Fast Doubly-Adaptive MCMC to Estimate the Gibbs Partition Function with Weak Mixing Time Bounds

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present a novel method for reducing the computational complexity of rigorously estimating the *partition functions* of Gibbs (or Boltzmann) distributions, which arise ubiquitously in probabilistic graphical models. A major obstacle to applying the Gibbs distribution in practice is the need to estimate their partition function (normalizing constant). The state of the art in addressing this problem is multi-stage algorithms which consist of a cooling schedule and a mean estimator in each step of the schedule. While the cooling schedule in these algorithms is adaptive, the mean estimate computations use MCMC as a black-box to draw approximately-independent samples. Here we develop a *doubly adaptive* approach, combining the adaptive cooling schedule with an adaptive MCMC mean estimator, whose number of Markov chain steps adapts dynamically to the underlying chain. Through rigorous theoretical analysis, we prove that our method outperforms the state of the art algorithms in several factors: (1) The computational complexity of our method is smaller; (2) Our method is less sensitive to loose bounds on mixing times, an inherent components in these algorithms; and (3) The improvement obtained by our method is particularly significant in the most challenging regime of high precision estimates. We demonstrate the advantage of our method in experiments run on classic factor graphs, such as voting models and Ising models.

1 Introduction

The Gibbs (Boltzmann) distribution is a family of probability distributions of exponential form. First introduced in the context of statistical mechanics [20], Gibbs distributions are now ubiquitous in a variety of other disciplines, such as chemistry [26, 19], economics [47, 1] and machine learning. Gibbs distributions are typically used to model the global state of a system as a function of a collection of interdependent random variables, each representing local states in the system. The dependencies in the system is modeled by a *Hamiltonian* function, and the probability distribution is inversely proportional to exponent of the Hamiltonian (see § 1.1).

Gibbs distributions provide potent statistical inference tools in many machine learning applications, including restricted Boltzmann machines [56, 36], Markov Random Fields [33, 41], probabilistic graphical models [34, 43, 45], and Bayes Networks [27], which are used in the analysis of images and graphical data [55, 37, 18, 16], topic modeling (LDA) [23, 50, 46, 53], and more [10, 49, 2, 15, 14, 58, 55, 49, 21, 22, 25, 43, 45, 42].

A major obstacle in applying the Gibbs distribution in practice is the need to compute, or estimate, its *partition function* (normalizing constant). The partition function is defined over the Cartesian product of supports of a (typically) large number of variables, making exact computation intractable. Monte Carlo solutions for this problem has been extensively studied [31, 17, 57, 54, 29, 6, 35, 24]. As in many other Monte Carlo algorithms, the state of the art method for estimating the partition function is

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

a multi-stage algorithm, estimating a sequence of functions, such that the expectation of the product of these functions, or the product of the expectations of the functions, is the value of the partition function. Most of the research in this area focused on designing dynamically adaptive sequences that minimize both the length of the sequence (the *cooling schedule*), and the variance of each of the functions. In contrast, the computation of the sequence of mean estimates, which dominate the total computation cost, is done by black-box MCMC estimators, with a priori known upper bounds on the mixing times of the chains. Such a priori bounds are often loose, and improving them for particular models is a challenging active area of research [5, 54, 8, 9, 4]).

Instead, in this work we develop a *doubly adaptive* approach, combining the adaptive cooling schedule with adaptive MCMC mean estimator that dynamically adapt the number of Markov chain steps to the observed underlying chain. Through rigorous theoretical analysis, we prove that our method outperforms the state of the art algorithms in several factors: (1) The computational complexity of our method is smaller; (2) Our method is less sensitive to loose bounds on mixing times, an inherent component in these algorithms; and (3) The improvement obtained by our method is particularly significant in the most challenging regime of high precision estimates.

We demonstrate the advantage of our method in experiments run on classic factor graphs, such as voting and Ising models [12, 5, 7].

