Minibatch Gibbs Sampling on Large Graphical Models

Christopher De Sa [1] **Vincent Chen** [1] **Wing Wong** [2]

Abstract

Gibbs sampling is the de facto Markov chain Monte Carlo method used for inference and learning on large scale graphical models. For complicated factor graphs with lots of factors, the performance of Gibbs sampling can be limited by the computational cost of executing a single update step of the Markov chain. This cost is proportional to the degree of the graph, the number of factors adjacent to each variable. In this paper, we show how this cost can be reduced by using minibatching: subsampling the factors to form an estimate of their sum. We introduce several minibatched variants of Gibbs, show that they can be made unbiased, prove bounds on their convergence rates, and show that under some conditions they can result in asymptotic single-update-run-time speedups over plain Gibbs sampling.

1. Introduction

Gibbs sampling is a Markov chain Monte Carlo method that is one of the most widespread techniques used with graphical models (Koller & Friedman, 2009). Gibbs sampling is an iterative method that repeatedly resamples a variable in the model from its conditional distribution, a process that is guaranteed to converge asymptotically to the desired distribution. Since these updates are typically simple and fast to run, Gibbs sampling can be applied to a variety of problems, and has been used for inference on large-scale graphical models in many systems (Newman et al., 2007; Lunn et al., 2009; McCallum et al., 2009; Smola & Narayanamurthy, 2010; Theis et al., 2012; Zhang & Ré, 2014).

Unfortunately, for large graphical models with many factors, the computational cost of running an iteration of Gibbs sampling can become prohibitive. Even though Gibbs sampling is a *graph-local* algorithm, in the sense that each update only needs to reference data associated with a local neighborhood of the factor graph, as graphs become large and highly connected, even these local neighborhoods can become huge. To make matters worse, complicated models tend to have categorical variables with large domains: for example, the variables can represent US States or ZIP codes, where there are dozens to thousands of possible choices. The cost of computing a single iteration of Gibbs sampling is proportional to the *product* of the size of the local neighborhood and the size of the domain of the categorical random variables. More explicitly, if the *maximum degree* of a variable in the factor graph is Δ, and each variable can take on D possible values, then a single Gibbs sampling update takes $O(D\Delta)$ time to compute in the worst case.

This effect limits the scalability of Gibbs sampling. What started as a fast, efficient system can become dramatically slower over the course of development as models become more complicated, new factors are added, and new values are included in the variables' domains. To address this, practitioners have developed techniques such as *pruning* (Rekatsinas et al., 2017) which improve scalability by making the factor graphs simpler. While this does make Gibbs sampling faster, it comes at the cost of introducing bias which reduces fidelity to the original model.

In this paper, we explore a more principled approach to making Gibbs sampling scalable. Our approach is inspired by *minibatching* in stochastic gradient descent, which has been used with great success to scale up machine learning training. The main idea is that, when a Gibbs sampling update depends on a large local graph neighborhood, we can instead *randomly subsample* the neighborhood. Supposing the random sample is representative of the rest of the neighborhood, we can proceed to perform the update using just the random sample, rather than the (much larger) entire neighborhood. In this paper, we study minibatching for Gibbs Sampling, and we make the following contributions:

- We introduce techniques for *minibatching* which can dramatically reduce the cost of running Gibbs sampling, while provably adding *no bias* to the samples.
- We prove bounds on the convergence rates of our minibatch Gibbs algorithms, as measured by the *spectral gap* of the Markov chain. We give a recipe for how to

[1]Department of Computer Science, Cornell University, Ithaca, NY, USA [2]Department of Statistics, Stanford University, Stanford, CA, USA. Correspondence to: Christopher De Sa <cdesa@cs.cornell.edu>.

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

Table 1. Single-iteration computational complexity bounds of our minibatching algorithms compared with ordinary Gibbs sampling, for parameter settings which can slow down convergence, as measured by the spectral gap, by no more than a $O(1)$ factor.

Algorithm	**Compute Cost/Iteration**	**Notes**
Gibbs sampling	$O(D\Delta)$	
MIN-Gibbs: Minibatch Gibbs	$O(D\Psi^2)$	with high probability
MGPMH: Minibatch-Gibbs-Proposal Metropolis Hastings	$O(DL^2 + \Delta)$	
DoubleMIN-Gibbs: Doubly Minibatch Gibbs	$O(DL^2 + \Psi^2)$	with high probability

set the algorithms' parameters so that the spectral gap of minibatch Gibbs can be bounded to be arbitrarily close to the spectral gap of the original Gibbs chain.

- For a class of graphs with bounded energy, we show doing this results in an asymptotic computation cost of $O(D + \Delta)$, a substantial speedup from the cost of Gibbs sampling which is $O(D\Delta)$.

1.1. Background and Definitions

First, to make our claims rigorous, we describe our setting by defining a factor graph and the various conditions and conventions we will be using in the paper. A factor graph (Koller & Friedman, 2009) with n variables determines a probability distribution π over a state space Ω. In general, a state $x \in \Omega$ is an assignment of a value to each of the n variables, and each variable has its own domain over which it can take on values. For simplicity, in this paper we will assume all variables take on values in the same domain $\{1, \ldots, D\}$ for some constant D, which would make Ω the set of functions $x : \{1, \ldots, n\} \to \{1, \ldots, D\}$.[1] The probability distribution π is determined as the product of some set of factors Φ, such that for any $x \in \Omega$,

$$\pi(x) \propto \exp\left(\sum_{\phi\in\Phi} \phi(x)\right);$$

this is sometimes called the *Gibbs measure*. In this document, we use the notation $\rho(v) \propto \exp(\epsilon_v)$ to mean the unique distribution that is proportional to $\exp(\epsilon_v)$, that is,

