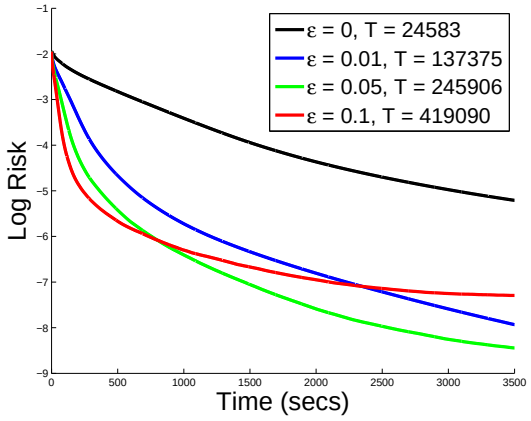


Figure 6: Risk in predictive mean of test data for variable selection using Reversible jump MCMC for different values of ϵ

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 figure 6, 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". We can, however, see that when initialized with the same values, we obtain similar results with the approximate algorithm and the exact algorithm.

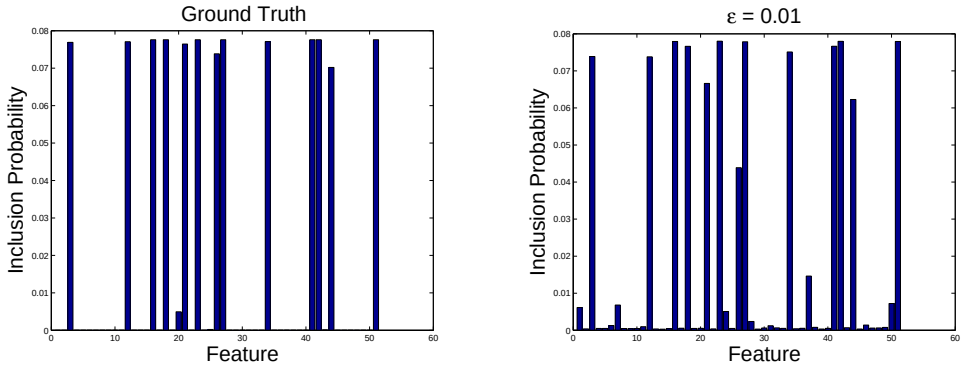


Figure 7: Marginal probability of features to be included in the model

5.4 Stochastic Gradient Langevin Dynamics (SGLD)

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

$$q(.|\theta, \mathcal{X}_n) = \mathcal{N}\left(\theta + \frac{\alpha}{2}\nabla_\theta \left\{\frac{N}{n}\sum_{x\in\mathcal{X}_n} \log p(x|\theta) + \log \rho(\theta)\right\}, \alpha\right) \tag{13}$$

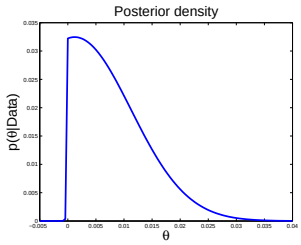
The proposed state θ' is always accepted (without conducting any MH test). Since the acceptance probability approaches 1 as we reduce α, the bias from not conducting the MH test can be kept under control by using $\alpha \approx 0$. However, we have to use a reasonably large α to keep the mixing rate 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 8(a) and 8(b) respectively.

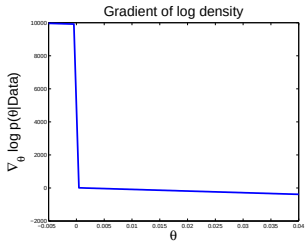
The empirical histogram of 100000 samples obtained by running SGLD with $\alpha = 5 \times 10^{-6}$ are shown as red bars in figure 8(c). The effect of omitting the MH test is quite severe in this case. When the sampler reaches the mode of the distribution, the Langevin noise occasionally throws it into the valley to the left, where the gradient is very high. This high gradient 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 each of the component kernels $q(\theta'|\theta, \mathcal{X}_n)$ satisfy detailed balance. Thus, we can use an MH test with:

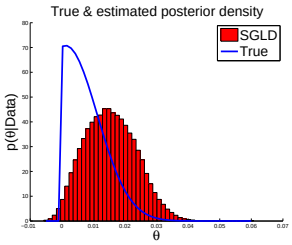
$$\mu_0 = \frac{1}{N}\log\left[u\frac{\rho(\theta_t)q(\theta'|\theta_t, \mathcal{X}_n)}{\rho(\theta')q(\theta_t|\theta', \mathcal{X}_n)}\right], \tag{14}$$



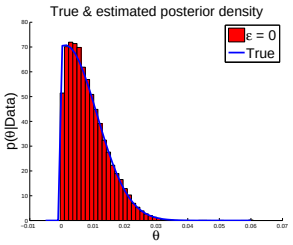
(a) Posterior density



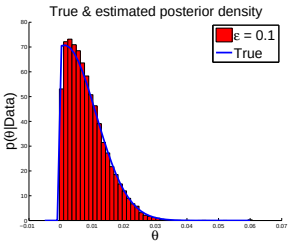
(b) Gradient of log posterior



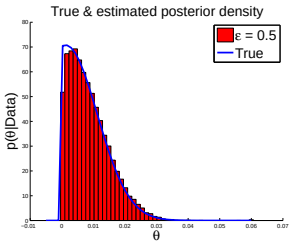
(c) Stationary distribution of uncorrected SGLD



(d) Stationary distribution of SGLD corrected using the exact MH test



(e) Stationary distribution of SGLD corrected using our approximate MH test, with $\epsilon = 0.1$.



