A Simple Minibatch Acceptance Test for MCMC

Anonymous Author(s)
Affiliation
Address
email

Abstract

Markov chain Monte Carlo (MCMC) methods have many applications in machine learning. We are particularly interested in their application to modeling very large datasets, where it is impractical to perform Metropolis-Hastings tests on the full data. To do this, we describe a novel acceptance test which is applied to minibatches of data. Previous work on minibatch Metropolis-Hastings use adaptive batch sizes, which may use far more than one minibatch of data and therefore yield few samples. Here we describe a simpler test on fixed-sized minibatches. Instead of adaptive batch sizes, our test requires a precondition on the variance of the log proposal probability ratio. Given a measurement of this variance, a minibatch size can be chosen to support the test in one step, which yields a number of samples proportional to the batch/minibatch ratio. Alternatively, the variance condition can be used to adjust the temperature of the sampling distribution to a "natural" value to provide regular samples at a given minibatch size. The resulting proposal/test is only slightly more complex than a simple SGD update. In this paper we derive the test, discuss its implementation, and present several experiments.

1 Introduction

Markov chain Monte Carlo (MCMC) sampling is a powerful method for computation on intractable distributions. MCMC algorithms propose samples from a proposal distribution q which is in general different from the target distribution p, and decide whether to accept or reject them based on an acceptance test. The acceptance test is usually a Metropolis-Hastings test [1, 2]. For example, for Bayesian posterior inference, the posterior distribution $p(\theta \mid x_1, \ldots, x_N) \propto p(\theta) \prod_{i=1}^{N} p(x_i \mid \theta)$ of parameter θ based on conditionally independent data $\{x_i\}_{i=1}^{N}$ is the target distribution which one aims to approximate by sampling θ values. When N is large, it is very expensive to compute the exact posterior because the likelihood must be evaluated on every x_i.

Many state-of-the-art machine learning methods, and deep learning in particular, are based on minibatch updates (such as SGD) to a model. Minibatch updates produce many improvements to the model for each pass over the dataset, and therefore have high sample efficiency. They also map very well onto hardware such as GPUs. In contrast, MCMC methods typically require calculations over the full dataset to produce a new sample. A recent result from [3] attempted to bridge the gap between minibatch optimization and MCMC by proposing an *adaptive minibatch* method. Their method uses a subset of the N data points for the MH test each iteration. However the number of points is not known a priori, and is adjusted dynamically during every test. This undermines many of the practical advantages of minibatch calculations: GPU kernels work best for certain minibatch sizes, memory blocks can be reused when minibatch size is fixed, etc.

In this paper, we develop an alternative minibatch acceptance test for MCMC methods which uses only one fixed-size minibatch per test. Instead of dynamically adjusting the batch size, we add a prerequisite for the test based on the variance of the log acceptance probability. If this variance is

Submitted to 29th Conference on Neural Information Processing Systems (NIPS 2016). Do not distribute.

small enough, the test requires only a single step. This variance changes slowly on typical problems, and we can estimate it before performing the test. The minibatch variance decreases as the minibatch size grows, so the variance measurement can be used to set the minimum minibatch size.

There is another advantage when using our test for Bayesian posterior inference. As the dataset size grows, the posterior distribution sharpens and often concentrates to a few sharp peaks. Samples from this distribution will come from those peaks, and will easily become stuck in one of them. The samples are not exploring the posterior parameter space, and are unlikely to concentrate at a strong local optimum. They therefore fail to accurately represent a sample of the posterior because of inadequate search (i.e., mixing). We argue that it is much more natural to sample the distribution at a higher temperature determined by the minibatch size rather than the dataset size. Starting at a high temperature (small minibatch size) which mixes easily, one can reduce the temperature (increasing minibatch size) until one reaches the desired target distribution temperature. Although this approach also involves changing minibatch size, the changes increase minibatch size monotonically, and only a few changes are performed in the course of a sampling session. Annealing can also be used for MAP and ML estimation because changing temperature preserves posterior modes.

To be precise, the contributions of this paper are as follows:

1. We develop a new minibatch acceptance test which satisfies detailed balance.
2. We compare performance of our new test and the adaptive sampling method.
3. We experiment using the approach for posterior estimation.

2 Preliminaries and Related Work

In standard MCMC methods [4, 5] for Bayesian inference with parameter θ and conditionally independent data, the goal is to compute the distribution $p(\theta \mid x_1, \ldots, x_N)$. To do so, one generates a chain of (correlated) samples $\theta_1, \ldots, \theta_T$ for large T and approximates p using the sample counts. Each iteration t has a current θ_t, and a *proposal distribution* $q(\theta' \mid \theta_t)$ determines a new candidate θ'. With probability P_a, the sample is accepted, so $\theta_{t+1} = \theta'$. Otherwise, $\theta_{t+1} = \theta_t$. This is done by drawing a uniform random variable $u \sim \mathrm{Unif}[0, 1]$ and accepting if $u < P_a$. Traditionally, P_a is

$$P_a = \min\left\{1, \frac{f(\theta')q(\theta_t \mid \theta')}{f(\theta_t)q(\theta' \mid \theta_t)}\right\} = \min\left\{1, \frac{p(\theta')\prod_{i=1}^N p(x_i \mid \theta')q(\theta_t \mid \theta')}{p(\theta_t)\prod_{i=1}^N p(x_i \mid \theta_t)q(\theta' \mid \theta_t)}\right\}, \tag{1}$$

where $f(\theta_t) = p(\theta_t \mid x_1, \ldots, x_N)$. This P_a satisfies detailed balance, so if one samples long enough, one will arrive at a stationary distribution matching the posterior, though a burn-in period and/or only using samples at regular intervals is often done in practice.

Unfortunately, computing f requires the use of all N training data points. Moreover, it is difficult to design tests using substantially fewer than N points that also satisfy detailed balance. To reconcile these competing objectives, [3] proposes an adaptive minibatch MCMC algorithm which uses a sequential hypothesis test. During each iteration, the algorithm starts with a small minibatch of data and tests the hypothesis that the sample θ' should be accepted or rejected based on a probability ratio. If the test cannot make a decision over a certain confidence threshold, then the minibatch size is increased and the test repeats. This process continues until a decision. The downside of this algorithm is the need to keep incrementing the minibatch size; in the worst case, all N data points may be needed to decide on θ'. A similar approach from [6] has slightly more robust theoretical properties based on concentration inequalities, but comes at the cost of more computational time.

The work of [7] presents another minibatch MCMC method which gets the *exact* posterior, so long as there exists a (cheap) lower bound on the likelihood for each data point. Their approach is only a starting point, however, and it can be difficult to derive these lower bounds in general cases. The extra auxiliary variables they introduce also hurt mixing.