$$\rho(v) = \frac{\exp(\epsilon_v)}{\sum_{w=1}^{D} \exp(\epsilon_w)}.$$

Equivalently, we can define an *energy function* ζ, where

$$\zeta(x) = \sum_{\phi\in\Phi} \phi(x),$$

and let $\pi(x) \propto \exp(\zeta(x))$. (Note that this definition implicitly excludes models with hard constraints or zero-probability states.) This is called a factor graph because the factors ϕ and variables i have a dependency relation in that each ϕ depends only on the values of some of the variables, but typically not all n. Formally, a factor ϕ depends on a variable if changing the value of that variable could change the value of ϕ—meaning, if there exists states x and y which differ only in variable i and for which $\phi(x) \neq \phi(y)$. Using this, we define the relation

$$A = \{(i, \phi) | \text{factor } \phi \text{ depends on variable } i\}.$$

Equivalently, this is a bipartite graph on the variables and factors; this edge relation, together with the factor function descriptions, defines a factor graph. Using this, we formally describe Gibbs sampling, which we write in Algorithm 1. Note that we use the notation $x(i) \leftarrow u$ to denote variable i within state x being reassigned the value u, and following standard relation notation we let $A[i] = \{\phi | (i, \phi) \in A\}$ be the set of factors that depend on variable i.

Algorithm 1 Gibbs sampling

```
given: initial state $x \in \Omega$
loop
  sample variable index $i$ uniformly from $\{1, \ldots, n\}$
  for all $u$ in $\{1, \ldots, D\}$ do
    $x(i) \leftarrow u$
    $\epsilon_u \leftarrow \sum_{\gamma \in A[i]} \phi(x)$
  end for
  construct distribution $\rho$ over $\{1, \ldots, D\}$ where
      $\rho(v) \propto \exp(\epsilon_v)$
  sample $v$ from $\rho$.
  $x(i) \leftarrow v$
  output sample $x$
end loop
```

Next, we describe several conditions on the factor functions that we will be using throughout this paper. For all that follows, we will suppose that $\phi(x) \geq 0$ for all ϕ and for all x; this holds without loss of generality (for models without hard constraints) because adding a constant to a factor function ϕ does not change the resulting distribution.

Definition 1 (Factor graph conditions)**.** The *maximum energy* of a factor ϕ is the smallest $M_\phi \in \mathbb{R}$ such that,

$$0 \leq \phi(x) \leq M_\phi \text{ for all } x \in \Omega.$$

[1]This means that the state space Ω is discrete, and we will be focusing on discrete-valued models in this paper. Continuous-valued models could be approximated with our algorithms by discretizing their domains to any desired level of accuracy.

The *total maximum energy* Ψ of a graph is the sum of all the maximum energies

$$\Psi = \sum_{\phi \in \Phi} M_\phi.$$

The *local maximum energy* L of a graph is the largest sum of all the maximum energies of factors that depend on a single variable i,

$$L = \max_{i \in \{1,\ldots,n\}} \sum_{\phi \in A[i]} M_\phi.$$

The *maximum degree* Δ of a factor graph is the largest number of factors that are connected to a variable,

$$\Delta = \max_{i \in \{1,\ldots,n\}} |A[i]|.$$

For large models with many low-energy factors, the total maximum energy can be much smaller than the number of factors, and the local maximum energy can be much smaller than the maximum degree. We will introduce several minibatch Gibbs variants in this paper, and their relative computation costs, in terms of the quantities defined in Definition 1, are summarized in Table 1.

1.2. Related Work

Several recent papers have explored applying minibatching to speed up large-scale Markov chain Monte Carlo methods. Our work is inspired by Li & Wong (2017), which shows how minibatching can be applied to the Metropolis-Hastings algorithm (Hastings, 1970). They show that their method, MINT-MCMC, can outperform existing techniques that use gradient information, such as stochastic gradient descent and stochastic gradient Langevin dynamics, on large-scale Bayesian neural network tasks. This is one among several papers that have also showed how to use minibatching to speed up Metropolis-Hastings under various conditions (Korattikara et al., 2014; Bardenet et al., 2014; Seita et al., 2017). More general related approaches include Firefly MC, which implements a minibatching-like computational pattern using auxiliary variables (Maclaurin & Adams, 2014), and block-Poisson estimation, which constructs unbiased log-likelihood estimates using Poisson variables (Quiroz et al., 2018). Our results are distinct from this line of work in that we are the first to study minibatch Gibbs sampling in depth[2], and we are the first to address the effect of categorical random variables with large domains that can limit computational tractability. Additionally, none of these papers provides any theoretical bounds on the convergence rates of their minibatched algorithm compared to the original MCMC chain.

2. MIN-Gibbs

[2]Johndrow et al. (2015) does evaluate a version of a minibatched Gibbs sampler on a particular application, but does not study Gibbs in depth.