(f) Stationary distribution of SGLD corrected using our approximate MH test, with $\epsilon = 0.5$.

Figure 8: Pitfalls of using uncorrected Stochastic Gradient Langevin Dynamics with certain distributions

The result of running SGLD (keeping $\alpha = 5 \times 10^{-6}$ as before) with the exact MH correction is shown in figure 8(d). As expected, we are now able to sample correctly as the MH test rejects most proposals from the mode to the valley. Results of running SGLD with our approximate MH test are shown in figure 8(e) ($\epsilon = 0.1$) and figure 8(f)($\epsilon = 0.5$) respectively. The stationary distribution obtained with $\epsilon = 0.1$ is almost indistinguishable from that obtained by the exact MH test, although on average, the approximate test uses only 14.2% of the data per test. The bias with $\epsilon = 0.5$ is also negligible, even though it uses only 5% of the data per test, which is no more than the size of the mini-batch we used in the SGLD proposal distribution. Note that when $\epsilon = 0.5$, a decision is always made in the first step, without querying more data sequentially.

5.5 Troubleshooting

Our algorithm will behave erratically if the Central Limit Theorem(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. However, we expect any algorithm that does not examine every datapoint to suffer from similar problems on such datasets.

6. Conclusions and Future Work

In this 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 have focused on a fixed decision threshold for accepting or rejecting a sample, but in theory a better algorithm can be obtained by adapting the decision 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.

7. Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Number 1216045. The authors would like to thank Alexander Ihler, Daniel Gillen and Babak Shahbaba for their invaluable suggestions.

A. Proof of Theorem 1

A.1 Upper Bound Based on One Step Error

We first prove a lemma that will be used for the proof of Theorem 1.

Lemma 2. *Given two transition kernels, $\mathcal{T}_0$ and $\mathcal{T}_\epsilon$, with respective stationary distributions, $\mathcal{S}_0$ and $\mathcal{S}_\epsilon$, if $\mathcal{T}_0$ satisfies the following contraction condition with a constant $\eta \in [0, 1)$ for all probability distributions P:*

$$d_v(P\mathcal{T}_0, \mathcal{S}_0) \leq \eta d_v(P, \mathcal{S}_0) \tag{15}$$

and the one step error between $\mathcal{T}_0$ and $\mathcal{T}_\epsilon$ is upper bounded uniformly with a constant $\Delta > 0$ as:

$$d_v(P\mathcal{T}_0, P\mathcal{T}_\epsilon) \leq \Delta, \forall P \tag{16}$$

then the distance between $\mathcal{S}_0$ and $\mathcal{S}_\epsilon$ is bounded as:

$$d_v(\mathcal{S}_0, \mathcal{S}_\epsilon) \leq \frac{\Delta}{1-\eta} \tag{17}$$

Proof. Consider a Markov chain with transition kernel $\mathcal{T}_\epsilon$ initialized from an arbitrary distribution P. Denote the distribution after t steps by $P^{(t)} \stackrel{\text{def}}{=} P\mathcal{T}_\epsilon^t$. At every time step, $t \geq 0$, we apply the transition kernel $\mathcal{T}_\epsilon$ on $P^{(t)}$. According to the one step error bound in Equation 16, the distance between $P^{(t+1)}$ and the distribution obtained by applying $\mathcal{T}_0$ to $P^{(t)}$ is upper bounded as:

$$d_v(P^{(t+1)}, P^{(t)}\mathcal{T}_0) = d_v(P^{(t)}\mathcal{T}_\epsilon, P^{(t)}\mathcal{T}_0) \leq \Delta \tag{18}$$

Following the contraction condition of $\mathcal{T}_0$ in Equation 15, the distance of $P^{(t)}\mathcal{T}_0$ from its stationary distribution $\mathcal{S}_0$ is less than $P^{(t)}$ as

$$d_v(P^{(t)}\mathcal{T}_0, \mathcal{S}_0) \leq \eta d_v(P^{(t)}, \mathcal{S}_0) \tag{19}$$

Now let us use the triangle inequality to combine Equation 18 and 19 to obtain an upper bounded for the distance between $P^{(t+1)}$ and $\mathcal{S}_0$:

$$d_v(P^{(t+1)}, \mathcal{S}_0) \leq d_v(P^{(t+1)}, P^{(t)}\mathcal{T}_0) + d_v(P^{(t)}\mathcal{T}_0, \mathcal{S}_0) \leq \Delta + \eta d_v(P^{(t)}, \mathcal{S}_0) \tag{20}$$