There is a separate line of MCMC work on simulating the physics of a probability distribution. By viewing random variables as particles in a system, one can apply Hamiltonian Monte Carlo (HMC) [8] methods which generate the best case scenario for proposals: high quality *and* distant. HMC methods require a full gradient computation of the posterior, which means they also face the problem of computing the likelihood at each point. Stochastic versions of HMC [9, 10] use a minibatch of data each iteration but require controlling the minibatch noise to avoid divergence.

Langevin Dynamics [11, 12] is a similar approach which does not use the momentum terms in HMC. Minibatch SGHMC and Langevin Dynamics are orthogonal to our work and are other ways of attempting to solve the same problem of MCMC methods on big data. We can combine our methods with these results, as we demonstrate in Section 5.3.

3 A New Metropolis-Hastings Test

For our new MH test, we use two key values, Δ and Δ':

$$\Delta = \log\left(\frac{p(\theta')\prod_{i=1}^{N} p(x_i \mid \theta')q(\theta_t \mid \theta')}{p(\theta_t)\prod_{i=1}^{N} p(x_i \mid \theta_t)q(\theta' \mid \theta_t)}\right);\ \Delta' = \log\left(\frac{p(\theta')(\prod_{i=1}^{n} p(x_i \mid \theta'))^{\frac{N}{n}} q(\theta_t \mid \theta')}{p(\theta_t)(\prod_{i=1}^{n} p(x_i \mid \theta_t))^{\frac{N}{n}} q(\theta' \mid \theta_t)}\right) \tag{2}$$

where p and q have similar definitions as in Equation 1. Thus, Δ and Δ' are probability ratios in log space. Note that the only difference between Δ and Δ' is that they use N and n data terms x_i, respectively, and that the latter uses an N/n scaling term. We assume $n \ll N$ (so Δ' is substantially faster to compute) and that the n data points x_i represent a random minibatch without replacement.

3.1 The Full Data Test

To motivate the use of Δ, we turn to Lemma 1.

Lemma 1. *An acceptance function g such that $g(\Delta) = \exp(\Delta)g(-\Delta)$ satisfies detailed balance. That is, $f(\theta_t)p(\theta' \mid \theta_t) = f(\theta')p(\theta_t \mid \theta')$, where $p(\theta_y \mid \theta_x)$ is the probability of jumping from θ_x to θ_y in our chain.*

Proof. We begin by deriving $p(\theta' \mid \theta_t)$. This is equivalent to the probability of proposing θ' and then accepting it, so $p(\theta' \mid \theta_t) = q(\theta' \mid \theta_t)g(\Delta)$. Similarly, $p(\theta_t \mid \theta') = q(\theta_t \mid \theta')g(-\Delta)$. Notice that the probability of accepting a transition from θ' to θ_t is $g(-\Delta)$ because this inverts the fraction inside the logarithm term of Δ. By assumption, we can expand $g(\Delta) = \exp(\Delta)g(-\Delta)$ in $p(\theta' \mid \theta_t)$. Doing this, and combining the result with the definition of $p(\theta_t \mid \theta')$, we get

$$g(-\Delta) = \frac{p(\theta' \mid \theta_t)}{q(\theta' \mid \theta_t)\exp(\Delta)} = \frac{p(\theta_t \mid \theta')}{q(\theta_t \mid \theta')}. \tag{3}$$

Rearranging terms and expanding $\exp(\Delta)$, we have

$$\frac{p(\theta' \mid \theta_t)f(\theta_t)q(\theta' \mid \theta_t)}{q(\theta' \mid \theta_t)f(\theta')q(\theta_t \mid \theta')} = \frac{p(\theta_t \mid \theta')}{q(\theta_t \mid \theta')}. \tag{4}$$

Cancellations result in $f(\theta')p(\theta_t \mid \theta') = f(\theta_t)p(\theta' \mid \theta_t)$. Thus, detailed balance is satisfied. □

As a sanity check, the standard Metropolis-Hastings acceptance function $g(\Delta) = \min\{1, e^{\Delta}\} = \min\left\{1, \frac{f(\theta')q(\theta_t|\theta')}{f(\theta_t)q(\theta'|\theta_t)}\right\}$ satisfies the condition $g(\Delta) = \exp(\Delta)g(-\Delta)$.

For our MH test, the key is that we use a different g, the logistic function: $g(\Delta) = (1+\exp(-\Delta))^{-1}$. Straightforward arithmetic shows that it satisfies the condition in Lemma 1. The logistic function is nice because we can easily sample from it using the following procedure. At any iteration with a current parameter θ_t and a candidate sample θ', we can compute Δ. Let u be a uniform random variable $u \sim \mathrm{Unif}[0, 1]$. We accept θ' if $g(\Delta) > u$, and reject otherwise. This process is equivalent to sampling a random variable X with cumulative distribution function $F_X(x) = g(x)$, and accepting if $\Delta > X$ and rejecting otherwise. We can define $X = g^{-1}(u)$ so that its CDF is the logistic function, which follows because for arbitrary $X = x$, we have

$$F_X(x) = \Pr(X \le x) = \Pr(g^{-1}(u) \le x) = \Pr(u \le g(x)) = \int_0^{g(x)} 1dx = g(x),$$

as the density of u is one. Thus, the criteria to accept the candidate θ' is equivalent to whether $\Delta > X$. Moreover, the density of X (a standard logistic random variable) is symmetric about zero, so the acceptance criteria can also be expressed as $\Delta + X > 0$.

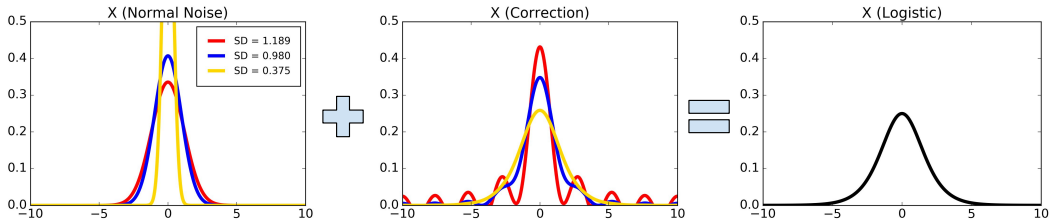

Figure 1: Three examples of X_{norm} and X_{corr} distributions that convolve to form the standard logistic distribution. We use three standard deviation values of X_{norm}. The two red (respectively, blue, green) curves form the logistic. The y-axis is capped at 0.5. This figure must be viewed in color.

