An Efficient Minibatch Acceptance Test for Metropolis-Hastings

John Canny, Haoyu Chen, Daniel Seita, Xinlei Pan
University of California, Berkeley
{canny,haoyuchen,seita,xinleipan}@berkeley.edu

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. Previous work on reducing the cost of Metropolis-Hastings tests yield variable data consumed per sample, with only constant factor reductions versus using the full dataset for each sample. Here we present a method that can be tuned to provide arbitrarily small batch sizes, by adjusting either proposal step size or temperature. Our approach uses the natural noise present in minibatch likelihood estimates to furnish the randomness in a Metropolis-Hastings test. Our test uses the noise-tolerant Barker acceptance test with a novel additive correction variable. The resulting test can be combined with minibatch proposals to yield updates with the same complexity as a simple SGD update. In this paper we derive the test, analyze its performance, discuss its implementation, and present several experiments.

1 Introduction

Markov chain Monte Carlo (MCMC) sampling is a powerful method for computation on intractable distributions. We are interested primarily in large dataset applications, where the goal is to sample a posterior distribution $p(\theta \mid x_1, \ldots, x_N)$ of parameter θ, and where the number of data instances N is large. The Metropolis-Hastings method (M-H) generates sample candidates from a proposal distribution q which is in general different from the target distribution p, and decides whether to accept or reject them based on an acceptance test. The acceptance test is usually a Metropolis test [1, 2]. Conventional Metropolis-Hastings requires all N data instances to generate one posterior sample.

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 have high sample efficiency. They also map very well onto hardware such as GPUs. In contrast, M-H requires calculations over the full dataset to produce a new sample. Recent results from [3, 4] perform approximate (bounded error) acceptance tests using subsets (minibatches) of the full dataset. The tests depend on minibatch statistics, and on the value of an additional uniform random variable u. The amount of data consumed for each test varies significantly from one minibatch to the next, and depends on the current sample, the proposed sample, and on the random variable u. By contrast, [5] performs exact tests but requires a lower bound on parameter distribution across its domain. The amount of data reduction depends on the accuracy of this bound, and such bounds are only available for relatively simple distributions.

Here we derive a new test which incorporates the variability in minibatch statistics as *a natural part of the test*. Because of this, the amount of data required for each test is fixed with high probability and in expected value. We use a Barker test function [6] rather than a Metropolis test, which makes our test naturally error tolerant. The idea of using a noise-tolerant test using Barker's test function was suggested but not explored empirically in [7] section 6.3. But the asymptotic test statistic CDF and the Barker function are different, which leads to fixed errors for the approach in [7]. Here we

show that the difference between the distributions can be corrected with an additive random variable. This leads to a test which is fast, and whose error can be made arbitrarily small.

Our test is applicable when the variance (over data samples) of the log acceptance probability is small enough (less than 1). Its not clear at first why this quantity should be bounded, but we will show that it is "natural" for well-specified models running Metropolis-Hastings sampling with optimal proposals [8] on a full dataset. When we reduce the amount of data for the test, the variance goes up. We have to reduce variance in one of several ways. Either:

- Increase the temperature of the target distribution. Log likelihoods scale as $1/T$, and so the variance of the likelihood ratio will vary as $1/T^2$. Our model is no longer well-specified (we are doing inference at a temperature different from that assumed during data generation), but higher temperature can be advantageous for parameter exploration.
- For continuous probability distributions, reduce the proposal step size and variance (for stochastic proposals) compared to an optimal proposal. The variance of the log acceptance probability scales as the square of proposal step size.
- Increase the minibatch size when needed for certain minibatches. Log acceptance variance scales as $1/k$ vs the minibatch size k. Our test is adaptive like earlier works, but unlike them, the distribution of minibatch size is gaussian, not long-tailed. Increased minibatch size also reduces the error rate for the test.

Its worth discussing at this point the typical goals of M-H sampling on very large datasets. By the Bernstein-von Mises Theorem, the posterior distribution of the parameter θ for a Bayesian inference task is asymptotically normal, and has variance that scales inversely with the number of data samples N. This mode is extremely sharp for large datasets, which may contain millions or billions of samples. Simply sampling from this distribution is one application, but an efficient proposal distribution [8] has similar variance to the target distribution and will diffuse to it extremely slowly from an initialization value which is (likely to be) many standard deviations away. If there are any other strong modes, it is very likely for the sampler to find one of them and become trapped in it when run at the normal distribution temperature ($T = 1$). A common solution is to anneal the sampler, running first at high temperature (scaling log likelihoods by $1/T$) which flattens the likelihood landscape. This is turn reduces the variance of the log acceptance probability and allows our acceptance test to be applied.

A second question concerns step size. Once we have fixed temperature, our variance constraint implies that we have to trade-off proposal step size s and batch size b ($b \propto p^2$), i.e. we can make many small steps, or one large step, with a given batch of data. One of the primary drivers of this work is our belief in the value of small steps. For applications to neural networks or other models where the posterior is multimodal, posterior inference is arguably a search process. Covering the search space densely with small steps is much more valuable than few sparse steps toward the nearest optimum. In this mode, Metropolis-Hastings can be used in similar fashion to Stochastic Gradient Descent. The goal in SGD is to make gradual progress to a posterior mode with each step, taking small steps so that the cumulative displacement has progressively lower variance. A substantial part of the computational work of MCMC on massive datasets will be similarly in reaching a stationary distribution, which really means finding a deep posterior mode. Taking noisy small steps will nevertheless make steady progress to a posterior mode since their bias is in that direction. We demonstrate this behavior in our logistic regression experiments.

The contributions of this paper are as follows:

- We develop a new, more efficient (in samples per test) minibatch acceptance test with quantifiable error bounds. The test uses a novel additive correction variable to implement a Barker test based on minibatch mean and variance.
- We analyze the test for accuracy and speed.
- We compared performance of our new test and prior approaches on several datasets. We demonstrate the test is several orders of magnitude more efficient than prior work measured as data consumed per test, and that it does not suffer from long-tailed minibatch sizes (up to the dataset size).

2 Preliminaries and Related Work

In the Metropolis-Hastings method [9, 10], a difficult-to-compute probability distribution $p(\theta)$ is sampled using a Markov chain $\theta_1, \ldots, \theta_n$. The sample θ_{t+1} at time $t+1$ is generated using a candidate θ' from a (simpler) proposal distribution $q(\theta' \mid \theta_t)$, filtered by an acceptance test. The acceptance test is usually a Metropolis test. The Metropolis test has acceptance probability:

$$\alpha(\theta_t, \theta') = \frac{p(\theta')q(\theta_t \mid \theta')}{p(\theta_t)q(\theta' \mid \theta_t)} \wedge 1 \tag{1}$$

where $a \wedge b$ denotes $\min(a, b)$. With probability $\alpha(\theta_t, \theta')$, we accept θ' and set $\theta_{t+1} = \theta'$, otherwise set $\theta_{t+1} = \theta_t$. The test is often implemented with an auxiliary random variable $u \sim \mathcal{U}(0, 1)$ with a comparison $u < \alpha(\theta_t, \theta')$; here, $\mathcal{U}(a, b)$ denotes the uniform distribution on the interval $[a, b]$. For notational simplicity, we drop the subscript t for the current sample θ_t and denote it as θ.

The acceptance test guarantees detailed balance, which means $p(\theta)p(\theta' \mid \theta) = p(\theta')p(\theta \mid \theta')$, where $p(\theta' \mid \theta)$ is the probability of a transition from state θ to θ'. Here $p(\theta' \mid \theta) = q(\theta' \mid \theta)\alpha(\theta, \theta')$ so the detailed balance equation becomes

$$p(\theta)q(\theta' \mid \theta)\alpha(\theta, \theta') = p(\theta')q(\theta \mid \theta')\alpha(\theta', \theta) \tag{2}$$

This condition, together with ergodicity, guarantees that the Markov chain has a unique stationary distribution $\pi(\theta) = p(\theta)$.