1.1 Preliminaries and Prior Work

Let Ω be a finite set, $H : \Omega \to \{0\} \cup [1, \infty)$ a function called *Hamiltonian*, and a parameter β, referred to as *inverse temperature*. The *Gibbs distribution* on Ω, H and β is:

$$\forall x \in \Omega : \ \pi_\beta(x) \doteq \frac{1}{Z(\beta)} \exp\big(-\beta H(x)\big), \quad \text{for } Z(\beta) \doteq \sum_{x \in \Omega} \exp\big(-\beta H(x)\big) \ . \tag{1}$$

The *normalizing constant* $Z(\beta)$ is known as the *partition function* of the distribution.

Estimating the partition function $Z(\beta)$ is computationally challenging since typically the size of Ω is exponential in the number of local variables, and the values of random terms in the sum have large variance. The following problem has been extensively studied, and is the focus of this paper:

Problem: Given a domain Ω, a Hamiltonian function H, and a parameter β, design a Fully Polynomial Randomized Approximation Scheme (FPRAS) for estimating the partition function $Z(\beta) \doteq \sum_{x \in \Omega} \exp\big(-\beta H(x)\big)$. In other words, for user-supplied ε, the task is to estimate $\hat{Z}(\beta)$ such that $(1-\varepsilon)Z(\beta) \leq \hat{Z}(\beta) \leq (1+\varepsilon)Z(\beta)$, in time polynomial in $\frac{1}{\varepsilon}$ and all other problem parameters.

All known scalable solutions to this problem rely on Monte Carlo Markov Chain (MCMC) methods, and their execution cost is dominated by the total number of Markov Chain steps they execute. We therefore, follow past work, and analyze our algorithms' complexity in terms of number of the Markov chain steps they execute. Building on extensive earlier work [17, 6, 57, 6], the current state of the art is the Huber and Schott's *TPA based paired product estimator* [30] with Kolmogorov's sharper analysis [35].

Huber and Schott's estimator applies *TPA based cooling schedule* to their *paired product estimator [29].* Assume a *cooling schedule* $\beta_0 < \beta_1, \cdots < \beta_{k-1} < \beta_k = \beta$, such that $Z(\beta_0)$ is easy to compute, thus $Z(\beta)$ can be estimated by estimating $Q \doteq {}^{Z(\beta)}/_{Z(\beta_0)}$. For each pair β_i, β_{i+1} in the schedule, we define two random variables, $f_{\beta_i,\beta_{i+1}} \doteq \exp(-\frac{\beta_{i+1}-\beta_i}{2} H(X_i))$ and $g_{\beta_i,\beta_{i+1}} \doteq (\frac{\beta_{i+1}-\beta_i}{2} H(Y_i))$, where $X_i \sim \pi_{\beta_i}, Y_i \sim \pi_{\beta_{i+1}}$, all independent. It is easy to verify that $\mathbb{E}[f_{\beta_i,\beta_{i+1}}] = Z(\frac{\beta_i+\beta_{i+1}}{2})/Z(\beta_i)$, and $\mathbb{E}[g_{\beta_i,\beta_{i+1}}] = Z(\frac{\beta_i+\beta_{i+1}}{2})/Z(\beta_{i+1})$. The *paired product estimators* are then $F = \prod_{i=1}^{k} f_{\beta_i,\beta_{i+1}}$, $G = \prod_{i=1}^{k} g_{\beta_i,\beta_{i+1}}$, and $Q = {}^{Z(\beta)}/_{Z(\beta_0)} = {}^{\mathbb{E}[F]}/_{\mathbb{E}[G]}$.

Denote by $\mathbb{V}_{\text{rel}}[X] \doteq {}^{\mathbb{E}[X^2]}/_{\mathbb{E}[X]^2} - 1 = {}^{\mathbb{V}[X]}/_{\mathbb{E}[X]^2}$ the *relative variance* of a random variable X. *The TPA schedule* [28, 30] is generated by an adaptive algorithm, which, by a proper setting of parameters, outputs a cooling schedule guaranteeing constant $\mathbb{V}_{\text{rel}}[F]$ and $\mathbb{V}_{\text{rel}}[G]$ (see alg. 3. in the supplementary material). Kolmogorov [35] presented a tighter analysis of Huber's TPA method, and proved that with slight modifications (see alg. 4. in the Appendix) the schedule has length $\Theta(\log Q \log H_{\max})$, while preserving constant relative variance for the paired product estimators. We denote the specific Kolmogorov's cooling schedule, that is also used in our algorithms, by TPA(k, d). For completeness, both of Huber's and Kolmogorv's versions of TPA are presented in the Appendix.