Algorithm 2 MIN-Gibbs: Minibatch Gibbs sampling

```
given: minibatch estimator distributions $\mu_x$ for $x \in \Omega$
given: initial state $(x, \epsilon) \in \Omega \times \mathbb{R}$
loop
  sample variable index $i$ uniformly from $\{1, \ldots, n\}$
  $\epsilon_{x(i)} \leftarrow \epsilon$
  for all $u$ in $\{1, \ldots, D\} \setminus \{x(i)\}$ do
    $x(i) \leftarrow u$
    sample energy $\epsilon_u$ from $\mu_x$
  end for
  construct distribution $\rho$ over $\{1, \ldots, D\}$ where

      $\rho(v) \propto \exp(\epsilon_v)$

  sample $v$ from $\rho$.
  $x(i) \leftarrow v$
  $\epsilon \leftarrow \epsilon_v$
  output sample $(x, \epsilon)$
end loop
```

In this section, we will present our first algorithm for applying minibatching to Gibbs sampling. As there are many options when doing minibatching[3] we will present our algorithm in terms of a general framework for Gibbs sampling in which we use an estimate for the energy rather than computing the energy exactly. Specifically, we imagine the result of minibatching on some state $x \in \Omega$ is a random variable ϵ_x such that

$$\epsilon_x \approx \sum_{\phi \in \Phi} \phi(x).$$

For example, we could assign ϵ_x as

$$\epsilon_x = \frac{|\Phi|}{B} \sum_{\phi \in S} \phi(x)$$

where S is a randomly chosen subset of Φ of size B. More formally, we let μ_x be the distribution of ϵ_x, and we assume our algorithm will have access to μ_x for every state x and can draw samples from it. For simplicity, we assume μ_x has finite support, which is true for any minibatch estimator.

We can now replace the exact sums in the Gibbs sampling algorithm with our approximations ϵ_x. If sampling from μ_x is easier than computing this sum, this will result in a faster algorithm than vanilla Gibbs sampling. There is one further optimization: rather than re-estimating the energy of the current state x at each iteration of our algorithm, instead we can *cache* its value as it was computed in the previous step. Doing this results in Algorithm 2, MIN-Gibbs. We call our algorithm MIN-Gibbs because it was inspired by the Mini-batch Tempered MCMC (MINT-MCMC) algorithm presented by Li & Wong (2017) for applying minibatching to Metropolis-Hastings, another popular MCMC algorithm.

[3]These options include: what the batch size is, whether the batch size is fixed or random, whether to do weighted sampling, whether to sample with replacement, etc.

MINT-MCMC also used the idea of caching the energy to form an augmented system with state space $\Omega \times \mathbb{R}$. Compared with their algorithm, our contributions are that: (1) we modify the technique to apply to Gibbs sampling; (2) we show that *tempering* (which is sampling from a modified distribution at a higher temperature) and other sources of bias can be circumvented by using a bias-adjusted minibatching scheme; and (3) we prove bounds on the convergence rate of our technique.

Because it uses an energy estimate rather than the true energy, this algorithm is not equivalent to standard Gibbs sampling. This raises the question: will it converge to the same distribution π, and if not, what will it converge to? This question can be answered by using the property of *reversibility* also known as *detailed balance*.

Definition 2. A Markov chain with transition matrix T is *reversible* if for some distribution $\bar{\pi}$ and all states $x, y \in \Omega$,

$$\bar{\pi}(x)T(x,y) = \bar{\pi}(y)T(y,x).$$

It is a well known result that if a chain T is reversible, the distribution $\bar{\pi}$ will be a stationary distribution of T, that is, $\bar{\pi}T = \bar{\pi}$. Thus we can use reversibility to determine the stationary distribution of a chain, and we do so for Algorithm 2 in the following theorem.

Theorem 1. *The Markov chain described in Algorithm 2 is reversible and has stationary distribution*

$$\bar{\pi}(x,\epsilon) \propto \mu_x(\epsilon) \cdot \exp(\epsilon)$$

and its marginal stationary distribution in x will be

$$\bar{\pi}(x) \propto \mathbf{E}_{\epsilon \sim \mu_x}\left[\exp(\epsilon)\right].$$

Interestingly, this theorem shows that if we choose our estimation scheme such that for all $x \in \Omega$,

$$\mathbf{E}_{\epsilon \sim \mu_x}\left[\exp(\epsilon)\right] = \exp(\zeta(x)) = \textstyle\prod_{\phi \in \Phi} \exp\left(\phi(x)\right) \qquad (1)$$

then MIN-Gibbs will actually be *unbiased*: we will have $\pi(x) = \bar{\pi}(x)$. Next, we will show how we can take advantage of this by constructing estimators that satisfy (1) using minibatching. Suppose without loss of generality that each ϕ is non-negative, $\phi(x) \geq 0$. Let λ be a desired average minibatch size, and for each factor ϕ, let s_ϕ be an independent Poisson-distributed random variable with mean $\lambda M_\phi / \Psi$. Let $S \subset \Phi$ denote the set of factors ϕ for which $s_\phi > 0$. Then for any state x, define the estimator

$$\epsilon_x = \sum_{\phi \in S} s_\phi \log\left(1 + \frac{\Psi}{\lambda M_\phi}\phi(x)\right). \qquad (2)$$

Lemma 1. *The estimator ϵ_x defined in (2) satisfies the unbiasedness condition in (1).*

Proof. The expected value of the exponential of the estimator defined in (2) is

$$\mathbf{E}\left[\exp(\epsilon_x)\right] = \mathbf{E}\left[\exp\left(\textstyle\sum_{\phi \in S} s_\phi \log\left(1 + \frac{\Psi}{\lambda M_\phi}\phi(x)\right)\right)\right].$$

Since the s_ϕ are independent, this becomes

$$\mathbf{E}\left[\exp(\epsilon_x)\right] = \textstyle\prod_{\phi \in \Phi} \mathbf{E}\left[\exp\left(s_\phi \log\left(1 + \frac{\Psi}{\lambda M_\phi}\phi(x)\right)\right)\right].$$

Each of these constituent expected values is an evaluation of the moment generating function of a Poisson random variable. Applying the known expression for this MGF and simplifying gives us the expression in (1), which is what we wanted to prove. □