For Bayesian inference, we would like to sample from a parameter distribution for θ based on some observed data, so the target distribution is $p(\theta \mid x_1, \ldots, x_N)$. The acceptance probability is now:

$$\alpha(\theta, \theta') = \frac{p_0(\theta') \prod_{i=1}^{N} p(x_i \mid \theta')q(\theta \mid \theta')}{p_0(\theta) \prod_{i=1}^{N} p(x_i \mid \theta)q(\theta' \mid \theta)} \wedge 1 \tag{3}$$

where $p_0(\theta)$ is a prior, and $p(x_i \mid \theta)$ are the probabilities of the observations. Computing samples this way requires the use of all N training data points, but this is very expensive for large datasets. To address this challenge, [3, 4] perform approximate Metropolis-Hasting tests using sequential hypothesis testing. During each iteration, they start with a small minibatch of data and test the hypothesis that the sample θ' should be accepted based on an approximate version of the test $u < \alpha(\theta, \theta')$. If the approximate test cannot make a decision with sufficient confidence, then the minibatch size is increased and the test repeats. This process continues until a decision. The bounds depend on either a asymptotic central limit theorem [3] or a concentration bound [4]. The latter requires direct bounds on the log likelihood ratio, which for general distributions requires knowing $p(x_i \mid \theta)$ and $p(x_i \mid \theta')$ for all N samples. In addition, both methods suffer the drawback of resolving small log likelihood ratio differences between the minibatch and full batch versions. In the worst case, all N data points may be needed, and we soon show that this worst case can occur about $\Omega(N)$ times during the performance of N tests.

Following [4], we write the test $u < \alpha(\theta, \theta')$ in the equivalent form $\Lambda(\theta, \theta') > \psi(u, \theta, \theta')$, where[1]

$$\Lambda(\theta, \theta') = \sum_{i=1}^{N} \log\left(\frac{p(x_i|\theta')}{p(x_i|\theta)}\right) \qquad \text{and} \qquad \psi(u, \theta, \theta') = \log\left(u\frac{q(\theta'|\theta)p_0(\theta)}{q(\theta|\theta')p_0(\theta')}\right) \tag{4}$$

To reduce computational effort, an unbiased estimate of $\Lambda(\theta, \theta')$ based on a minibatch can be used:

$$\Lambda^*(\theta, \theta') = \frac{N}{b} \sum_{i=1}^{b} \log\left(\frac{p(x_i|\theta')}{p(x_i|\theta)}\right) \tag{5}$$

Finally, it will be convenient for our analysis to define the individual components that contribute to the sums above:

$$\Lambda_i(\theta, \theta') = N \log\left(\frac{p(x_i|\theta')}{p(x_i|\theta)}\right) \tag{6}$$

Thus, $\Lambda(\theta, \theta')$ is the mean of $\Lambda_i(\theta, \theta')$ over the entire dataset, and $\Lambda^*(\theta, \theta')$ is the mean of $\Lambda_i(\theta, \theta')$ over its minibatch.

Since minibatches contains randomly selected samples x_i, the values Λ_i are independent, identically distributed (iid) random variables[2]. By the Central Limit Theorem, we expect $\Lambda^*(\theta, \theta')$ to be

[1] Our definitions differ from those in [4] by a factor of N to simplify our analysis later.

[2] The analysis assumes sampling with replacement although implementations on typical large datasets will approximate this by sampling without replacement

approximately Gaussian. The acceptance test then becomes a statistical test of the hypothesis that $\Lambda(\theta, \theta') > \psi(u, \theta, \theta')$ by establishing that $\Lambda^*(\theta, \theta')$ is substantially larger than $\psi(u, \theta, \theta')$. In [3] an asymptotic central limit argument was used to derive this gap, while in [4] a concentration bound was used. In both cases, the resulting tests were shown to give useful reductions in number of samples required over using the full dataset, but there were no worst-case bounds given on the average number of samples needed per iteration.

We next show that for a simple distribution, the lower bound of average number of data instances consumed for one iteration of [3] and [4] is $\Omega(N)$, where N is the number of data points.

2.1 A Worst-Case Gaussian Example

Consider a Gaussian model where $x_1, \ldots, x_N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\theta, 1)$ with a known variance $\sigma^2 = 1$, true mean $\theta = 0.5$, and a uniform prior on θ. The log likelihood ratio is

$$\Lambda^*(\theta, \theta') = \frac{N}{b} \sum_{i=1}^{b} \log \frac{p(x_i|\theta')}{p(x_i|\theta)} = N(\theta' - \theta) \left(\frac{1}{b} \sum_{i=1}^{b} x_i - \theta - \frac{\theta' - \theta}{2} \right) \tag{7}$$

which is normally distributed over selection of the Normal samples x_i. Since the x_i have unit variance, their mean has variance $1/b$, and the variance of $\Lambda^*(\theta, \theta')$ is $\sigma^2(\Lambda^*) = (\theta' - \theta)^2 N^2 / b$. In order to pass a hypothesis test that $\Lambda > \psi$, there needs to be a large enough gap (several $\sigma(\Lambda^*)$) between $\Lambda^*(\theta', \theta)$ and $\psi(u, \theta', \theta)$. Daniel: I changed the variance to be $(\theta' - \theta)^2 N^2 / b$; before it had N instead of N^2.

Our goal is to sample from the posterior $p(\theta \mid x_1, \ldots, x_N)$, which is a normal distribution $\mathcal{N}(\mu, 1/N)$ centered on the sample mean μ, and with variance $1/N$. In one dimension, an efficient proposal distribution has the same variance as the target distribution [8], so we use the proposal $\mathcal{N}(\theta' - \theta, 1/N)$, which is implemented as $q(\theta' \mid \theta) = \phi((\theta' - \theta)N)$, where $\phi(x)$ is the Normal density function with zero mean and unit variance. This proposal is symmetric $q(\theta' \mid \theta) = q(\theta \mid \theta')$, and since we assumed a uniform prior on θ, we see that $\psi(u, \theta', \theta)$ reduces to $\log u$. Our worst-case scenario is specified in Lemma 1. Daniel: this might be a silly question, but why do we assume the proposal has mean $\theta' - \theta$? I thought it would have a mean of θ because that is the current sample and the Gaussian is centered at that point? That will also clarify my confusion about the ψ function.

Lemma 1. *For the model in Section 2.1, there exists a fixed (independent of N) constant c such that with probability $\geq c$ over the joint distribution of (θ, θ', u), the tests from [3, 4] consume all N samples.*

Proof. See Appendix A. □

Similar results can be shown for other distributions and proposals by identifying regions in product space $(\theta, \theta' - \theta, u)$ such that the hypothesis test needs to separate nearly-equal values. It follows that the accelerated M-H tests in [3, 4] require at least a constant fraction $\geq c$ in the amount of data consumed per test compared to full-dataset tests, so their speed-up is at most $1/c$.

Daniel: quick question: I think we should explain why our method *doesn't* run into this issue. I think it is because we rely on an approximation to $\Delta + X_{\log} > 0$ and that test does not require any "difference between an approximation and an exact version" because it is *already* an exact version, assuming the approximation is enough.

Also, in terms of this example, if we are in the special case with using a lot of data, why not force early stopping?

2.2 MCMC Posterior Inference

There is a separate line of MCMC work drawing principles from statistical physics. By viewing random variables as particles in a system, one can apply Hamiltonian Monte Carlo (HMC) [11] methods which generate high acceptance *and* distant proposals when run on full batches of data. Recently Langevin Dynamics [12, 13] has been applied to Bayesian estimation on minibatches of data. This simplified dynamics uses local proposals and avoids MH tests by using small proposal steps whose acceptance approaches 1 in the limit. However, the constraint on proposal step size is severe,

and the state space exploration reduces to a random walk. Full minibatch HMC for minibatches was recently described in [14] which allows momentum-augmented proposals with larger step sizes. However, step sizes are still limited by the need to run accurately without MH tests. Our work opens the door to applying those methods with much more aggressive step sizes without loss of accuracy. We demonstrate this in Section ... Daniel: do we ever demonstrate this? I guess another issue is that this implies we can use larger step sizes, but earlier we *also* argue that we need *small* step sizes (to reduce variance).