3.2 The Minibatch Version

With large datasets, Δ is intractable to compute. An intuitive fix is to replace Δ with the minibatch version, Δ', for our MH test. The following lemma characterizes the distribution of Δ'.

Lemma 2. *If the minibatch data is chosen randomly without replacement and has sufficiently many elements, then the distribution of Δ' is approximately Gaussian.*

Proof. By expanding the definition of Δ' from Equation 2, we get

$$\Delta' = \log p(\theta') - \log p(\theta_t) + \log q(\theta_t \mid \theta') - \log q(\theta' \mid \theta_t) + \frac{N}{n}\sum_{i=1}^{n}(\log p(x_i \mid \theta') - \log p(x_i \mid \theta_t)).$$

During a given iteration, θ_t and θ' are fixed. Therefore, the randomness in Δ' comes only from the minibatch. Since the minibatch is chosen randomly, the distribution of $\sum_{i=1}^{n}(\log p(x_i \mid \theta') - \log p(x_i \mid \theta_t))$ converges to a Gaussian for sufficiently large N by the Central Limit Theorem. □

From Lemma 2, since Δ' is a noisy approximation of Δ, the relationship is expressed as

$$\Delta' = \Delta + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2), \tag{5}$$

for a Gaussian noise term ε. The noise means we cannot just replace $\Delta + X$ with $\Delta' + X$ for our test, so we need to slightly change our acceptance criteria. Our insight is to decompose X as

$$X = X_{\text{norm}} + X_{\text{corr}}, \tag{6}$$

where we assume X_{norm} is a zero-mean Gaussian and X_{corr} is a zero-mean "correction" term. These two add (i.e., their distributions convolve) to form X. The criteria to accept is now

$$\Delta + X = \Delta + X_{\text{norm}} + X_{\text{corr}} \approx \Delta' + X_{\text{corr}} > 0. \tag{7}$$

The approximation exists because X_{norm} is an estimate of ε. If X_{norm} is exact, then we get the same MH test with Δ' as we do with the full data version Δ.

It is important to understand why we use Equations 6 and 7. The deconvolution process allows us to use $\Delta' + X_{\text{corr}}$ as a proxy for $\Delta + X$. It works because if we know the distribution[1] of ε (from Equation 5), then we can pretend that our Δ' is in fact Δ. To "insert" ε into our test, we deconvolve X so that one of its components, X_{norm}, plays the part of ε. In light of our revised MH test, the previous discussion raises the question: in an iteration of MCMC with current θ_t and proposed θ', how do we know the distribution of X_{corr}? We need the correct distribution to sample the *values* of X_{corr} appropriately. This procedure consists of two major steps.

1. We first need to estimate the standard deviation of X_{norm}. We generate K values of Δ', each using a different random minibatch of n data points, but each using the same θ_t and θ'. In our experiments (see Section 5) we make K small, around 5-10, for speed purposes, and n is usually 50-200. Using these K values, we compute the empirical standard deviation $\text{std}(\Delta')$.

[1] Computing the distributions of ε and X_{norm} reduces to finding their standard deviations.

Input : Number of samples T, number of Δ' estimates K, minibatch size m, pre-computed X_{corr} distributions, $\{x_i\}_{i=1}^N$ values from a target distribution, and initial sample θ_0.
Output : A chain of T samples $\{\theta_1, \ldots, \theta_T\}$ for either posterior inference or MAP/ML estimation.

```
for t = {1, ..., T} do
    Propose a candidate θ' and create an empty list for Δ' estimates, d = [];
    for k = {1, ..., K} do
        Sample a random minibatch of size m from the complete data;
        Compute Δ' from this minibatch (see Equation 2), and add Δ' to d;
    end
    Compute std(d), the standard deviation estimate of Δ' from list d;
    if std(d) ≥ 1.2 then
        Skip loop (set θ_{t+1} = θ_t) and apply variance preconditioning fix (if desired);
    else
        Choose the closest X_corr distribution, and sample a value X_c from it;
        Using another random minibatch of m data points, compute a new Δ' (call it Δ'_real);
        if X_c + Δ'_real > 0 then
            Accept the candidate, θ_{t+1} = θ';
        else
            Reject and re-use the old one, θ_{t+1} = θ_t;
        end
    end
end
```

Algorithm 1: A description of our MH test within the MCMC algorithm.

2. Once we have $\text{std}(\Delta')$, we can determine the distribution of X_{corr} from standard deconvolution techniques. These rely on the well-known fact that the Fourier transform of a convolution is the product of the individual Fourier transforms. In practice, we pre-compute distributions of X_{corr} for different $\text{std}(\Delta')$ values to facilitate the sampling process.

Algorithm 1 describes our MH test within the MCMC algorithm.

3.3 Discussions and Practical Considerations of the New MH Test

The Convolution. Figure 1 demonstrates Equation 6 and provides three (color-coded) examples of densities for X_{norm} and X_{corr}. The density of X, a logistic random variable with mean $\mu = 0$ and scale $s = 1$, is known, as are the X_{norm} densities for different standard deviations. The deconvolution provides us with the X_{corr} distributions. Note, however, that as the standard deviation of X_{norm} increases, the X_{corr} distribution becomes increasingly unstable and "bumpier", because the logistic curve has fatter tails than Gaussians. In addition, we cap the standard deviation possibilities of X_{norm} at ≈ 1.2 in our experiments. In our experiments, we discretize the possible standard deviations of X_{norm}, and we also limit the densities to consider the range $[-10, 10]$ as shown in Figure 1.

Variance Preconditioning. Returning to the discussion from Section 1, our test requires a variance check. Equation 6 means that the variance of X, which is $\pi^2/3 \approx 3.29$ is an upper bound on the variances of X_{norm}. Therefore, the test cannot work if the variance of X_{norm} is too large. The standard deviation of X is about $\sqrt{3.29} \approx 1.81$, so this bounds $\text{std}(X_{\text{norm}})$ (as well as $\text{std}(\Delta')$). From the discussion above, we bound the standard deviation at 1.2 so that X_{corr} can handle the remaining noise. Recall from Section 3.2 that we estimate the $\text{std}(X_{\text{norm}})$ each iteration. If the variance/standard deviation test fails, then we can (a) skip the iteration, (b) increase the minibatch size, (c) change the proposal to decrease step sizes, or (d) increase temperature, as we now discuss.