From the result of this lemma, we can see that using this *bias-adjusted* estimator with MIN-Gibbs will result in an unbiased chain with stationary distribution π. However, the chain being unbiased does not, by itself, mean that this method will be more effective than standard Gibbs. For that to be true, it must also be the case that approximating the energy did not affect the convergence rate of the algorithm—at least not too much. The convergence times of Gibbs samplers can vary dramatically, from time linear in the number of variables to exponential time. As a result, it is plausible that approximating the energy as we do in Algorithm 2 could switch us over from a fast-converging chain to a slow-converging one, and as a result even though the individual iterations would be computationally faster, the overall computation would be slow—or worse, would silently give incorrect answers when the chain fails to converge. Recently, other methods that have been used to speed up Gibbs sampling, such as running asynchronously (De Sa et al., 2016) and changing the order in which the variables are sampled (He et al., 2016), have been shown in some situations to result in algorithms that converge significantly slower than plain Gibbs sampling. Because of this, if we want to use algorithms like MIN-Gibbs with confidence, we need to show that this will not happen for minibatching.

To show that minibatching does not have a disastrous effect on convergence, we need to bound the convergence rate of our algorithms. To measure the convergence rate, we use a metric called the *spectral gap* (Levin et al., 2009).

Definition 3. Let T be the transition matrix of a reversible Markov chain. Since it is reversible, its eigenvalues must all be real, and they can be ordered as

$$1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{|\Omega|}.$$

The spectral gap γ is defined as $\gamma = \lambda_1 - \lambda_2$.

The spectral gap measures the convergence rate of a Markov chain in terms of the ℓ_2-distance, in that the larger the spectral gap, the faster the chain converges. The spectral gap is related to several other metrics of convergence for Markov chains. For example, it is a standard result that for a lazy Markov chain (a Markov chain is called lazy if at each iteration, it stays in its current state with probability at least $1/2$) the *mixing time* of the chain, which measures the number of steps required to converge to within total-variation distance ϵ of the stationary distribution π, is bounded by

$$t_{\text{mix}}(\epsilon) \leq \frac{1}{\gamma} \log \left(\frac{1}{\epsilon \cdot \min_{x \in \Omega} \pi(x)} \right).$$

In order to determine the effect of using minibatching on convergence, we would like to bound the spectral gap of MIN-Gibbs in terms of the spectral gap of the original Gibbs sampling chain. In the following theorem, we show that if the energy estimator ϵ_u is always sufficiently close to the true energy, then we can bound the spectral gap.

Theorem 2. *Let $\bar{\gamma}$ be the spectral gap of MIN-Gibbs running with an energy estimator μ_x that has finite support and that satisfies, for some constant $\delta > 0$ and every $x \in \Omega$,*

$$\mathbf{P}_{\epsilon_x \sim \mu_x} \left(|\epsilon_x - \zeta(x)| \leq \delta \right) = 1.$$

Let γ be the spectral gap of a vanilla Gibbs sampling chain running using the exact energy. Then,

$$\bar{\gamma} \geq \exp(-6\delta) \cdot \gamma.$$

That is, the convergence is slowed down by at most a constant factor of $\exp(-5\delta)$—which is independent of the size of the problem. This theorem guarantees that if we can restrict our estimates to being within a distance of $O(1)$ of the exact energy, then the convergence rate will not be slowed down by more than an $O(1)$ constant factor.

Unfortunately, the estimator presented in (2) is not going to be always bounded by a constant distance from the exact energy. Still, since it is the sum of independent terms with bounded variance, we *can* bound it with high probability.

Lemma 2. *For any constants $0 < \delta$ and $0 < a < 1$, if we assign an expected batch size*

$$\lambda \geq \max \left(\frac{8\Psi^2}{\delta^2} \log \left(\frac{2}{a} \right), \frac{2\Psi^2}{\delta} \right),$$

then the estimator in (2) satisfies $\mathbf{P}\left(|\epsilon_x - \zeta(x)| \geq \delta \right) \leq a$.

This lemma lets us construct minibatch estimators that remain arbitrarily close to the true energy with arbitrarily high probability. Furthermore, we can do this with a minibatch size independent of the number of variables or factors, depending instead on the total energy. This means that if we have a very large number of low-energy factors, we can get significant speedup from MIN-Gibbs with high-probability theoretical guarantees on the convergence rate. In particular, since the computational cost of running a single epoch of MIN-Gibbs is D times the minibatch size, and this lemma suggests we need to set $\lambda = \Omega(\Psi^2)$, it follows that the total computational cost of MIN-Gibbs will be $O(\Psi^2 D)$.

Validation of MIN-Gibbs. Having characterized the effects of minibatching on Gibbs sampling, we present a synthetic scenario where Algorithm 2 can be applied. The Ising model (Ising, 1925) is a probabilistic model over an $N \times N$ lattice, with domain $x(i) \in \{-1, 1\}$. The Ising model has a physical interpretation in which each $x(i)$ represent the magnetic *spin* at each site. The energy of a configuration is given by:

$$\zeta_{\text{Ising}}(x) = \textstyle\sum_{i=1}^N \sum_{j=1}^N \beta \cdot A_{ij} \cdot (x(i)x(j) + 1),$$

where A_{ij} is called the *interaction* between variable i and j, and β is the *inverse temperature*.[4] We chose to use the Ising model to validate MIN-Gibbs because it is a simple model that nevertheless has non-trivial statistical behavior.