3 A New Metropolis-Hastings Acceptance Test

3.1 Log-Likelihood Ratios

For our new M-H test, we denote the exact and approximate log likelihood ratios as Δ and Δ^*. First, Δ is defined as

$$\Delta(\theta, \theta') = \log \frac{p_0(\theta') \prod_{i=1}^N p(x_i \mid \theta') q(\theta \mid \theta')}{p_0(\theta) \prod_{i=1}^N p(x_i \mid \theta) q(\theta' \mid \theta)}, \tag{8}$$

where p_0, p, and q match the corresponding functions within Equation 3. We separate out terms dependent and independent of the data x as:

$$\Delta(\theta, \theta') = \sum_{i=1}^N \log \frac{p(x_i \mid \theta')}{p(x_i \mid \theta)} - \psi(1, \theta, \theta') = \Lambda(\theta, \theta') - \psi(1, \theta, \theta'). \tag{9}$$

A minibatch estimator of Δ, denoted as Δ^*, is

$$\Delta^*(\theta, \theta') = \frac{N}{b} \sum_{i=1}^b \log \frac{p(x_i \mid \theta')}{p(x_i \mid \theta)} - \psi(1, \theta, \theta') = \Lambda^*(\theta, \theta') - \psi(1, \theta, \theta') \tag{10}$$

Note that Δ and Δ^* are evaluated on the full dataset and a minibatch of size b respectively. The scaling term N/b ensures that $\Delta^*(\theta, \theta')$ is an unbiased estimator of $\Delta(\theta, \theta')$. Daniel: a brief question repeated from earlier, should we mention sampling with and/or without replacement here?

The key to our test is a smooth acceptance function. We consider functions other than the classical Metropolis test that satisfy the detailed balance (see Equation 2) condition needed for accurate posterior estimation. A class of functions leading to detailed balance is specified as follows:

Lemma 2. *If $g(s)$ is any function such that $g(s) = \exp(s) g(-s)$, then the acceptance function $\alpha(\theta, \theta') \triangleq g(\Delta(\theta, \theta'))$ satisfies detailed balance.*

This result is used in [6] to define the Barker acceptance test. As a sanity check, choosing $g(s) = \exp(s) \wedge 1$ — a function satisfying the requirement of Lemma 2 — produces the classical Metropolis acceptance test $\alpha(\theta, \theta') = g(\Delta(\theta, \theta')) = \frac{p(\theta') q(\theta \mid \theta')}{p(\theta) q(\theta' \mid \theta)} \wedge 1$. In fact, $g(s) = \exp(s) \wedge 1$ is the optimal acceptance function in terms of acceptance rate, since it accepts with probability 1 for $\Delta > 0$.

3.2 Barker (Logistic) Acceptance Function

For our new MH test we use the Barker logistic [6] function: $g(s) = (1 + \exp(-s))^{-1}$. Straightforward arithmetic shows that it satisfies the condition in Lemma 2. While it is slightly less efficient than the Metropolis test when used on the full dataset, we will show that its smoothness allows it to naturally tolerate substantial variance in its input argument. This in turn will lead to a much more efficient test on subsets of data. Daniel: aside from the experiments, where do we show it "naturally tolerates substantial variance" in the input?

Assume we begin with the current sample θ and a candidate sample θ', and that $V \sim \mathcal{U}(0, 1)$ is a uniform random variable. We accept θ' if $g(\Delta(\theta, \theta')) > V$, and reject otherwise. Since $g(s)$ is monotonically increasing, its inverse $g^{-1}(s)$ is well-defined and unique. So an equivalent test is to accept θ' iff

$$\Delta(\theta, \theta') > X = g^{-1}(V) \tag{11}$$

where X is a random variable with the logistic distribution (its CDF is the logistic function). To see this notice that $\frac{dV}{dX} = g'$, that g' is the density corresponding to a logistic CDF, and finally that $\frac{dV}{dX}$ is

the density of X. The density of X is symmetric, so we can equivalently test whether

$$\Delta(\theta, \theta') + X > 0 \tag{12}$$

for a logistic random variable X. Daniel: I think the text before Equation 12 should be similar to what we had in our NIPS submission because I find it much clearer.

3.3 A Minibatch Acceptance Test

We now describe acceptance testing using the minibatch estimator $\Delta^*(\theta, \theta')$. From Equation 10, $\Delta^*(\theta, \theta')$ can be represented as a constant term plus the mean of b IID terms $\Lambda_i(\theta, \theta')$ of the form $N \log \frac{p(x_i|\theta')}{p(x_i|\theta)}$. As b increases, $\Delta^*(\theta, \theta')$ therefore has a distribution which approaches a normal distribution by the Central Limit Theorem. We now describe this using an asymptotic argument and defer specific bounds between the CDFs of $\Delta^*(\theta, \theta')$ and a Gaussian to Section 5.

In the limit, since Δ^* is normally distributed about its mean Δ, we can write

$$\Delta^* = \Delta + X_{\text{norm}}, \quad X_{\text{norm}} \sim \bar{\mathcal{N}}(0, \sigma^2(\Delta^*)), \tag{13}$$

where $\bar{\mathcal{N}}(0, \sigma^2(\Delta^*))$ denotes a distribution which is approximately normal with variance $\sigma^2(\Delta^*)$. But to perform the test in Equation 12 we want $\Delta + X$ for a logistic random variable X (call it $X_{\log}$ from now on). In [7] it was proposed to use Δ^* in a Barker test anyway and tolerate the fixed error caused by this approximation.

Our approach is to instead decompose $X_{\log}$ as

$$X_{\log} = X_{\text{norm}} + X_{\text{corr}}, \tag{14}$$

where we assume $X_{\text{norm}} \sim \mathcal{N}(0, \sigma^2)$ and that X_{corr} is a zero-mean "correction" variable with density $C_\sigma(X)$. The two variables are added (i.e., their distributions convolve) to form $X_{\log}$. This decomposition requires an appropriate X_{corr} distribution. We show in Section 4 that it is possible to use deconvolution to derive a highly accurate representation of $C_\sigma(X)$. Figure 1 shows three examples of X_{norm} and X_{corr} densities that sum to $X_{\log}$.

Using X_{corr} samples from $C_\sigma(X)$, the acceptance test is now

$$\Delta + X_{\log} = (\Delta + X_{\text{norm}}) + X_{\text{corr}} = \Delta^* + X_{\text{corr}} > 0. \tag{15}$$

Therefore, assuming the variance of Δ^* is small enough, if we have an estimate of Δ^* from the current data minibatch, we test acceptance by adding a random variable X_{corr} and then accept θ' if the result is positive (and reject otherwise).

If $\bar{\mathcal{N}}(0, \sigma^2(\Delta^*))$ is exactly $\mathcal{N}(0, \sigma^2(\Delta^*))$, the above test is exact as well, and as we show in Section 5, if there is a maximum error ϵ between the CDF of $\bar{\mathcal{N}}(0, \sigma^2(\Delta^*))$ and the CDF of $\mathcal{N}(0, \sigma^2(\Delta^*))$, then the acceptance test has an error of at most ϵ relative to the full batch version.

Daniel: this subsection is improved from earlier. My main concern now is that X_{norm} is described as being both approximately Gaussian $\bar{\mathcal{N}}$ and Gaussian $\mathcal{N}$ here. We may want to clear that up; in our NIPS submission we actually used a new variable ε for that and assigned it to be the $\mathcal{N}$ distribution. My current suggestion is to have Δ^* be the $\bar{\mathcal{N}}$ approx. normal distribution (but with mean Δ, of course), and keep X_{norm} as an exact Gaussian.

4 Computing the Correction Distribution

Our proposed test in Equation 15 requires knowing the distribution of the correction variable X_{corr} such that $X_{\text{norm}} + X_{\text{corr}} = X_{\log}$, where $X_{\text{norm}} \sim \mathcal{N}(0, \sigma^2)$ and $X_{\log}$ has a standard logistic CDF, $(1 + \exp(-X))^{-1}$. In Section 5, we show that the accuracy of the test depends on the absolute error between the CDFs of $X_{\text{norm}} + X_{\text{corr}}$ and $X_{\log}$. Consequently, we need to minimize this in our construction of X_{corr}. More formally, let $\Phi_\sigma(X) = \Phi(X/\sigma)$ where Φ is the standard normal CDF[3], $S(X)$ be the logistic function, and $C_\sigma(X)$ be the *density* of the correction X_{corr} distribution. Our goal, based on Equation 14, is to solve the following optimization problem:

$$C_\sigma^* = \arg\min_{C_\sigma} |\Phi_\sigma * C_\sigma - S| \tag{16}$$

[3]Hence, Φ_σ is the CDF of a zero-mean Gaussian with variance σ.