Kolmogorov [35] nearly matches known lower bounds when given *oracle access* to near-independent samples, but leaves open the possibility of better use of the dependent sequence of samples generated by MCMC chains. This fertile ground is ill-explored, since if an approximate sampling oracle draws samples by running a chain for T steps, there is a factor T potential improvement.

MCMC Mean-estimator: Huber and Schott [30] and Kolmogorov's [35] analyses assume a unit-cost oracle for sampling from π_{β_i}. Kolmogorov's [35] extended their analysis to include the complexity of generating the samples with standard MCMC processes, assuming a priori known upper bounds on their mixing time. The main contribution of our paper is a specialized, adaptive, *multiplicative* MCMC-mean estimator for the TPA-based paired product estimator that is significantly more efficient than using a standard MCMC for this problem. Thus, improving the best known method for estimating the partition function.

Let $\mathcal{M}$ be an ergodic Markov chain with state space S and stationary distribution π. Let $\tau_{\text{mix}}(\varepsilon)$ denote the ε-*mixing time* of $\mathcal{M}$, and define $\tau_{\text{mix}} \doteq \tau_{\text{mix}}(1/4)$. Let λ denote *second largest absolute eigenvalue* $\mathcal{M}$'s transition matrix. The *relaxation time* of $\mathcal{M}$ is $\tau_{\text{rx}} \doteq (1-\lambda)^{-1}$, and it is related to the mixing time τ_{mix}, by $\big(\tau_{\text{rx}}(\mathcal{M})-1\big)\ln(2) \leq \tau_{\text{mix}}(\mathcal{M}) \leq \tau_{\text{rx}}(\mathcal{M})\ \ln\big(2/\sqrt{\pi_{\min}}\big)$ [39]. We use T to denote an upper-bound on $\max(\tau_{\text{mix}}, \tau_{\text{rx}})$, and Λ to denote a bound on the second absolute eigenvalue λ. We use $\mathcal{G}_{H,\beta}$ to denote a chain with stationary distribution π_β.

Consider any i.i.d sampling concentration bound like Chebyshev's, Hoeffding or Bernstein type bound [44] with, say, complexity m_ε. Using MCMC as a black-box sampling tool, we obtain the same precision estimation having computational cost of $m_\varepsilon \cdot T$, where T should be taken as a known upper bound on $\tau_{\text{mix}}(\varepsilon/m_\varepsilon)$. Other concentration bounds compute the average over the entire trace of a Markov chain, and their complexity is dependent on *known upper-bounds* on *relaxation time* [48, 44, 40, 11, 32]. Note that since $\log(\frac{1}{2\varepsilon})(\tau_{\text{rx}}-1) \leq \tau_{\text{mix}}(\varepsilon) \leq \log(\frac{1}{\varepsilon\pi_{\min}})\tau_{\text{rx}}$, using these bounds is often more efficient, saving at least $\log(m_\varepsilon)$ steps. However, lack of tight upper bounds on relaxation time might hinder such benefits. More recent work in this line introduces *function specific mixing time* [51], which proves better computational cost, but requires a full spectral decomposition of the transition matrix of the chain.

Finally Cousins et al. [13] introduce the *trace averaging technique* and a new notion of variance, the *trace variance*. By using progressive sampling, their result requires little *a priori* knowledge of the complexity parameters, such as relaxation time and trace variance. We follow Cousins et al. approach here and tailor it to our setting, which requires developing new algorithms and analysis tools.

1.2 Our Main Contributions

- We present a specialized mean estimator method that significantly improves the state of the art computational complexity of computing the partition function of Gibbs distribution - a hard computational problem with numerous applications in statistical machine learning.
- While all rigorous MCMC based estimates depend on some a priori knowledge of the Markov chain properties (such as bounds on its mixing or relaxation time), the complexity of our method is less dependent on the tightness of these properties.
- The improvement of our method is particularly significant in the more challenging regime of computing estimates with very small error margin.
- Our method improves the computational cost by replacing the standard black-box MCMC mean estimators of prior work with an *adaptive MCMC estimator*, specially tailored for this problem.
- The analysis of our method relies on a novel notion of sample variance in a sequence of observations obtained by Markov chains runs which we term *the relative trace variance*.
- We demonstrate the practicality of our new method through experiments on different examples of Ising and voting models.