Temperature. One way to satisfy the variance precondition is to increase the temperature of the target distribution. For a temperature $T > 1$, the augmented target distribution would become

$$\log p(\theta \mid x_1, \ldots, x_N) \approx \log p(\theta) + \frac{1}{T}\frac{N}{n}\sum_{i=1}^{n} \log p(x_i \mid \theta). \tag{8}$$

with an extra $1/T$ term augmented. As the temperature increases, the values of Δ' get smaller in absolute value, thus reducing variance. The flatter posterior results in easier mixing.

4 Theoretical Results

In this section, we explore the convergence properties of our MH test. Due to space constraints, all our proofs are in Appendix A.

We first introduce some notation. Let our true and approximated acceptance probabilities be $P_a = g(\Delta) = \Pr(\Delta + X > 0)$ and $P'_a = \Pr(\Delta' + X_{\text{corr}} > 0)$. Define $X_\xi = \varepsilon - X_{\text{norm}}$ which means, invoking Equations 5 and 6, that X_ξ is the random variable characterizing the accuracy of our X_{norm}. Thus, $\Delta' = \Delta + X_{\text{norm}} + X_\xi$. We assume the standard deviation of X_{norm} is less than $\pi/\sqrt{3}$, so that we can use Equation 6. We denote $F_X(x)$ as the CDF of X.

We use some terms from standard Markov chain theory [13]. Let the *transition kernels* of MCMC on complete and minibatch data at step i be P_i and P'_i, respectively. When these kernels (e.g., P_i) are applied on a probability distribution D_1, they generate a new distribution D_2, which we write as $P_i \circ D_1 = D_2$. Define π to be the stationary distribution obtained by P_i, and π'_i to be the stationary distribution obtained by P'_i (π' needs a subscript in this case). Finally, we indicate the *total variation distance* between two kernels using $\|\cdot\|$.

Our first result shows that the distance $\|P_i \circ D - P'_i \circ D\| \leq 4\zeta\ell$ for all distributions D.

Lemma 3. *If $|X_\xi| < \zeta$ and $|\nabla F_X(x)| < \ell$ for all x, then $\|P_i \circ D - P'_i \circ D\| \leq 4\zeta\ell \;\; \forall D$.*

Thus, the accuracy of our transition kernel P'_i is related to the quality of our estimated X_{norm}. We next show that $\text{Var}(\Delta')$ is roughly proportional to the step size of our proposer.

Lemma 4. *For one step sampling from θ_t to θ', if the jumping step $\|\theta_t - \theta'\|_2 < \epsilon$, and the gradient of the log likelihood is bounded by a constant factor, i.e. $\|\nabla(\log \Pr(x_i \mid \theta))\|_2 < k$, the variance of Δ' is bounded by $4\epsilon^2 k^2(m - \frac{m(m-1)}{N-1})$, where m is the minibatch size, and N is the total data size.*

Since $\text{Var}(\Delta') < 4\epsilon^2 k^2(m - \frac{m(m-1)}{N-1})$, we can assume the maximum value of X_ξ is proportional to the standard deviation of Δ', i.e. ϵ. Thus, we can assume $\zeta = \epsilon C$, where C is a constant factor. We finally introduce Theorem 1 to characterize the convergence of our samples.

Theorem 1. *Assume π satisfies $\|P_i \circ D_0 - \pi\| \leq \eta\|D_0 - \pi\|$, where $\eta \in [0, 1)$ for all distributions D_0. Also assume there exists $0 < \rho_t < 1$ such that $\|P'_i \circ D - \pi'_i\| \leq 2\rho_i \;\; \forall D$. We can get $\|P'_t \circ P'_{t-1} \circ \cdots \circ P'_1 \circ D_0 - \pi\| \leq \sum_{s=1}^{t}\{\prod_{u=s+1}^{t} \rho_u(1 - \alpha_u)\}\rho_s\alpha_s + \alpha_t$, where $\alpha_i = \frac{\epsilon_i C}{1-\eta}$ and ϵ_i is the step size at iteration i.*

Theorem 1 indicates that the difference between the target and sampled distributions consists of two components: ρ_i, which is determined by the efficiency of P'_i, and the α_is, which are determined by the error of our transition kernels. These involve competing objectives: the larger the step size, the more efficient the P'_i but its error increases. The reverse happens with a step size too small.

5 Experiments

We conduct three sets of experiments to explore the benefits of our minibatch MH test and to benchmark it with previous work. In Section 5.1, we show that our test enables samples to converge to the posterior distribution of a heated Gaussian mixture model. In Section 5.2, we analyze its efficiency on logistic regression. In Section 5.3, we apply the test for deep learning. Appendices B, C, and D contain more detailed information on these respective experiments.

5.1 Mixture of Gaussians

We start with a simple Gaussian mixture model, borrowing an experiment from [11]. The parameter is 2-D, $\theta = (\theta_1, \theta_2)$, and the parameter/data generation process is

$$(\theta_1, \theta_2) \sim \mathcal{N}((0, 0), \text{diag}(\sigma_1^2, \sigma_2^2)); \qquad x_i \sim 0.5 \cdot \mathcal{N}(\theta_1, \sigma_x^2) + 0.5 \cdot \mathcal{N}(\theta_1 + \theta_2, \sigma_x^2). \tag{9}$$

We set $\sigma_1^2 = 10, \sigma_2^2 = 1$ and $\sigma_x^2 = 2$. Fixing $\theta = (0, 1)$, we draw 10000 data points so that the target distribution is $p(\theta)\prod_{i=1}^{10000} p(x_i \mid \theta)$, with the prior based on the θ generation process in Equation 9. This results in rather sharp posterior modes and high $\text{std}(\Delta')$ estimates, so we apply a temperature $T = 110$ to reduce $\text{std}(\Delta')$. Taking logs, we get the target as shown in the far left of Figure 2.

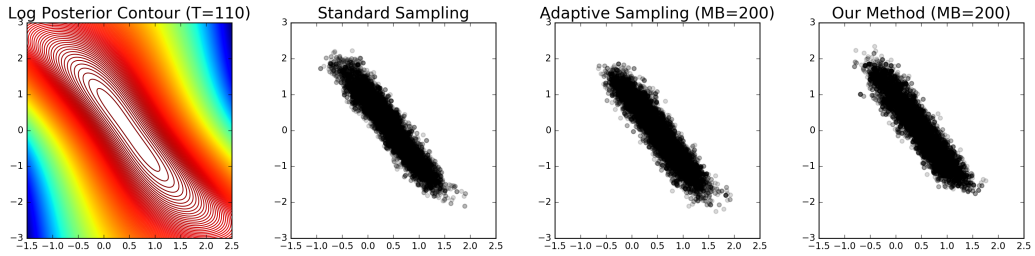

Figure 2: The log posterior contours (temperature 110) and three scatter plots of sampled θ values.