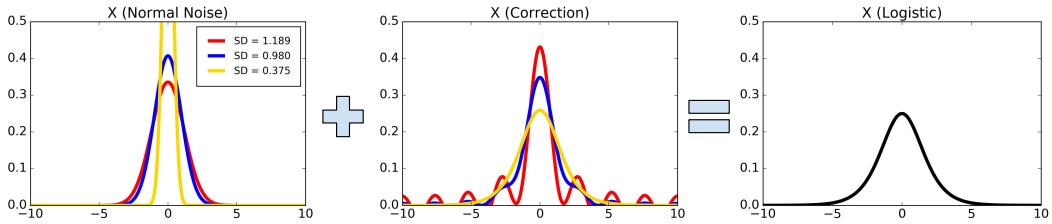

Figure 1: Three examples of X_{norm} and X_{corr} distributions that convolve to form the standard logistic distribution. We use three standard deviation values of X_{norm}. The two red curves convolve to form the logistic, etc. The y-axis is capped at 0.5 for readability. This figure must be viewed in color. Daniel: TODO Change figure to use LaTeX for the titles.

where $*$ denotes convolution. Daniel: This is very unusual use of convolutions (i.e., CDF * DENSITY = CDF). It's valid but I was very surprised to know that. For computation of C_σ, we assume that its input Y and another variable X lie in the intervals $[-V, V]$ and $[-2V, 2V]$, respectively. Daniel: I am a little confused at the different interval sizes. In continuous form, the optimization problem is therefore

$$C_\sigma^* = \arg\min_{C\sigma} \sup_{x\in[-2V,2V]} \left| \int_{-V}^{V} \Phi_\sigma(x-y) C_\sigma(y) dy - S(x) \right| \tag{17}$$

To get C_σ^* in a practical form, we discretize X and Y into $4N+1$ and $2N+1$ values respectively. If $i \in \{-2N, \ldots, 2N\}$ and $j \in \{-N, \ldots, N\}$, then we can write $X_i = i(V/N)$ and $Y_j = j(V/N)$, and the condition in Equation 17 can be re-expressed as:

$$C_\sigma^* = \arg\min_{C_\sigma} \max_{i\in\{-2N,\ldots,2N\}} \left| \sum_{j=-N}^{N} \Phi_\sigma(X_i - Y_j) C_\sigma(Y_j) - S(X_i) \right|. \tag{18}$$

To make the problem easier to understand, we can define a matrix M and vectors u and v such that $M_{ij} = \Phi_\sigma(X_i - Y_j)$, $u_j = C_\sigma(Y_j)$, and $v_i = S(X_i)$, where the indices i and j are appropriately translated to be non-negative indices for M, u, and v. Thus, Equation 18 is equivalent to minimizing the L_1 norm:

$$u^* = \arg\min_u \|Mu - v\|_1. \tag{19}$$

Daniel: Equation 19 is finding the C_σ minimizing the L_1 norm, but the previous one, Equation 18, is finding the C_σ that is minimizing the L_∞ norm, i.e., the maximum value! If we want this to be the L_1 norm (and I think we do) then Equation 18 needs an outer sum over the i terms. In addition, Equation 17 also needs to be modified, perhaps eliminate the "sup" but keep $x \in [-2V, 2V]$

We have the additional constraint that $u_j > 0$ for all j, since u represents a density. We first explored optimizing this problem with linear programming to find a u minimizing $|Mu - v|$ subject to $u \geq 0$. This was tractable up to a few hundred dimensions for u. However, the discretization error is bounded by the curvature of the underlying functions which are here slightly less than one, i.e. the errors are of the order of $(V/N)^2$. Daniel: unfortunately, I do not understand what this means (i.e., the curvature and how the $(V/N)^2$ is obtained). Here we chose $V = 20$ to provide adequate containment of the distributions (the CDFs are extremely close to either zero or one outside this range). So with 200 points, we have a discretization error of approximately 0.01, which is relatively large. To yield higher resolution and lower error, we switched to a least squares solution:

$$u^* = \arg\min_u \ \|Mu - v\|_2^2 + \lambda \|u\|_2^2, \tag{20}$$

where we add the standard regularization factor λ for stability in high dimensions. The solution is well-known from the normal equations: $u^* = (M^T M + \lambda I)^{-1} M^T v$, and in practice, it yields an acceptable L_1 norm for Equation 19.

With this approach, there is no guarantee that $u^* \geq 0$. However, we have some flexibility in the choice of σ in Equation 17. Daniel: is this the correct equation? As we decrease the variance of X_{norm}, the variance of X_{corr} grows by the same amount and is in fact the result of convolution with a Gaussian whose variance is the difference. Thus as σ decreases, $C(X)$ grows and approaches the

Input : Number of samples T, minibatch size m, error bound δ, pre-computed correction $C_1(X)$ distribution, initial sample θ_1.
Output : A chain of T samples $\{\theta_1, \ldots, \theta_T\}$ from $p(\theta)$;
for $t = \{1, \ldots, T\}$ **do**
  Propose a candidate θ' from proposal distribution $q(\theta' \mid \theta_t)$;
  Draw a minibatch of m points x_i, compute $\Delta^*(\theta_t, \theta')$ and sample variance $\sigma^2(\Delta^*)$;
  Estimate moments $E|\Lambda_i - \Lambda|$ and $E|\Lambda_i - \Lambda|^3$ from the sample, and error ϵ from Equation 26;
  while $\sigma^2(\Delta^*) \geq 1$ **or** $\epsilon > \delta$ **do**
    Draw m more samples to augment the minibatch, update Δ^*, $\sigma^2(\Delta^*)$ and ϵ estimates;
  end
  Draw $X_{\text{norm}} \sim \mathcal{N}(0, 1 - \sigma^2(\Delta^*))$ and X_{corr} from the correction distribution $C_1(X)$;
  if $\Delta^* + X_{\text{norm}} + X_{\text{corr}} > 0$ **then**
    Accept the candidate, $\theta_{t+1} = \theta'$;
  else
    Reject and re-use the old sample, $\theta_{t+1} = \theta_t$;
  end
end

Algorithm 1: A description of M-H sampling with our acceptance test. Daniel: I think we may want to be careful about the use of $C_1(X)$ because that implies that we have a fixed X_{corr} density, but that $C_1(X)$ will change depending on the iteration and variance of Δ^*, right? Hopefully not by much, but it still changes sometimes? ALSO bring up Xinlei's old points about X_{norm} confusion.

Table 1: Daniel: TODO need a good title.

N	σ	λ	L_1 error
4000	0.9	1	1.0e-4
4000	0.8	0.03	5.0e-6

derivative of a logistic function at $\sigma = 0$. It retains some very weak negative values for $\sigma > 0$ but removal of those values leads to very small error.

Daniel: TODO improve/describe Table 1 and try to relocate Algorithm 1 to be somewhere nicer.

5 Analysis

We now derive error bounds for our M-H test, and for the approximate target distribution that it generates. From Section 4, we know that it is possible to generate the correction samples X_{corr} with a CDF error of at most single-precision floating point error, and possibly better (Daniel: I can fix this to cite Table 1's values, that will probably be better.). We therefore treat the X_{corr} variable as a sample from the exact correction distribution and we will not analyze its errors.

In the most similar prior works, [3] uses asymptotic arguments based on the Central Limit Theorem to argue that its approximate acceptance test error tends to zero as batch size increases. But no quantitative bounds are given. In [4], explicit bounds are given, but they depend on bounding

$$C_{\theta,\theta'} = \max_{1 \leq i \leq N} |\log p(x_i \mid \theta') - \log p(x_i \mid \theta)|. \tag{21}$$

Such bounds can be derived efficiently for models such as logistic regression, but it is unclear how to derive them for a complex model such as a neural network. In general, since a new θ' value is obtained each iteration, one would need to iterate through all the $p(x_i \mid \theta')$ terms, defeating the purpose of minibatch MCMC.