2 Algorithms

In this section, we combine the state of the art cooling schedule [29, 35] and the trace averaging technique [13] to design *doubly adaptive* algorithms for the Gibbs partition function. Our main algorithm SUPERCHAINTRACEGIBBS draws samples from a product Markov chain, $\mathcal{G}_\times$ and using a new *adaptive* technique, estimates the means of the paired product estimators. The paired product estimators are defined as $F = \prod_{i=1}^{l} f_{\beta_i,\beta_{i+1}}$ and $G = \prod_{i=1}^{l} g_{\beta_i,\beta_{i+1}}$, and $\mathcal{G}_\times$ is a *product chain* of

$\mathcal{G}_{H,\beta_i}$s, where the schedule $\{\beta_i\}_{i=1}^{l}$ is obtained by TPA(k,d) (where $k=\Theta(H_{\max})$ and $d=\Theta(1)$ are as derived in [35]). We denote the true relaxation time of each $\mathcal{G}_{H,\beta_i}$ by τ_i, and $\mathcal{G}_{\times}$'s relaxation time by τ_{prx}, τ is used to denote some arbitrary step number.

The core of SUPERCHAINTRACEGIBBS's functionality is our new *adaptive MCMC multiplicative mean estimator*, MEANESTIMATOR (alg. 1), whose complexity is predominantly characterized by our novel variance concept, the *the relative trace variance*. MEANESTIMATOR receives as input only a loose upper-bound on the chain's relaxation time, and the estimators' ranges. In thm. 2.4 we prove its efficiency, and using properties of *relative trace variance* we conclude two major advantages compared to classic MCMC mean estimators: (1) for any $\mathcal{M}$ with relaxation time τ_{rx}, we have $\text{Reltrv}_{\mathcal{M}}^{\tau_{\text{rx}}} \leq 2\mathbb{V}_{\text{rel}}$ (see lemma 2.2) thus we always beat the state of the art. (2) While the efficiency of the state of the art is deteriorated by poor bounds on mixing time, our algorithm is robust to that. Details and analysis of MEANESTIMATOR is presented in § 2.1.

After analysing the computation cost of SUPERCHAINTRACEGIBBS, we introduce the alternative method PARALLELTRACEGIBBS, which mitigates some issues present in SUPERCHAINTRACEGIBBS, and has different complexity tradeoffs. While SUPERCHAINTRACEGIBBS estimates *telescoping products* (F or G) with a *single chain* on the *product space*, PARALLELTRACEGIBBS instead estimates each term of the paired product estimators *in parallel* with *independent chains*. Because of their smaller ranges (derived in the appendix), estimating means of the terms $f_{\beta_i,\beta_{i+1}}$ and $g_{\beta_i,\beta_{i+1}}$ are easier using MEANESTIMATOR. However since we need tighter precisions (because of accumulative errors), we obtain worse complexity guarantees. Note that independently running $\mathcal{G}_{H,\beta_i}$ as opposed to their product, changes the mixing time terms from $\tau_{\text{prx}} = l\max_i \tau_i$ to $\sum_{i=1}^{l}\tau_i$, which is beneficial when $\sum_{i=1}^{l}\tau_i \ll l\max_i \tau_i$ i.e., when we have one slow chain which slows down the product chain.

Due to lack of the space, here we present only a pseudo-code of PARALLELTRACEGIBBS and present its efficiency in thm. 2.9, and the full analysis is presented in the supplementary material. The following lemma, whose proof is presented in the supplementary material, shows that all intervals produced in TPA are w.h.p. small, and is used later in the analysis.