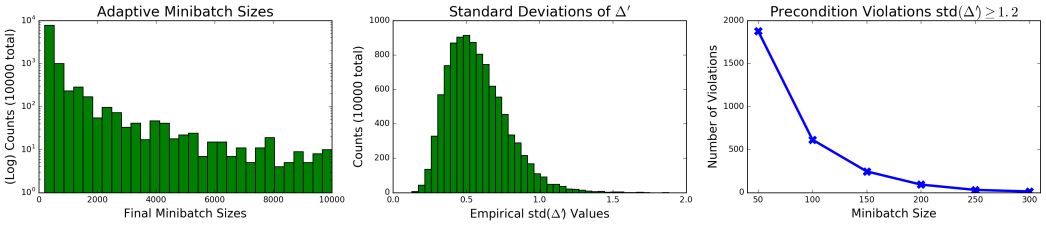

Figure 3: Histograms of minibatch sizes and $\mathrm{std}(\Delta')$ values, along with "$\mathrm{std}(\Delta')$ violations" (right).

We run MCMC with our MH test using minibatch size $m = 200$. We also run this using standard full-batch MCMC ("standard sampling") with the standard test from Equation 1, and the method from [3] ("adaptive sampling"). For the latter, we also use $m = 200$ and increment minibatches by that amount within an iteration if necessary. The tolerance for making a decision is $\epsilon = 0.1$. To make comparisons easier, all three use the same random walk proposer with covariance $\Sigma = \mathrm{diag}(0.03, 0.03)$. This is a poor proposer, but it is sufficient for our purposes as the quality of the samples will be primarily due to the MH test. All methods are run 10000 times to collect 10000 samples.

Figure 2 shows scatter plots of the resulting θ samples for the three methods, with darker regions indicating a greater density of points. All three methods obtain the same rough form of the posterior, so our MH test can indeed result in the same posterior as the other two methods.

Figure 3 (to the left) shows a histogram of the final minibatch sizes used by the adaptive subsampling method each iteration (on a log scale for readability). Usually, the method can make a decision with 200 samples. Occasionally, however, it must use all 10000 points, resulting in lots of wasted computation. For this particular experiment, that happened eight times. In contrast, our method keeps the minibatch size fixed at 200, but requires that $\mathrm{std}(\Delta') < 1.2$. Figure 3 (in the middle) shows a histogram of the estimated $\mathrm{std}(\Delta')$ values each iteration. Only 75 iterations (out of 10000) resulted in $\mathrm{std}(\Delta') \geq 1.2$ and for these, we skipped the rest of the iteration (setting $\theta_{t+1} = \theta_t$). To smooth the $\mathrm{std}(\Delta')$ values, we use moving average updates. The third plot in Figure 3 investigates the number of times $\mathrm{std}(\Delta') \geq 1.2$ based on minibatch sizes. For six sizes (50, 100, 150, 200, 250, and 300), we ran MCMC with our MH test five times and averaged the number of "violations." As the size increases, the number of violations decreases, as expected.

5.2 Logistic Regression

We next use logistic regression for the binary classification of 1s versus 7s in the MNIST dataset [14]. The data has 12007 and 1000 training and testing points, respectively (we used a random subset of the test data). The proposer is again a random walk with covariance matrix $0.1I$ for the 784×784 identity matrix I. We set the posterior temperature at $T = 3000$. We set the minibatch size $m = 50$ and compare with adaptive sampling with tolerance 0.02.

Figure 4 shows the prediction accuracy and log likelihood on the test set as a function of the cumulative training data points processed. Our test increments the cumulative data by a fixed amount per iteration, but the adaptive method may require more data per iteration. We see that our minibatch MH test is more efficient; it has similar or better performance while consuming fewer data points.

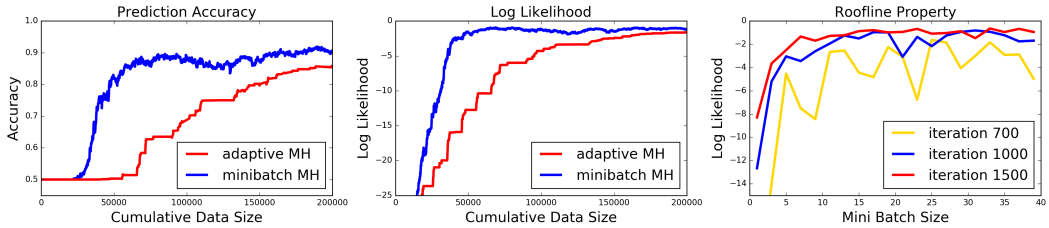

Figure 4: Logistic regression performance (accuracy/log likelihood) and minibatch size analysis.

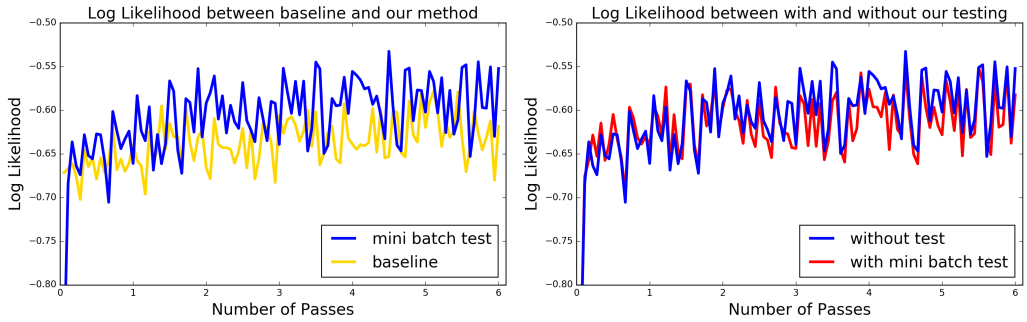

Figure 5: Log likelihood versus number of passes for optimization of neural network.

The third plot of Figure 4 shows performance based on minibatch size. The parameter is evaluated on the test set after training iterations 700, 1000, and 1500. For those three cases, when the minibatch size increases beyond about 10, the performance of the sampled data will not improve with a large minibatch size. This makes sense because the error of our MH test is proportional to our estimation error of $\mathrm{std}(\Delta')$, and as the minibatch increases, $\mathrm{std}(\Delta')$ decreases. Beyond a certain standard deviation (or minibatch) threshold, the error in our minibatch MH test is negligible.