In this paper, we use quantitative forms of the Central Limit Theorem which rely on measurable statistics from a single minibatch. Thus a sampler using our approach does not need to "see" data beyond the current minibatch. This supports use of the sampler on very large datasets, and in online mode where the dataset is presented as a stream.

For the analysis, in Section 5.1, we first present bounds on the absolute and relative error (in terms of the CDFs) of the distribution of Δ^* vs a Gaussian. We then show in Section 5.2 that these error

bounds are preserved after the addition of other random variables, in particular X_{norm} and X_{corr}. From this it follows that the acceptance test has the same error bound.

5.1 Bounding the Distribution of Δ^* Versus a Gaussian

Daniel: the following lemma is, I think, from Corollary 2 in [15] but I cannot derive the exact lemma as we have it here. To start, we use the following quantitative central-limit result (from [15]):

Lemma 3. *Let $X_1, \ldots, X_n$ be a set of zero-mean, independent, identically-distributed random variables with sample mean $\hat{X}$ and variance $\sigma^2(\hat{X})$ where:*

$$\hat{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \qquad \text{and} \qquad \sigma(\hat{X}) = \frac{1}{n}\left(\sum_{i=1}^{n}(X_i - \hat{X})^2\right)^{\frac{1}{2}}. \tag{22}$$

This means $t = \hat{X}/\sigma(\hat{X})$ has an approximate Student's distribution which approaches a normal distribution in the limit. Then from [15]:

$$\sup_x |\Pr(t < x) - \Phi(x)| \leq \frac{6.4E|X|^3 + 2E|X|}{\sqrt{n}}. \tag{23}$$

Daniel: the notation here was (and still is, to a lesser extent) the source of considerable confusion for me. Do we want $\sigma(X)$ or $\sigma(\hat{X})$? I think in Lemma 3 we want the former, not the latter. OH ALSO, this section uses n as the full data size, which means before this section, we should use n, not N, for the full data size, right? The Bardenet paper uses n to represent the full data size.

PS: I'd recommend not putting sample mean and sample variance in an equation if the $\sigma(\hat{X})$ gets clarified, since those computations are common knowledge.

PPS: The sample mean is probably more commonly denoted as $\bar{X}$.

Lemma 3 demonstrates that as long as we know the first and third absolute moments $E|X|$ and $E|X|^3$, we can bound the error of the normal approximation, which decays as $O(n^{-\frac{1}{2}})$. Making the change of variables $y = x\sigma(\hat{X})$, Equation 23 becomes

$$\sup_y \left|\Pr(\hat{X} < y) - \Phi\left(\frac{y}{\sigma(\hat{X})}\right)\right| \leq \frac{6.4E|X|^3 + 2E|X|}{\sqrt{n}}, \tag{24}$$

showing that the distribution of $\hat{X}$ approaches the normal distribution $\mathcal{N}(0, \sigma^2(\hat{X}))$ whose variance is $\sigma^2(\hat{X})$, measured from the sample.

To apply this to our test, let $X_i = \Lambda_i(\theta, \theta') - \Lambda(\theta, \theta')$, so that the X_i are zero-mean, i.i.d. variables. If instead of all n samples, we only extract a subset of b samples corresponding to our minibatch, we can connect $\hat{X}$ with our Δ^* term:

$$\hat{X} = \Delta^*(\theta, \theta') - \Delta(\theta, \theta'), \tag{25}$$

so that $\sigma(\hat{X}) = \sigma(\Delta^*)$. Daniel: This time it makes sense to use $\sigma(\hat{X})$ instead of $\sigma(X)$. This results in the following Corollary:

Corollary 1. *We can now substitute into Equation 24 and displace by the mean, giving:*

$$\sup_y \left|\Pr(\Delta^* < y) - \Phi\left(\frac{y - \Delta}{\sigma(\Delta^*)}\right)\right| \leq \frac{6.4E|X|^3 + 2E|X|}{\sqrt{b}} = \epsilon(\theta, \theta', b) \tag{26}$$

Corollary 1 shows that the distribution of Δ^* approximates a Normal distribution with mean Δ and variance $\sigma^2(\Delta^*)$. Furthermore, it bounds the error with *estimable quantities*: both $E|X|$ and $E|X|^3$ can be determined as means of $|\Lambda_i - \Lambda|$ and $|\Lambda_i - \Lambda|^3$, respectively, on each minibatch. We expect this will often be accurate enough on minibatches with hundreds or thousands of points, but otherwise bootstrap CIs can be computed from those sequences. Since the bounds are monotone in $E|X|$ and $E|X|^3$, using upper bootstrap CI limits will provide high-confidence error bounds.

5.2 Bounds are Preserved After Adding Random Variables

We next relate the CDFs of distributions and show that bounds are preserved after adding random variables.

Lemma 4. *Let $P(x)$ and $Q(x)$ be two cumulative distributions satisfying $\sup_x |P(x) - Q(x)| \leq \epsilon$ with x in some real range. Let $R(y)$ be the* density *of another random variable y. Let P' be the convolution $P * R$ and Q' be the convolution $Q * R$. Then $P'(z)$ (resp. $Q'(z)$) is the CDF of sum $z = x + y$ of independent random variables x with CDF $P(x)$ (resp. $Q(x)$) and y with density $R(y)$. Then*

$$\sup_x |P'(x) - Q'(x)| \leq \epsilon \tag{27}$$

Proof. We first observe that

$$P'(z) - Q'(z) = \int_{-\infty}^{+\infty} (P(z-x) - Q(z-x))R(x)dx, \tag{28}$$

and since $\sup_x |P(x) - Q(x)| \leq \epsilon$ it follows that for all z:

$$-\epsilon = \int_{-\infty}^{+\infty} -\epsilon R(x)dx \leq \int_{-\infty}^{+\infty} (P(z-x) - Q(z-x))R(x)dx \leq \int_{-\infty}^{+\infty} \epsilon R(x)dx = \epsilon \tag{29}$$

□

Daniel: Hold on, the following Corollary seems to add X_{norm} to both random variables we are comparing (and *then* X_{corr}). But didn't the previous section already justify that $\Delta^* \approx \Delta + X_{\text{norm}}$? That is, I don't think we need to add another X_{norm}; I think we just want to add X_{corr}. Otherwise, this new test does not match what we wrote earlier.

I think the easiest way to resolve this is to do the following: we assume that Δ^* is already $\Delta + X_{\text{noise}}$. Then all we need to do is add X_{corr}. So this is the same as what we are doing, except we do not need to apply Lemma 4 twice; we just need to do it once because we are only adding X_{corr}, not X_{norm}.

Relating Lemma 4 to our acceptance test, we observe that:

Corollary 2. *If $\sup_y |\Pr(\Delta^* < y) - \Phi(\frac{y-\Delta}{\sigma(\Delta^*)})| \leq \epsilon(\theta, \theta', b)$, then*

$$\sup_y |\Pr(\Delta^* + X_{\text{norm}} + X_{\text{corr}} < y) - S(y - \Delta)| \leq \epsilon(\theta, \theta', b) \tag{30}$$

where $S(x)$ is the standard logistic function, and X_{norm} and X_{corr} are generated as per Algorithm 1.

Proof. We apply Lemma 4 twice. First take:

$$P(y) = \Pr(\Delta^* < y) \qquad \text{and} \qquad Q(y) = \Phi\left(\frac{y - \Delta}{\sigma(\Delta^*)}\right) \tag{31}$$

and convolve with the distribution of X_n (Daniel: to be clear I think this was supposed to be X_{norm} instead of X_n but see comments earlier about why I think it is not necessary.) which has density $\phi(X/\sigma_n)$ where $\sigma_n^2 = 1 - \sigma^2(\Delta^*)$. This yields the next iteration of P and Q:

$$P'(y) = \Pr(\Delta^* + X_{\text{norm}} < y) \qquad \text{and} \qquad Q'(y) = \Phi\left(y - \Delta\right) \tag{32}$$

Now we convolve with the distribution of X_{corr} which gives:

$$P''(y) = \Pr(\Delta^* + X_{\text{norm}} + X_{\text{corr}} < y) \qquad \text{and} \qquad Q''(y) = S\left(y - \Delta\right) \tag{33}$$

Both steps preserve the error bound $\epsilon(\theta, \theta', b)$. Finally $S(y - \Delta)$ is a logistic CDF centered at Δ, and so $S(y - \Delta) = \Pr(\Delta + X_{\log} < y)$ for a logistic random $X_{\log}$. We conclude that the probability of acceptance for the actual test $\Pr(\Delta^* + X_{\text{norm}} + X_{\text{corr}} > 0)$ differs from the exact test $\Pr(\Delta + X_{\log} > 0)$ by at most ϵ. □

As our discussion in Section 3 showed, as the distribution of Δ^* approaches a Gaussian, our new MH test becomes more accurate. Corollary 2 shows that the bounds from Section 5.1 are preserved after the addition of the random variables we use, showing that our test should remain accurate.