We chose an Ising model in which each site is fully connected (i.e. $\Delta = N^2 - 1$), and the strength of interaction A_{ij} between any two sites is determined based on their distance by a Gaussian kernel. We simulated Algorithm 2 on a graph with $n = N^2 = 400$ and inverse temperature $\beta = 1$. This β parameter was hand-tuned such that the Gibbs sampler seemed to mix in about the number of iterations we wanted to run. To have a fair comparison, the same setup was then used to evaluate MIN-Gibbs. For this model, $L = 2.21$ and $\Psi = 416.1$.[5] We started the algorithm with a unmixed configuration where each site takes on the same state ($x(i) = 1$ for all i).

As the algorithm ran, we used the output samples to compute a running average of the marginal distributions of each variable. By symmetry (since negating a state will not change its probability), the marginal distribution of each variable in the stationary distribution π should be uniform, so we can use the distance between the estimated marginals and the uniform distribution as a proxy to evaluate the convergence of the Markov chain. Figure 1 shows the average ℓ_2-distance error in the estimated marginals compared with the fully-mixed state. Notice that as the batch size increases, MIN-Gibbs approaches vanilla Gibbs sampling.

Using local structure. Although MIN-Gibbs has a computational cost that is independent of the size of the graph, it still depends on the total maximum energy Ψ. That is, unlike vanilla Gibbs sampling, which has a computational cost that depends only on the number of factors *local* to the variable that is being sampled, MIN-Gibbs depends on *global* properties of the graph. This is because the estimators used by MIN-Gibbs are approximating the whole energy sum $\zeta(x)$, rather than approximating the sum over just those factors which depend on the variable that is being resampled. How

[4]In some settings, the Ising model is used with an additional bias term (which physically represents a static magnetic field), but here for simplicity we do not include this term.

[5]Note that since we chose β large enough that $\Psi^2 > \Delta$ for this model, we do not expect MIN-Gibbs to be faster than Gibbs sampling for this particular synthetic example.

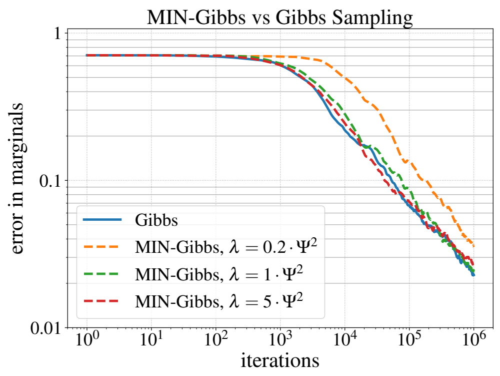


Figure 1. Convergence of marginal estimates for MIN-Gibbs compared with vanilla Gibbs sampling.

Algorithm 3 Local Minibatch Gibbs

```
given: initial state $x \in \Omega \times \mathbb{R}$, minibatch size $B$
loop
  sample variable index $i$ uniformly from $\{1, \ldots, n\}$
  sample minibatch $S \subset A[i]$ uniformly s.t. $|S| = B$
  for all $u$ in $\{1, \ldots, D\}$ do
    $x(i) \leftarrow u$
    $\epsilon_u \leftarrow \frac{|A[i]|}{|S|} \sum_{\gamma \in S} \phi(x)$
  end for
  construct distribution $\rho$ over $\{1, \ldots, D\}$ where
      $\rho(v) \propto \exp(\epsilon_v)$
  sample $v$ from $\rho$.
  $x(i) \leftarrow v$
  output sample $x$
end loop
```

can we modify MIN-Gibbs to take advantage of this local structure?

One straightforward way to do it is to allow the estimators ϵ_x to be *dependent*, rather than independent. That is, instead of choosing a unique minibatch every time we estimate the energy, we choose one fixed minibatch for each iteration. Once our minibatch is fixed, the conditional distribution computation will exhibit the same cancellation of irrelevant factors that vanilla Gibbs has, and we will be able to compute the transition probabilities using only local information. One side-effect of this is that we can no longer use the energy-caching technique that MIN-Gibbs uses, as we will no longer ever be estimating the total energy, but rather only the local component of the energy (the part that is dependent on the variable we are re-sampling). This would result in something like Algorithm 3. (We could also consider variants of this algorithm that use different minibatching schemes, such as (2), but here for simplicity we present the version that uses standard minibatching.) Note that Algorithm 3 is nearly identical to vanilla Gibbs sampling, except that it uses a single minibatch to estimate all the energies in each iteration.

Algorithm 3 will run iterations in time $O(BD)$, which can be substantially faster than plain Gibbs, which runs in $O(\Delta D)$. We evaluate Algorithm 3 empirically, and demonstrate in Figure 2(a) that it converges, with almost the same trajectory as plain Gibbs, for various values of the batch size B. The simulation here is on the same Ising model as in Algorithm 2, with the same parameters. Unfortunately, it is unclear if there is anything useful we can say about its convergence rate or even what it converges to. Unlike MIN-Gibbs, because of the lack of energy-caching there is no obvious reversibility argument to be made here, and so we can not prove bounds on the spectral gap, since those bounds require reversibility.

3. Minibatch-Gibbs-Proposal MH

We left off in the previous section by presenting Algorithm 3, which we showed has promising computational cost but gives us no guarantees on accuracy or convergence rate. There is a general technique that we can use to transform such chains into ones that *do* have accuracy guarantees: Metropolis-Hastings (Hastings, 1970). This well-known technique uses updates from an arbitrary Markov chain, called the *proposal distribution*, but then chooses whether or not to *reject* the update based on the true target distribution π. This rejection step reshapes the proposal chain into one that is reversible with stationary distribution π. A natural next step for us is to use Metropolis-Hastings with Algorithm 3 as a proposal distribution. Doing this results in Algorithm 4.[6]

Because Algorithm 4 is based on Metropolis-Hastings, it is natural for it to be reversible and have π as its stationary distribution. We prove that this must be the case. We also prove a bound on the spectral gap of the chain.