5.3 Neural Network Optimization

In this experiment, we apply our minibatch MH test with a Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) [9] proposer to sample the complete parameters for a fully connected neural network. We use the Higgs data set from the UCI dataset [15], which is a binary classification task with one million instances, each of which are 28-dimensional. The network we use has four layers: the first is the input, the last is the softmax, and the intermediate ones apply the sigmoid activation unit. We also employ dropout and batch normalization [16]. In order to control $\mathrm{std}(\Delta')$, we initialize the temperature at 1000, and adjust it at iteration i according to $T_i = \max\{1, 1000/(i+1)^{0.5}\}$.

For our comparison baseline, we train the same network architecture using the Adaptive Gradient Descent method [17]. Since the frequencies of the features are quite different, we use the same strategy as in Adaptive Gradient to rescale the gradient before applying SGHMC. We implement our MH test and SGHMCin the open-source BIDMach project [18].

Figure 5 illustrates our experiment results and reveals that SGHMC with our minibatch MH test achieves higher log likelihood than the baseline. By comparing SGHMC with and without our minibatch MH test, we see that our MH test has limited benefit. Because we decrease the step size quickly, the acceptance rate of SGHMC is already high, so there is little need to test.

6 Conclusions

In this paper, we have derived a new MH test for minibatch MCMC methods. We demonstrated how a simple deconvolution process allows us to use a minibatch approximation to the full data tests. We experimentally show the benefits of our test on Gaussian mixtures, logistic regression, and deep learning. Straightforward directions for future work include running more experiments with a particular focus on investigation of the variance precondition. More elaborate extensions include combining our results with Hamiltonian Monte Carlo methods, providing a recipe for how to use our algorithm (following the framework of [19]), or integrating parallel MCMC [20, 21] concepts.

References

[1] 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*, vol. 21, 1953.

[2] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[3] A. K. Balan, Y. Chen, and M. Welling, "Austerity in MCMC land: Cutting the metropolis-hastings budget," in *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[4] W. Gilks and D. Spiegelhalter, *Markov chain Monte Carlo in practice*. Chapman & Hall/CRC, 1996.

[5] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

[6] R. Bardenet, A. Doucet, and C. Holmes, "Towards scaling up markov chain monte carlo: an adaptive subsampling approach," in *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[7] D. Maclaurin and R. P. Adams, "Firefly monte carlo: Exact MCMC with subsets of data," in *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, (UAI)*, 2014.

[8] R. M. Neal, "MCMC using Hamiltonian dynamics," *Handbook of Markov Chain Monte Carlo*, vol. 54, pp. 113–162, 2010.

[9] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient Hamiltonian Monte Carlo," in *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[10] N. Ding, Y. Fang, R. Babbush, C. Chen, R. D. Skeel, and H. Neven, "Bayesian sampling using stochastic gradient thermostats," in *Advances in Neural Information Processing Systems 27*, 2014.

[11] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

[12] S. Ahn, A. K. Balan, and M. Welling, "Bayesian posterior sampling via stochastic gradient fisher scoring.," in *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

[13] S. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability*. Cambridge University Press, 2009.

[14] Y. LeCun and C. Cortes, "MNIST handwritten digit database,"

[15] M. Lichman, "UCI machine learning repository," 2013.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015.

[17] P. Bartleet, E. Hazan, and A. Rakhlin, "Adaptive online gradient descent," in *Advances in Neural Information Processing Systems 20*, 2007.

[18] J. Canny and H. Zhao, "Bidmach: Large-scale learning with zero memory allocation," *BIGLearn NIPS Workshop*, 2013.

[19] Y. Ma, T. Chen, and E. Fox, "A complete recipe for stochastic gradient mcmc," in *Advances in Neural Information Processing Systems 28*, 2015.

[20] E. Angelino, E. Kohler, A. Waterland, M. Seltzer, and R. P. Adams, "Accelerating MCMC via parallel predictive prefetching," in *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, (UAI)*, 2014.

[21] S. Ahn, B. Shahbaba, and M. Welling, "Distributed stochastic gradient MCMC," in *Proceedings of the 31st International Conference on Machine Learning, (ICML)*, 2014.

[22] Y. Yang and D. B. Dunson, "Sequential markov chain monte carlo," *arXiv preprint arXiv:1308.3861*, 2013.

Outline of Appendix

In this appendix, we describe the following topics, with an emphasis on clarity and understanding:

- Proofs for material in Section 4.
- More information on Section 5.1.
- More information on Section 5.2.
- More information on Section 5.3.

A Proofs

A.1 Proof of Lemma 3

We first prove that the difference in our acceptance rates is bounded: $|P_a - P_a'| \leq 2\zeta\ell$ for some small constant $\zeta > 0$. Given $X_\xi = \xi$ the distance $|P_a - P_a'|$ is:

$$\begin{aligned}
|P_a - P_a'| &= |\Pr(\Delta + X > 0) - \Pr(\Delta' + X_{\text{corr}} > 0)| \\
&= |\Pr(\Delta + X > 0) - \Pr(\Delta + X_{\text{norm}} + \xi + X_{\text{corr}} > 0)| \\
&= |\Pr(\Delta + X > 0) - \Pr(\Delta + X + \xi > 0)| \\
&= |F_X(-\Delta) - F_X(-\Delta - \xi)| \\
&= |\nabla F(-\Delta)\xi + o(\xi^2)| \\
&\leq |\nabla F(-\Delta)\xi| + |o(\xi^2)| \leq 2|\nabla F(-\Delta)||\xi| \leq 2\ell\xi,
\end{aligned}$$

where we use Taylor's Theorem ($o(\xi^2)$ represents higher-order terms that have smaller absolute value) and the Triangle Inequality. Since $|X_\xi| < \zeta$, $|P_a - P_a'| < 2\zeta\ell$.

Next, we show the distance of new distributions obtained by applying P_i and P_i' on an arbitrary distribution D once is bounded if we use the same proposer, i.e. $\|P_i \circ D - P_i' \circ D\| \leq 4\zeta\ell \ \ \forall D$. We write $P_i(\theta_t \mid \theta') = P_a(\theta_t, \theta')q(\theta' \mid \theta_t) + (1 - P_a(\theta_t, \theta'))\delta_D(\theta' - \theta_t)$, where δ_D is the Dirac delta function. The total variation distance between two generated distributions is

$$\begin{aligned}
\|P_i \circ D - P_i' \circ D\| &= \int_{\theta'} \left| \int_{\theta_t} ((P_a - P_a')q(\theta'|\theta_t) + (1 - P_a - 1 + P_a')\delta_D(\theta_t - \theta'))dD(\theta_t) \right| d\theta' \\
&= \int_{\theta'} \left| \int_{\theta_t} (P_a - P_a')(q(\theta'|\theta_t) - \delta_D(\theta_t - \theta'))dD(\theta_t) \right| d\theta' \\
&\leq 2\zeta\ell \int_{\theta'} \left| \int_{\theta_t} (q(\theta'|\theta_t) + \delta_D(\theta_t - \theta'))dD(\theta_t) \right| d\theta' \\
&\leq 2\zeta\ell \int_{\theta'} (D'(\theta') + D(\theta'))d\theta' \\
&= 4\zeta\ell,
\end{aligned}$$

where D' is the distribution generated if we always accept the proposed point. (In the last line, we integrate two probability density functions, resulting in a value of two.) This matches our desired bound.