Daniel: Do we want to include this new analysis? There seem to be two different ways we show the bound. The first part (above) is nice and I like it (and it seems to show all we need for Algorithm 1, doesn't it?

In fact we can do better than this approximation (showing the error decreases as $O(n^{-1})$) by using a more precise limit distribution under an additional assumption. Let μ_i denote the i^{th} moment, and b_i denote the i^{th} absolute moment of X. If Cramer's condition holds:

$$\lim_{t\to\infty} \sup |E(\exp(itX))| < 1, \tag{34}$$

then Equation 2.2 in Bentkus et al.'s work on Edgeworth expansions [16] provides the following:

Lemma 5. *Let $X_1, \ldots, X_n$ be a set of zero-mean, independent, identically-distributed random variables with sample mean $\hat{X}$ and with t defined as in Lemma 3. If X satisfies Cramer's condition, then*

$$\sup_x \left| \Pr(t < x) - G\left(x, \frac{\mu_3}{b_2^{3/2}}\right) \right| \leq \frac{c(\epsilon, b_2, b_3, b_4, b_{4+\epsilon})}{n} \tag{35}$$

where

$$G_n(x, y) = \Phi(x) + \frac{y(2x^2+1)}{6\sqrt{n}}\Phi'(x). \tag{36}$$

Lemma 5 shows that the average of the X_i has a more precise, skewed CDF limit $G_n(x, y)$ where the skew term has weight proportional to a certain measure of skew derived from the moments: $\frac{\mu_3}{b_2^{3/2}}$. Note that if the X_i are symmetric, the weight of the correction term is zero, and the CDF of the average of the X_i converges to $\Phi(x)$ at a rate of $O(n^{-1})$.

Here the limit $G_n(x, y)$ is a normal CDF plus a correction term that decays as $n^{1/2}$. (Daniel: how did the equation in the next sentence appear? I verified it and it seems correct, but it looks like it popped out of nowhere.) Importantly, since $x^2\phi(x) = \phi''(x) + \phi(x)$ where $\phi(x) = \Phi'(x)$, the correction term can be rewritten giving:

$$G_n(x, y) = \Phi(x) + \frac{y}{6\sqrt{n}}(2\phi''(x) + 3\phi(x)) \tag{37}$$

From which we see that $G_n(x, y)$ is a linear combination of $\Phi(x)$, $\phi(x)$ and $\phi''(x)$. This is quite fortuitous. In Algorithm 1, we correct for the difference in σ between Δ^* and the variance needed by X_{corr} using X_{norm}. This same method works when we wish to estimate the error in Δ^* vs $G_n(x, y)$. Since all of the component functions of $G_n(x, y)$ are derivatives of a (unit variance) $\Phi(x)$, adding a normal variable with variance σ' increases the variance of all three functions to $1 + \sigma'$. Thus we add X_{norm} as per Algorithm 1 preserving the limit in Equation 37.

The deconvolution approach can be used to construct a correction variable X_{corr} between $G_n(x, y)$ and $S(x)$ the standard logistic function. An additional complexity is that $G_n(x, y)$ has additional parameters y and n. Since these act as a single multiplier $\frac{y}{6\sqrt{n}}$ in Equation 37, its enough to consider a function $g(x, y')$ parametrized by $y' = \frac{y}{6\sqrt{n}}$. This function can be computed and saved offline. As we have shown above, errors in the "limit" function propagate directly through as errors in the acceptance test. To achieve a test error of say $1e-6$ (close to single floating point precision), we need a y' spacing of $1e-6$. It should not be necessary to tabulate values all the way to $y' = 1$, since y' is scaled inversely by the square root of minibatch size. Assuming a max y' of 0.1 requires us to tabulate about 100,000. Since our x resolution is 10,000, this leads to a table with about 1 billion values, which can comfortably be stored in memory. However, if $g(x, y)$ is moderately smooth in y, it should be possible to achieve similar accuracy with a much smaller table. We leave further analysis and experiments with $g(x, y)$ as future work. Daniel: I will look at the code to understand this paragraph better because right now I am confused.

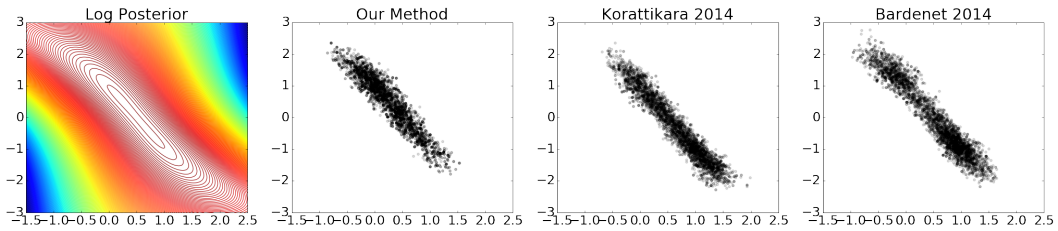

Figure 2: The log posterior contours and scatter plots of sampled θ values using different methods.

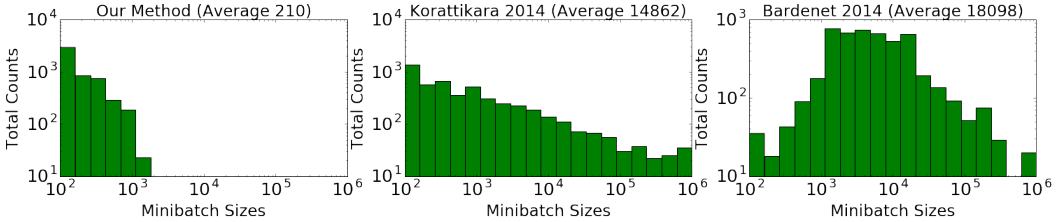

Figure 3: Histograms of minibatch sizes for the three methods used in Section 6.1. Daniel: The y-axis should probably just be "Counts" not "Total Counts." Also, was this run with the `plt.tight_layout()` command? The y-axis text for the second and third subplots almost overlap with the graphs to their left. Finally, I am still thinking about whether the y-axis should be on a log scale. I think the log scale is appropriate for the x-axis, it's a question of whether we should also apply it for the y-axis.

6 Experiments

We conduct two sets of experiments to explore the benefits of our minibatch MH test and to benchmark it with previous work. In Section 6.1, we show that our test enables samples to converge to the posterior distribution of a heated Gaussian mixture model. In Section 6.2, we analyze its efficiency on logistic regression. Appendices B and C contain more detailed information on these respective experiments.

6.1 Mixture of Gaussians

We start with a simple Gaussian mixture model, borrowing an experiment from [12]. The parameter is 2-D, $\theta = (\theta_1, \theta_2)$, and the parameter/data generation process is

$$(\theta_1, \theta_2) \sim \mathcal{N}((0,0), \mathrm{diag}(\sigma_1^2, \sigma_2^2)); \qquad x_i \sim 0.5 \cdot \mathcal{N}(\theta_1, \sigma_x^2) + 0.5 \cdot \mathcal{N}(\theta_1 + \theta_2, \sigma_x^2). \tag{38}$$

We set $\sigma_1^2 = 10, \sigma_2^2 = 1$ and $\sigma_x^2 = 2$. Fixing $\theta = (0, 1)$, we draw 1,000,000 data points so that the target distribution is $p(\theta) \prod_{i=1}^{1,000,000} p(x_i \mid \theta)$, with the prior based on the θ generation process in Equation 38. This results in rather sharp posterior modes and high $\mathrm{std}(\Delta')$ estimates, so we apply a temperature $T = 10,000$ to reduce $\mathrm{std}(\Delta')$. Taking logs, we get the target as shown in the far left of Figure 2. Daniel: the $\mathrm{std}(\Delta')$ stuff is outdated; let's figure out a way to change this. We may also have to justify such a high temperature, because this is far from the "real" (log) posterior.