Theorem 3. *Algorithm 4 is reversible, and it has stationary distribution π.*

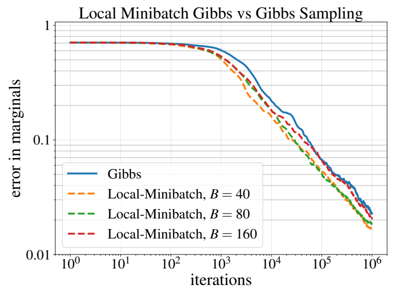
Theorem 4. *Let $\bar{\gamma}$ be the spectral gap of MGPMH, and let γ be the spectral gap of a vanilla Gibbs sampling chain running on the same factor graph. Suppose the expected minibatch size is large enough that $L \leq \lambda$. Then,*

$$\bar{\gamma} \geq \exp\left(-L^2/\lambda\right) \cdot \gamma.$$

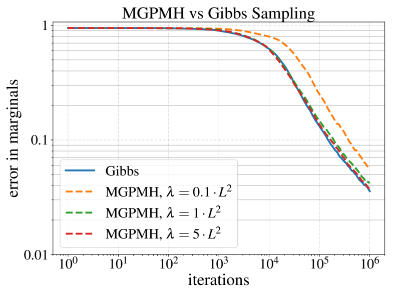
These theorems mean that MGPMH will converge to the correct stationary distribution π, and will do so with a convergence rate that is at most a factor of $\exp(L^2/\lambda)$ slower than the vanilla Gibbs chain. By making $\lambda \approx L^2$, this difference in convergence rates can be made $O(1)$.

On the other hand, when we look at the computational cost of an iteration of Algorithm 4, we notice it will be $O(\Delta +$

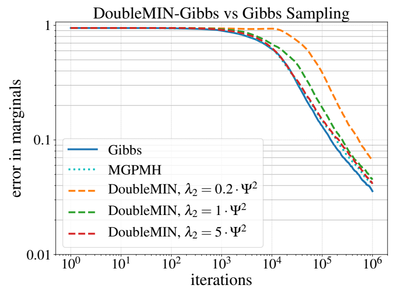
[6]Note that Algorithm 4 is not *precisely* Metropolis-Hastings, because its acceptance probability differs somewhat from what Metropolis-Hastings would have as it is dependent on prior randomness (the selected variable i and minibatch weights s_ϕ).



(a)



(b)



(c)

Figure 2. Convergence of (a) Local Minibatch Gibbs, (b) MGPMH, and (c) DoubleMIN-Gibbs, compared with vanilla Gibbs sampling.

Algorithm 4 MGPMH: Minibatch-Gibbs-Proposal MH

```
Require: Initial model $x \in \Omega$ and average batch size $\lambda$
  loop
    Sample $i$ uniformly from $\{1, \dots, n\}$.
    for all $\phi$ in $A[i]$ do
      Sample $s_\phi \sim$ Poisson $\left(\frac{\lambda M_\phi}{L}\right)$
    end for
    $S \leftarrow \{\phi | s_\phi > 0\}$
    for all $u$ in $\{1, \dots, D\}$ do
      $\epsilon_u \leftarrow \sum_{\phi \in S} \frac{s_\phi L}{\lambda M_\phi} \phi(x)$
    end for
    construct distribution $\psi(v) \propto \exp(\epsilon_v)$
    sample $v$ from $\psi$.
    construct update candidate $y \leftarrow x$; $y(i) \leftarrow v$
    compute the update probability
      $$a = \frac{\exp\left(\sum_{\phi \in A[i]} \phi(y)\right)}{\exp\left(\sum_{\phi \in A[i]} \phi(x)\right)} \cdot \frac{\exp\left(\epsilon_{x(i)}\right)}{\exp\left(\epsilon_{y(i)}\right)}.$$
    Update $x \leftarrow y$ with probability $\min(a, 1)$.
  end loop
```

$D|S|$), since we spend $O(\Delta)$ time computing the minibatch coefficients s_ϕ and the acceptance probability a, and we spend $O(D|S|)$ time computing the energy estimates ϵ_u. In expectation, the minibatch size $|S|$ will be

$$\mathbf{E}[|S|] \leq \sum_\phi s_\phi = \sum_\phi \frac{\lambda M_\phi}{L} = \lambda,$$

so our average runtime for an iteration will be $O(\lambda D + \Delta)$. If we want a guaranteed $O(1)$-factor-difference on the spectral gap, we need to set $\lambda = O(L^2)$. Doing this produces a final computation cost of $O(L^2 D + \Delta)$ for MGPMH. This can represent a significant speedup over the $O(D\Delta)$ complexity of vanilla Gibbs sampling.

Validation of MGPMH. We again turn to a synthetic example to validate the theoretical guarantees in Algorithm 4. We simulate a generalization of the Ising model known as the Potts model (Potts, 1952) with domain $\{1, \dots, D\}$. The energy of a configuration is the following:

$$\zeta_{\text{Potts}} = \sum_{i=1}^N \sum_{j=1}^N \beta \cdot A_{ij} \cdot \delta(x(i), x(j))$$

where the the δ function equals one whenever $x(i) = x(j)$ and zero otherwise. We simulate a graph with $n = N^2 = \Delta + 1 = 400$, $\beta = 4.6$, $D = 10$, and a fully connected configuration A_{ij} that depends on site distance in the same way as before. This model has $L = 5.09$ and $\Psi = 957.1$: note that $L^2 \ll \Delta$ for this model. We run the simulation for one million iterations and illustrate the error in the marginals of MGPMH and that of vanilla Gibbs sampling in Figure 2(b). We evaluate MGPMH for three average batch sizes λ, written in Figure 2(b) as multiples of the local maximum energy squared (L^2). MGPMH approaches vanilla Gibbs sampling as batch size increases, which validates Theorem 4.