Let $r < 1-\eta$ be any positive constant and consider the ball $\mathcal{B}(\mathcal{S}_0, \frac{\Delta}{r}) \stackrel{\text{def}}{=} \{P : d_v(P, \mathcal{S}_0) < \frac{\Delta}{r}\}$. When $P^{(t)}$ is outside the ball, we have $\Delta \leq r d_v(P^{(t)}, S)$. Plugging this into Equation 20, we can obtain a contraction condition for $P^{(t)}$ towards $\mathcal{S}_0$:

$$d_v(P^{(t+1)}, \mathcal{S}_0) \leq (r+\eta) d_v(P^{(t)}, \mathcal{S}_0) \tag{21}$$

So if the initial distribution P is outside the ball, the Markov chain will move monotonically into the ball within a finite number of steps. Let us denote the first time it enters the ball as t_r. If the initial distribution is already inside the ball, we simply let $t_r = 0$. We then show by induction that $P^{(t)}$ will stay inside the ball for all $t \geq t_r$.

1. At $t = t_r$, $P^{(t)} \in \mathcal{B}(\mathcal{S}_0, \frac{\Delta}{r})$ holds by the definition of t_r.
2. Assume $P^{(t)} \in \mathcal{B}(\mathcal{S}_0, \frac{\Delta}{r})$ for some $t \geq t_r$. Then, following Equation 20, we have

$$d_v(P^{(t+1)}, \mathcal{S}_0) \leq \Delta + \eta\frac{\Delta}{r} = \frac{r+\eta}{r}\Delta < \frac{\Delta}{r} \implies P^{(t+1)} \in \mathcal{B}(\mathcal{S}_0, \frac{\Delta}{r}) \tag{22}$$

Therefore, $P^{(t)} \in \mathcal{B}(\mathcal{S}_0, \frac{\Delta}{r})$ holds for all $t \geq t_r$. Since $P^{(t)}$ converges to S_ϵ, it follows that:

$$d_v(\mathcal{S}_\epsilon, \mathcal{S}_0) < \frac{\Delta}{r}, \forall r < 1-\eta \tag{23}$$

Taking the limit $r \to 1-\eta$, we prove the lemma:

$$d_v(\mathcal{S}_\epsilon, \mathcal{S}) \leq \frac{\Delta}{1-\eta} \tag{24}$$

□

A.2 Proof of Theorem 1

We first derive an upper bound for the one step error of the approximate Metropolis-Hastings algorithm, and then use Lemma 2 to prove Theorem 1. The transition kernel of the exact Metropolis-Hastings algorithm can be written as

$$\mathcal{T}_0(\theta, \theta') = P_a(\theta, \theta') q(\theta'|\theta) + (1 - P_a(\theta, \theta'))\delta_D(\theta' - \theta) \tag{25}$$

where δ_D is the Dirac delta function. For the approximate algorithm proposed in this paper, we use an approximate MH test with acceptance probability $P_{a,\epsilon}(\theta, \theta')$ where the error, $\Delta P_a \stackrel{\text{def}}{=} P_{a,\epsilon} - P_a$, is upper bounded as $|\Delta P_a| \leq \Delta_{\text{max}}$. Now let us look at the distance between the distributions generated by one step of the exact kernel $\mathcal{T}_0$ and the approximate kernel $\mathcal{T}_\epsilon$. For any P,

$$\begin{aligned}
&\int_{\theta'} d\Omega(\theta') |(P\mathcal{T}_\epsilon)(\theta') - (P\mathcal{T}_0)(\theta')| \\
&= \int_{\theta'} d\Omega(\theta') \left| \int_\theta dP(\theta) \Delta P_a(\theta, \theta') \left(q(\theta'|\theta) - \delta_D(\theta' - \theta) \right) \right| \\
&\leq \Delta_{\text{max}} \int_{\theta'} d\Omega(\theta') \left| \int_\theta dP(\theta) (q(\theta'|\theta) + \delta_D(\theta' - \theta)) \right| \\
&= \Delta_{\text{max}} \int_{\theta'} d\Omega(\theta') \left(g_Q(\theta') + g_P(\theta') \right) = 2\Delta_{\text{max}}
\end{aligned} \tag{26}$$

where $g_Q(\theta') \stackrel{\text{def}}{=} \int_\theta dP(\theta) q(\theta'|\theta)$ is the density that would be obtained by applying one step of Metropolis-Hastings without rejection. So we get an upper bound for the total variation distance as

$$d_v(P\mathcal{T}_\epsilon, P\mathcal{T}_0) = \frac{1}{2} \int_{\theta'} d\Omega(\theta') |P\mathcal{T}_\epsilon - P\mathcal{T}_0| \leq \Delta_{\text{max}} \tag{27}$$

Apply Lemma 2 with $\Delta = \Delta_{\text{max}}$ and we prove Theorem 1.

References

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

S.-i. Amari, A. Cichocki, H. H. Yang, et al. A new learning algorithm for blind signal separation. *Advances in neural information processing systems*, pages 757–763, 1996.

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

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

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

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

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

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

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

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21: 1087, 1953.

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

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

S. Singh, M. Wick, and A. McCallum. 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*, pages 1104–1113. Association for Computational Linguistics, 2012.

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