We run MCMC with our MH test using minibatch size $m = 100$. We also run this using the methods from Korattikara 2014 [3] and Bardenet 2014 [4]. For [3], we use $m = 100$ and increment minibatches by that amount within a test if necessary, and for [4], we use $m = 100$ and increase the minibatch size geometrically with a ratio of $\gamma = 1.5$. The tolerance for making a decision in [3] is $\epsilon = 0.005$, and the error bound control parameters in [4] are $p = 2$ and $\delta = 0.01$. To make comparisons easier, all three use the same random walk proposer with covariance $\Sigma = \mathrm{diag}(0.3, 0.3)$. 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 5000 times to collect 5000 samples.

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

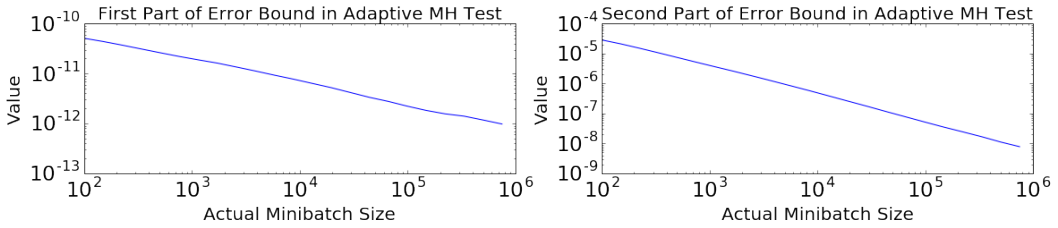

Figure 4: Value of first and second part of equation (9) in [4] with respect to actual minibatch size. Daniel: probably debatable on whether we should include this. Let's talk it over.

In addition, our method is superior based on the Poisson likelihood of the MCMC-generated parameters over the range of $\theta_1 \in (-1.5, 2.5)$ and $\theta_2 \in (-2.5, 2.5)$. This is a measure of divergence from the ground-truth distribution. The log likelihood values for our method, Korattikara 2014, and Bardenet 2014 are -2144.6, -4007.4, and -2598.4, respectively. Daniel: I rewrote this paragraph to make it slightly less offensive. Also, should we clarify what this means? Because to be honest, it seems hard to tell a significant difference among the methods by looking at the scatterplots.

Figure 3 shows a histogram of the final minibatch sizes used by the three methods in each iteration. Our methods consumes significantly less data; most minibatch sizes are smaller than 1000, and the average size is 420. The other two methods occasionally need to consume a large proportion of the entire data set of one million elements; the average minibatch sizes are 16378 and 16755 for Korattikara 2014 and Bardenet 2014, respectively. Daniel: the plot says our average minibatch size is 210, but the paragraph here says 420.

It is useful to notice that, in [4], they need to evaluate the likelihood term $C_{\theta,\theta'}$ over the entire the data set at every iteration in order to calculate the error bound in equation (9) in [4]. In Figure 4, the value of the first part and second part of equation (9) in [4] is shown with respect to the size of actual minibatch data used. The second part which contains the $C_{\theta,\theta'}$ term is larger than the first part, which means that the second part is non-negligible and has to be calculated at every iteration. This is one of the very time-consuming parts in this method. Daniel: I am not sure if we should include this plot. Let me think about it and if we decide to include it I will help rewrite the explanation.

6.2 Logistic Regression

We next use logistic regression for the binary classification of 1s versus 7s in the MNIST dataset [17]. The data has 12007 training and 1000 testing elements (we used a random subset of the test data). For the proposer, we use a random walk with covariance matrix $0.05I$ for the 784×784 identity matrix I. We set the posterior temperature at $T = 1000$ and use the minibatch size $m = 100$ and compare with [3] with tolerance 0.005 and with [4] with error bound control parameters $p = 2$ and $\delta = 0.01$. Daniel: just observed this, but why are we using a subset of the test data? We really should use all 2000 (I think it's 2000, right?).

Figure 5 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 (Daniel: are we sure about "fixed amount per iteration" because our Algorithm 1 implies we may have to increment the minibatch size, but it happens *less frequently* than the other methods.), but the other two methods 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 6 shows the histogram of minibatch sizes for all three methods. With an initial minibatch size of 100, Korattikara 2014 and Bardenet 2014 achieve average minibatch size of 585 and 2731 respectively over the MNIST classification task, while our method achieves that of 100, showing significant better performance than the benchmark test methods. Daniel: these numbers don't match the figure titles.

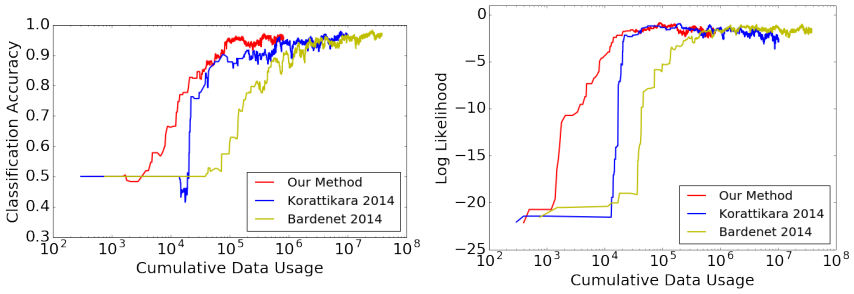

Figure 5: Logistic regression performance (accuracy/log likelihood) and minibatch size analysis. Daniel: let's use matplotlib's "subplot" functionality, so that the images appear the same size. We can handle this later; it will be easy.

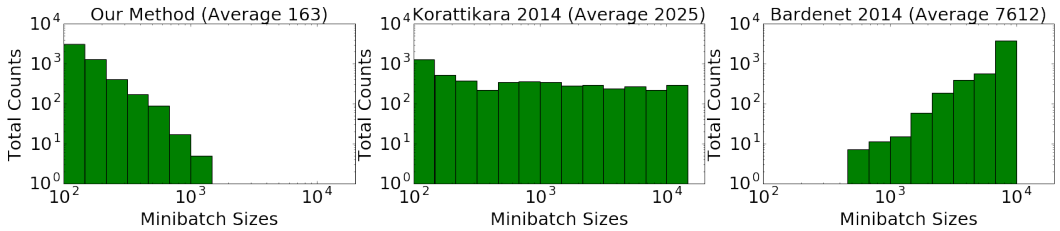

Figure 6: Counts of minibatch sizes needed for the three methods in the MNIST classification task.

7 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 and logistic regression. 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 [22]), or integrating parallel MCMC [23, 24] 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. Korattikara, 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] 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.

[5] 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.

[6] A. A. Barker, "Monte-carlo calculations of the radial distribution functions for a proton-electron plasma," *Australian Journal of Physics*, vol. 18, pp. 119–133, 1965.

[7] R. Bardenet1, A. Doucet, and C. Holmes, "On markov chain monte carlo methods for tall data," *arXiv preprint arXiv:1505.02827v1*, 2015.

[8] G. O. Roberts and J. S. Rosenthal, "Optimal scaling for various metropolis–hastings algorithms," *Statistical Science*, vol. 16, no. 4, p. 351–367, 2001.

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

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

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

[12] 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.

[13] 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.

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

[15] Y. Novak, "On self-normalized sums and student's statistic," *Theory of Probability and its Applications*, vol. 49, no. 2, pp. 336–344, 2005.

[16] V. Bentkus, F. Gotze, and W.R.vanZwet, "An edgeworth expansion for symmetric statistics," *Annals of Statistics*, vol. 25, no. 2, 1997.

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

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

[19] 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.

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

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

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

[23] 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.

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

Outline of Appendix

In this appendix, we describe the following topics:

- The proof of Lemma 1.
- More information on Section 6.1.
- More information on Section 6.2.

Daniel: Note that everything after Appendix A has yet to be revised. I don't know what we will include in them.