A.2 Proof of Lemma 4

From Lemma 2, only the $\sum_{i=1}^{n}(\log p(x_i \mid \theta') - \log p(x_i \mid \theta_t))$ term brings randomness into Δ'. To simplify the subsequent notation, we define $g_i(\theta) = \log p(x_i \mid \theta)$, and $q_i = g_i(\theta') - g_i(\theta_t)$. Then by Taylor's Theorem:

$$|q| = |g(\theta') - g(\theta_t)| = |\nabla g(\theta_t)^T(\theta' - \theta_t) + o(\theta' - \theta_t)^2| \leq 2\epsilon k.$$

Thus, given θ and θ', the variance of Δ' can be written as:

$$\begin{aligned}\mathrm{Var}(\Delta') &= \mathrm{Var}\left[\sum_{i=1}^{m} g_i(\theta') - g_i(\theta_t)\right] = \mathrm{Var}\left(\sum_{i=1}^{m} q_i\right) \\ &= m\mathrm{Var}(q_i) + m(m-1)\mathrm{Cov}_{i\neq j}(q_i, q_j) \\ &= m\mathrm{Var}(q_i) - \frac{m(m-1)}{N-1}\mathrm{Var}(q_i) \\ &\leq 4\left(m - \frac{m(m-1)}{N-1}\right)k^2\epsilon^2,\end{aligned}$$

as desired.

A.3 Proof of Theorem 1

First, by Theorem 1 in [3], $\|P_i \circ D_0 - \pi\| \leq \eta\|D_0 - \pi\|$ and $\|P_i \circ D - P_i' \circ D\| \leq \epsilon_i C$ for all D. The norm between the stationary distributions is $\|\pi - \pi_i'\| \leq \frac{\epsilon_i C}{1-\eta}$.

Second, by Theorem 3.6 in [22], we have

$$\|P_t' \circ \cdots \circ P_1' \circ D_0 - \pi_t\| \leq \sum_{s=1}^{t}\left\{\prod_{u=s+1}^{t}\rho_u(1-\alpha_u)\right\}\rho_s\alpha_s,$$

where $\alpha = \frac{\epsilon_i C}{1-\eta}$. Thus, we get:

$$\begin{aligned}\|P_t' \circ \cdots \circ P_1' \circ D_0 - \pi_0\| &\leq \|P_t' \circ \cdots \circ P_1' \circ D_0 - \pi_t\| + \|\pi_t - \pi_0\| \\ &\leq \sum_{s=1}^{t}\left\{\prod_{u=s+1}^{t}\rho_u(1-\alpha_u)\right\}\rho_s\alpha_s + \alpha_t,\end{aligned}$$

as desired.

B Gaussian Mixture Experiment Details

In this section, we go over the math details on the Gaussian mixture model problem borrowed from [11]. Our parameter is a 2-D vector $\theta = (\theta_1, \theta_2)$, where

$$\theta_1 \sim \mathcal{N}(0, \sigma_1^2) \quad \text{and} \quad \theta_2 \sim \mathcal{N}(0, \sigma_2^2) \tag{10}$$

where $\mathcal{N}$ indicates the normal distribution (more generally, the multivariate normal). We consider the above as our prior. Following [11], we set $\sigma_1^2 = 10$ and $\sigma_2^2 = 1$, so the covariance matrix of θ is $\Sigma = \mathrm{diag}(10, 1)$. Therefore, the log prior probability we endow on θ is

$$\log p(\theta) = \log\left(\frac{1}{2\pi\sqrt{10}}\right) - \frac{1}{2}\theta^T\Sigma^{-1}\theta. \tag{11}$$

To generate the data, we use the following Gaussian mixture with tied means:

$$x_i \sim \frac{1}{2}\mathcal{N}(\theta_1, \sigma_x^2) + \frac{1}{2}\mathcal{N}(\theta_1 + \theta_2, \sigma_x^2) \tag{12}$$

where, again following [11], we set $\sigma_x^2 = 2$. This means the log likelihood of a single data instance is

$$\log p(x_i \mid \theta) = \log\left(\frac{1}{4\sqrt{\pi}}\right) + \log\left(\exp\left(-\frac{1}{4}(x_i - \theta_1)^2\right) + \exp\left(-\frac{1}{4}(x_i - (\theta_1 + \theta_2))^2\right)\right) \tag{13}$$

Here is the problem statement: given some number of conditionally independent data points $x_1, x_2, \ldots, x_N$ generated according to (12), determine the posterior distribution of θ:

$$\log p(\theta \mid x_1, \ldots, x_N) = \log p(\theta) + \sum_{i=1}^{N}\log p(x_i \mid \theta). \tag{14}$$

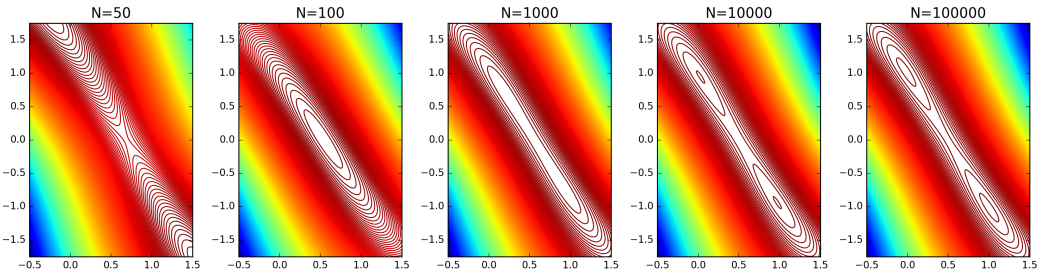

Figure 6: The posterior distribution, from 50 to 100k samples, with temperature set at 1.