Combining methods. Still, under some conditions, even MGPMH might be too slow. For example, Δ could be very large relative to $L^2 D$. In that setting, even though MGPMH would be faster than Gibbs sampling because it decouples the dependence on D and Δ, it still might be intractably slow. Can the decoupling of D and Δ from MGPMH be combined with the Δ-independence of MIN-Gibbs?

The most straightforward way to address this question is to replace the exact energy computation in the acceptance probability of MGPMH with a *second* minibatch approximation, like in MIN-Gibbs. Doing this combines the effects of MIN-Gibbs, which conceptually replaces Δ with Ψ^2 in the computational cost, and MGPMH, which conceptually replaces $D\Delta$ with $DL^2 + \Delta$. We call this algorithm *DoubleMIN-Gibbs* because of its doubly minibatched nature. It turns out DoubleMIN-Gibbs, Algorithm 5, has the same stationary distribution as MIN-Gibbs, and we can also prove a similar bound on its spectral gap.

Theorem 5. *The Markov chain described in Algorithm 5 is reversible and has the same stationary distribution (and therefore the same marginal stationary distribution) as MIN-Gibbs with the same estimator (as described in Theorem 1).*

Theorem 6. *Let $\bar{\gamma}$ be the spectral gap of DoubleMIN-Gibbs running with an energy estimator μ_x that has finite support*

Algorithm 5 DoubleMIN-Gibbs: Doubly-Minibatched

```
Require: Initial state $(x, \xi_x) \in \Omega \times R$
Require: Average first batch size $\lambda_1$
Require: Second minibatch estimators $\mu_x$ for $x \in \Omega$
  loop
    Sample $i$ uniformly from $\{1, \ldots, n\}$.
    for all $\phi$ in $A[i]$ do
      Sample $s_\phi \sim$ Poisson $\left(\frac{\lambda M_\phi}{L}\right)$
    end for
    $S \leftarrow \{\phi | s_\phi > 0\}$
    for all $u$ in $\{1, \ldots, D\}$ do
      $\epsilon_u \leftarrow \sum_{\phi \in S} \frac{s_\phi L}{\lambda M_\phi} \phi(x)$
    end for
    construct distribution $\psi(v) \propto \exp(\epsilon_v)$
    sample $v$ from $\psi$.
    construct update candidate $y \leftarrow x$; $y(i) \leftarrow v$
    sample $\xi_y \sim \mu_y$
    compute the update probability
        $$a = \exp\left(\xi_y - \xi_x + \epsilon_{x(i)} - \epsilon_{y(i)}\right).$$
    Update $(x, \xi_x) \leftarrow (y, \xi_y)$ with probability $\min(a, 1)$.
  end loop
```

and that satisfies, for some constant $\delta > 0$ and every $x \in \Omega$,

$$\mathbf{P}_{\xi_x \sim \mu_x}\left(|\xi_x - \zeta(x)| \leq \delta\right) = 1.$$

Let γ be the spectral gap of a MGPMH sampling chain running using the same graph and average batch size. Then,

$$\bar{\gamma} \geq \exp(-4\delta) \cdot \gamma.$$

Essentially, this theorem tells us the same thing Theorem 2 told us about MIN-Gibbs. As long as we can bound our estimator to be at most $O(1)$ away from the true energy $\zeta(x)$, convergence is slowed down by only at most a constant factor from MGPMH. By Lemma 2, this happens with high probability when we use an estimator of the same average batch size (for the second minibatch) as we used for MIN-Gibbs: $B = O(\Psi^2)$. Of course, this will only ensure an $O(1)$ difference from MGPMH; to ensure an $O(1)$ difference from vanilla Gibbs, we also need to invoke Theorem 4, which tells us we need to use an average batch size of $O(L^2)$ for the first minibatch. Along with the fact that the first minibatch sum needs to be computed D times, this results in an overall computational complexity of $O(L^2 D + \Psi^2)$.

One potential obstacle to an implementation actually achieving this asymptotic rate is the sampling of the Poisson random variables to select s_ϕ. Naively, since there are up to Δ potential values of ϕ, this could take up to $O(\Delta)$ time to compute. However, it is known to be possible to sample a (sparse) vector of Poisson random variables in time proportional to *the sum of their parameters* instead of the length of the vector. To illustrate, suppose we want to sample $x_1, \ldots, x_m$ independently where $x_i \sim \text{Poisson}(\lambda_i)$. To do this fast, first we notice if we let $B = \sum_{i=1}^m x_i$, then B will *also* be Poisson distributed, with parameter $\Lambda = \sum_{i=1}^m \lambda_i$. Conditioned on their sum B, the variables $x_1, \ldots, x_m$ have a *multinomial distribution* with trial count B and event probabilities $p_i = \lambda_i / \Lambda$. It is straightforward to sample from a multinomial distribution in time proportional to its trial count (ignoring log factors). Thus, we can sample $x_1, \ldots, x_m$ by first sampling $B \sim \text{Poisson}(\lambda)$ and then sampling $(x_1, \ldots, x_m) \sim \text{Multinomial}(B, (p_1, \ldots, p_m))$, and this entire process will only take on average $O(\Lambda)$ time.[7] For Algorithm 5, this means we can sample all the s_ϕ in average time $O(\lambda)$.[8] This is enough to confirm that its overall computational complexity is in fact $O(L^2 D + \Psi^2)$.