A Proof of Lemma 1

Assume $(\theta' - \theta) \in \pm\frac{1}{\sqrt{N}}[0.5, 1]$ and $\theta - 0.5 \in \pm\frac{1}{\sqrt{N}}[0.5, 1]$ with matching sign. These events occur with probability (Daniel: sorry for interjecting here but don't we need to take into account the $1/\sqrt{N}$ in the probabilities? If $N \to \infty$ (and it's going to be large with big datasets) the probabilities should be about zero.) $p_0 = 2 * (\Phi(1) - \Phi(0.5)) = 0.2997$ and $p_1 = (\Phi(1) - \Phi(0.5)) = 0.14988$ respectively, and guarantee that $\Lambda^*(\theta', \theta)$ is negative. This then guarantees that we can find a $u \in [0, 1]$ so that $\psi(u, \theta', \theta)$ equals $\mathbb{E}[\Lambda^*(\theta', \theta)]$. Specifically, choose u_0 to satisfy

$$\log u_0 = N(\Lambda^*(\theta', \theta) - \psi(1, \theta', \theta)) \tag{39}$$

which evaluates (Daniel: sorry again for interjecting but I really think $\log u_0$ does not have that N there. Also, for the subsequent equation, I think we need to be using *expected* values of Λ^*) to

$$\log u_0 = -N(\theta' - \theta)\left(\theta - 0.5 + \frac{\theta' - \theta}{2}\right). \tag{40}$$

This means we can rewrite the acceptance test as

$$\frac{1}{b}\sum_{i=1}^{b} x_i - \left(\theta + \frac{\theta' - \theta}{2} + \frac{\log u_0}{N(\theta' - \theta)}\right) \quad \not\approx \quad 0 \tag{41}$$

where $\not\approx$ means "significantly different from" under the distribution over samples of x_i. The above choice of u_0 ensures that the terms in parenthesis above sum to 0.5. Since the x_i have mean 0.5, the test will never correctly succeed. Furthermore, if we sample values of u near enough to u_0, the terms in parenthesis will not be sufficiently different from 0.5 to allow the test to succeed.

Daniel: I would like to ask you about these probabilities because I do not know how we got these intervals. The choices above for θ and θ' guarantee that

$$\log u_0 \in -[0.5, 1][0.75, 1.5] = -[0.375, 1.5] \tag{42}$$

and consider the range of u values

$$\log u \in \log u_0 + [-0.5, 0.375] \tag{43}$$

the size of the range in u is at least $\exp([-2, -1.125]) = [0.13534, 0.32465]$ and occurs with probability at least $p_2 = 0.18932$. With u in this range, we rewrite the test as:

$$\frac{1}{b}\sum_{i=1}^{b} x_i - 0.5 \quad \not\approx \quad \frac{\log u/u_0}{N(\theta' - \theta)} \tag{44}$$

so that the LHS has expected value zero. Given our choice of intervals for the variables, the RHS is in the range $1/\sqrt{N}[-1, 0.75]$. The standard deviation of the LHS given the interval constraints is at least $0.5/\sqrt{b}$. And so the gap between LHS and RHS is at most $2\sqrt{b/N}$ standard deviations.

The samples θ, $(\theta' - \theta)$ and u are drawn independently and so the probability of the conjunction of these events is $c = p_0 p_1 p_2 = 0.0085$.

B Gaussian Mixture Experiment Details

In this section, we go over the math details on the Gaussian mixture model problem borrowed from [12]. Our parameter is a 2-D vector $\theta = (\theta_1, \theta_2)$, where

$$\theta_1 \sim \mathcal{N}(0, \sigma_1^2) \quad \text{and} \quad \theta_2 \sim \mathcal{N}(0, \sigma_2^2) \tag{45}$$

where $\mathcal{N}$ indicates the normal distribution (more generally, the multivariate normal). We consider the above as our prior. Following [12], 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

$$\log p(\theta) = \log\left(\frac{1}{2\pi\sqrt{10}}\right) - \frac{1}{2}\theta^T \Sigma^{-1} \theta. \tag{46}$$

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

$$x_i \sim \frac{1}{2}\mathcal{N}(\theta_1, \sigma_x^2) + \frac{1}{2}\mathcal{N}(\theta_1 + \theta_2, \sigma_x^2) \tag{47}$$

where, again following [12], we set $\sigma_x^2 = 2$. This means the log likelihood of a single data instance is

$$\log p(x_i \mid \theta) = \log\left(\frac{1}{4\sqrt{\pi}}\right) + \log\left(\exp\left(-\frac{1}{4}(x_i - \theta_1)^2\right) + \exp\left(-\frac{1}{4}(x_i - (\theta_1 + \theta_2))^2\right)\right) \tag{48}$$

Here is the problem statement: given some number of conditionally independent data points $x_1, x_2, \ldots, x_N$ generated according to (47), determine the posterior distribution of θ:

$$\log p(\theta \mid x_1, \ldots, x_N) = \log p(\theta) + \sum_{i=1}^{N} \log p(x_i \mid \theta). \tag{49}$$

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 (49). In that case, the literature generally uses

$$\log p(\theta \mid x_1, \ldots, x_N) \approx \log p(\theta) + \frac{N}{n}\sum_{i=1}^{n} \log p(x_i \mid \theta). \tag{50}$$

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

$$\log p(\theta \mid x_1, \ldots, x_N) \approx \log p(\theta) + \frac{1}{T}\frac{N}{n}\sum_{i=1}^{n} \log p(x_i \mid \theta). \tag{51}$$

which has the extra $1/T$ to decrease the scale factor. Equation (51) 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 7 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 8 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

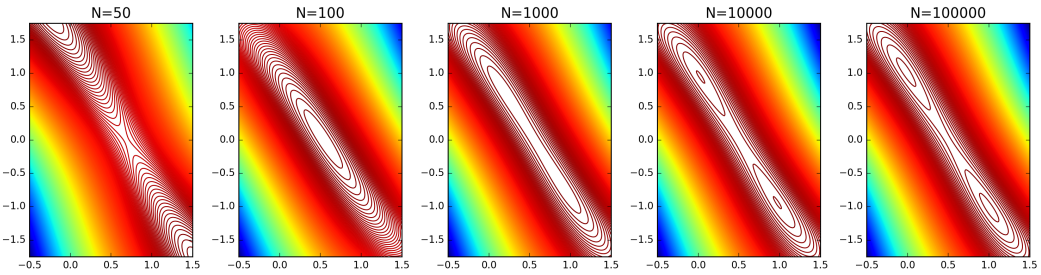

Figure 7: The posterior distribution, from 50 to 100k samples, with temperature set at 1.

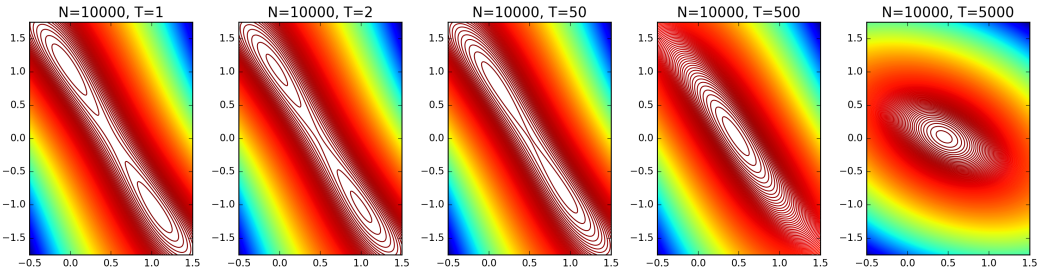

Figure 8: The posterior distribution, with $N = 10000$ but with temperature T varying.

the binary classification case, so we only use digits one (denoted as output $y = 1$) and seven (denoted as $y = -1$). The probability of the i^{th} output $y_i \in \{-1, +1\}$ with the input vector x_i is

$$p(y_i \mid x_i) = \frac{1}{1 + \exp(-y_i \theta^T x_i)}, \tag{52}$$

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 $\sigma^2 = 0.05$ for all components. We initialize θ_0 to be a vector of all ones, and set our minibatch size as $m = 100$.

For our minibatch MH test, in order to enforce the $\text{std}(\Delta') < 1.2$ condition, we use a constant temperature $T = 1000$. If our estimated $\text{std}(\Delta') \geq 1.2$, we ignore the current iteration.