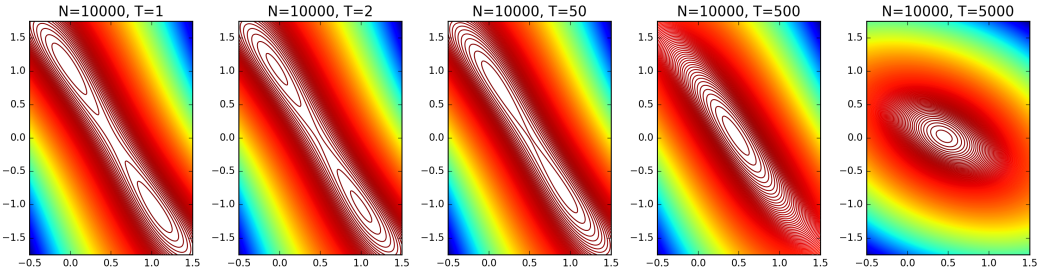

Figure 7: The posterior distribution, with $N = 10000$ but with temperature T varying.

Alternatively, if there are too many data points, we may opt to instead pick a point estimate of θ, generally the MAP estimate. (If N is extremely large, it will cause the posterior to peak sharply at its modes, reducing distribution estimates to point estimates.) Note that in many cases, we will need to take a *minibatch estimate* of (14). In that case, the literature generally uses

$$\log p(\theta \mid x_1, \ldots, x_N) \approx \log p(\theta) + \frac{N}{n} \sum_{i=1}^{n} \log p(x_i \mid \theta). \tag{15}$$

where we only use $n \ll N$ samples, but we must scale up the likelihood contribution by N/n. If we didn't add this scaling factor, then the contribution of the likelihood terms would be weaker.

One technique we use is adding a *temperature* to our distribution. In general, we will want to add $T > 1$ so that our posterior is $p(\theta)((\prod_{i=1}^{n} p(x_i \mid \theta))^{N/n})^{1/T}$, resulting in the log posterior of

$$\log p(\theta \mid x_1, \ldots, x_N) \approx \log p(\theta) + \frac{1}{T} \frac{N}{n} \sum_{i=1}^{n} \log p(x_i \mid \theta). \tag{16}$$

which has the extra $1/T$ to decrease the scale factor. Equation (16) is what we use for our experiments, because warmer distributions help us satisfy our $\mathrm{std}(\Delta') < 1.2$ requirement.

To gain some intuition on what the posterior looks like, Figure 6 shows simulated contour plots of the posterior based on varying numbers of data points N, with the temperature set at $T = 1$. Note that because we are using all N points here, the scale factor $N/n = 1$. As N increases, the posterior converges to a multimodal distribution with modes at $(0, 1)$ and $(1, -1)$. Figure 7 is similar, except this time we fix the number of samples at $N = 10000$, but show how changing the temperature T affects the distribution. A larger T implies a flatter posterior, one that (weakly) peaks in between the two true modes.

C Logistic Regression Experiment Details

In this section, we go over some details of our logistic regression experiment. The feature vector for an image consists of its pixel values, normalized between 0 and 1. For simplicity, we only consider the binary classification case, so we only use digits one (denoted as output $y = 1$) and seven (denoted

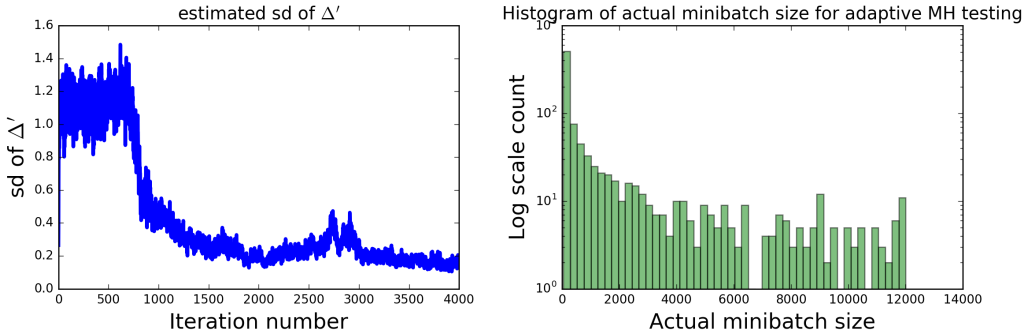

Figure 8: Additional results for the logistic regression experiment.

as $y = -1$). The probability of the i^{th} output $y_i \in \{-1, +1\}$ with the input vector x_i is

$$p(y_i \mid x_i) = \frac{1}{1 + \exp(-y_i \theta^T x_i)}, \tag{17}$$

where θ is the 784-length parameter vector.

For our experiment, we impose a uniform prior to represent our lack of knowledge about θ. We use a random walk proposer, which can be modeled as $\theta' = \theta_i + \mathcal{N}(0, \sigma^2 I)$, where θ_i is the current sample, θ' is the proposed sample, and we choose the variance to be a constant $\sigma^2 = 0.01$ for all components. We initialize θ_0 to be a vector of all ones, and set our minibatch size as $m = 50$.

For our minibatch MH test, in order to enforce the $\text{std}(\Delta') < 1.2$ condition, we use a constant temperature $T = 3000$. If our estimated $\text{std}(\Delta') \geq 1.2$, we ignore the current iteration. Figure 8 plots our estimated $\text{std}(\Delta')$ values versus iteration count.

For adaptive MH testing, our experimental settings are the same as with our MH test, except we do not impose a temperature. The minibatch size of adaptive MH testing is also initialized as 50, but it may increase by that amount each iteration. Figure 8 shows the histogram of the actual minibatch size at the end of each iteration in adaptive MH testing.

D Neural Network Experiment Details

For our neural network experiment, we use the architecture discussed in Section 5.3. We use the SGHMC as the proposer for our minibatch MH testing. For the baseline, we use a tuned adaptive gradient descent optimizer, whose step size changes by $0.01/(i+1)^{0.4}$. Both methods have minibatch sizes set at $m = 200$. There are one million total data instances x_i.

For SGHMC, we use the simplified update equations [9]:

$$\Delta\theta = v, \qquad \Delta v = -\eta \nabla U(\theta) - \alpha v + \mathcal{N}(0, 2(\alpha - \hat{\beta})\eta), \tag{18}$$

where v represents auxiliary momentum variables, and $\nabla U(\theta)$ is the gradient of the system. We set the hyperparameters to be $\eta_i = 0.01/(i+1)^{0.4}$, and $\alpha = 0.1$. We use the empirical Fisher [12] information $V(\theta)$ to estimate the value of $\hat{\beta}$, so that $\hat{\beta} = \frac{1}{2}\eta V(\theta)$. In order to control $\text{std}(\Delta')$, we initialize the temperature at 1000, and adjust it at iteration i according to $T_i = \max\{1, 1000/(i+1)^{0.5}\}$.