Validation of DoubleMIN-Gibbs. We evaluated the DoubleMIN-Gibbs algorithm on the same synthetic Potts model that we used for MGPMH. Figure 2(c) illustrates the performance of DoubleMIN-Gibbs with a batch size of L^2 for the first (MGPMH) minibatch, while the batch size of the second (MIN-Gibbs) minibatch, which we denote λ_2 in Figure 2(c), is adjusted to multiples of Ψ^2. As the second minibatch size increases, DoubleMIN Gibbs approaches the trajectory of MGPMH and vanilla Gibbs sampling, which is what we would expect from the result of Theorem 6.

4. Conclusion

We studied applying minibatching to Gibbs sampling. First, we introduced MIN-Gibbs, which improves the asymptotic per-iteration computational cost of Gibbs sampling from $O(D\Delta)$ to $O(D\Psi^2)$ in the setting where the total maximum energy Ψ^2 is small compared to the maximum degree Δ. Second, we introduced MGPMH, which has an asymptotic cost of $O(DL^2 + \Delta)$ and is an improvement in the setting where $L^2 \ll \Delta$. Finally, we combined the two techniques to produce DoubleMIN-Gibbs, which achieves a computational cost of $O(DL^2 + \Psi^2)$ and further improves over the other algorithms when $L^2 \leq \Psi^2 \ll \Delta$. We proved that all these algorithms can be made to be unbiased, and that these computation costs can be achieved with parameter settings that are guaranteed to have the same asymptotic convergence rate as plain Gibbs sampling, up to a constant-factor slowdown that can be made arbitrarily small. Our techniques will potentially enable graphical model inference engines that use Gibbs sampling to scale up to much larger and more complicated graphs than were previously possible, and could become a part of a future of highly scalable probabilistic inference on big data.

[7]This still requires $O(m)$ time to compute Λ and the probabilities p_i, but this can be computed once at the start of Algorithm 5, thereby not affecting the per-iteration computational complexity.

[8]This technique can also be used to speed up the other algorithms in this paper, yet it is not necessary to establish their asymptotic computational complexity.

Acknowledgments

The authors acknowledge the support of NSF DMS-1721550 and 1407557.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the NSF or the U.S. Government.

References

Bardenet, R., Doucet, A., and Holmes, C. Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *International Conference on Machine Learning*, pp. 405–413, 2014.

De Sa, C., Ré, C., and Olukotun, K. Ensuring rapid mixing and low bias for asynchronous Gibbs sampling. *International Conference on Machine Learning (ICML)*, pp. 1567–1576, 2016.

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

He, B., De Sa, C., Mitliagkas, I., and Ré, C. Scan order in Gibbs sampling: Models in which it matters and bounds on how much. *Advances in neural information processing systems (NIPS)*, 2016.

Ising, E. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik A Hadrons and Nuclei*, 31(1):253–258, 1925.

Johndrow, J. E., Mattingly, J. C., Mukherjee, S., and Dunson, D. Approximations of Markov chains and Bayesian inference. *arXiv preprint arXiv:1508.03387*, 2015.

Koller, D. and Friedman, N. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Korattikara, A., Chen, Y., and Welling, M. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *International Conference on Machine Learning (ICML)*, pp. 181–189, 2014.

Levin, D. A., Peres, Y., and Wilmer, E. L. *Markov chains and mixing times*. American Mathematical Soc., 2009.

Li, D. and Wong, W. H. Mini-batch tempered MCMC. *arXiv preprint arXiv:1707.09705*, 2017.

Lunn, D., Spiegelhalter, D., Thomas, A., and Best, N. The BUGS project: evolution, critique and future directions. *Statistics in medicine*, (25):3049–3067, 2009.

Maclaurin, D. and Adams, R. P. Firefly Monte Carlo: exact MCMC with subsets of data. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 543–552. AUAI Press, 2014.

McCallum, A., Schultz, K., and Singh, S. Factorie: Probabilistic programming via imperatively defined factor graphs. In *NIPS*, pp. 1249–1257, 2009.

Newman, D., Smyth, P., Welling, M., and Asuncion, A. U. Distributed inference for latent dirichlet allocation. In *Advances in neural information processing systems (NIPS)*, pp. 1081–1088, 2007.

Potts, R. B. Some generalized order-disorder transformations. In *Mathematical proceedings of the cambridge philosophical society*, volume 48, pp. 106–109. Cambridge University Press, 1952.

Quiroz, M., Tran, M.-N., Villani, M., Kohn, R., and Dang, K.-D. The block-Poisson estimator for optimally tuned exact subsampling MCMC. *ArXiv e-prints*, March 2018.

Rekatsinas, T., Chu, X., Ilyas, I. F., and Ré, C. Holoclean: Holistic data repairs with probabilistic inference. *Proceedings of the VLDB Endowment*, 10(11):1190–1201, 2017.

Seita, D., Pan, X., Chen, H., and Canny, J. F. An efficient minibatch acceptance test for metropolis-hastings. *Conference on Uncertainty in Artificial Intelligence (UAI)*, abs/1610.06848, 2017.

Smola, A. and Narayanamurthy, S. An architecture for parallel topic models. *Proceedings of the VLDB Endowment (PVLDB)*, 2010.

Theis, L., Sohl-dickstein, J., and Bethge, M. Training sparse natural image models with a fast Gibbs sampler of an extended state space. In *Advances in neural information processing systems (NIPS)*, pp. 1124–1132. 2012.

Zhang, C. and Ré, C. DimmWitted: A study of main-memory statistical analytics. *PVLDB*, 2014.