Lemma 2.1. *Let $z(\beta) \doteq \log\big(Z(\beta)\big)$, d and k the parameters of the* TPA *method, and β_i and β_{i+1} two consecutive points generated by* TPA(k,d)*, we have:*

1. For any $\varepsilon \geq 0$, we have $\mathbb{P}(z(\beta_j) - z(\beta_{j+1}) \leq \varepsilon) \geq (1-\exp(-\varepsilon k/d))^d \simeq 1 - d\exp(-\varepsilon k/d)$,
2. For any $\varepsilon \geq 0$, $\mathbb{P}\left(\Delta_i \geq \varepsilon/\mathbb{E}[H(x)]\right) \leq d\exp(-\varepsilon k/d)$, where the expectation of $H(x)$ is taken with respect to distribution $x \sim \pi_{\beta_{i+1}}$.

2.1 Relative trace variance and MEANESTIMATOR

Both of the algorithms we present are based on a trace averaging technique which consists of running a Markov chain for, say, τ steps to generate $\vec{X}_{1:\tau} \doteq X_1, X_2, \ldots, X_\tau$ and taking $\bar{f}(\vec{X}_{1:\tau}) \doteq (\frac{1}{\tau})\sum_{i=1}^{\tau} f(X_i)$ as one sample, and repeat. We show in thm. 2.4 that using this technique, the complexity of MEANESTIMATOR is dominated, for small ε, by the true relaxation time and the *relative trace variance* defined below.

Definition 2.1 (Relative Trace variance)**.** *For arbitrary τ, consider a trace of length τ of a Markov chain $\mathcal{M}$, and a real valued function f. On $\mathcal{M}$, we define the relative trace variance of f as*

$$\text{Reltrv}_{\mathcal{M}}^{\tau}[f] \doteq \frac{\mathbb{E}[\bar{f}(\vec{X}_{1:\tau})^2]}{(\mathbb{E}[\bar{f}(\vec{X}_{1:\tau})])^2} - 1 \ ,$$

where $\vec{X}_{1:\tau} \doteq X_1, X_2, \ldots, X_\tau$ is a trace of length τ of $\mathcal{M}$, $\bar{f}(\vec{X}_{1:\tau}) \doteq (\frac{1}{\tau})\sum_{i=1}^{\tau} f(X_i)$. We may drop the subscript when the chain is clear from the context.

We follow the past work with *paired product estimators* F and G [29, 35]. Estimating means of the paired product estimators F and G (as opposed to each $f_{\beta_i,\beta_{i+1}}, g_{\beta_i,\beta_{i+1}}$ individually) leads to smaller complexity (because we no longer need a union bound). This approach requires bounding variances of F and G when the distribution is the Cartesian product of all π_{β_i}s. Thus the efficiency of estimators are analysed using *relative variance*, which unlike variance, can be computed as $\mathbb{V}_{\text{rel}}[F]+1 = \prod_{i=1}^{l}\left(\mathbb{V}_{\text{rel}}[f_{\beta_i,\beta_{i+1}}]+1\right)$ and $\mathbb{V}_{\text{rel}}[G]+1 = \prod_{i=1}^{l}\left(\mathbb{V}_{\text{rel}}[g_{\beta_i,\beta_{i+1}}]+1\right)$ for independent $f_{\beta_i,\beta_{i+1}}$ and $g_{\beta_i,\beta_{i+1}}$ (see e.g., Lemma 2 of [35] for details). We now develop similar machinery for the *relative trace variance*.

Lemma 2.2. *We have* $\mathrm{Reltrv}_{\mathcal{M}}^{\tau_{\mathrm{rx}}(\mathcal{M})}[f] \leq 2\mathbb{V}_{\mathrm{rel}}[f]$. *Furthermore, for* $\tau \geq \tau_{\mathrm{rx}}(\mathcal{M})$ *we have,* $\mathrm{Reltrv}_{\mathcal{M}}^{\tau}[f] = O\left(\frac{\tau_{\mathrm{rx}}(\mathcal{M})}{\tau}\mathrm{Reltrv}_{\mathcal{M}}^{\tau_{\mathrm{rx}}(\mathcal{M})}[f]\right)$.

Product chain [39]. Consider k Markov chains $\{\mathcal{M}_i\}_{i=1}^k$ each defined on state space S_i and assume real valued functions $\{f_i : S_i \to \mathbb{R}\}_{i=1}^k$. The *product chain* $\mathcal{M}_{1:k}^{\times}$ is defined on the Cartesian product of S_i as follows: at any step $\mathcal{M}_{1:k}^{\times}$ chooses i with probability ω_i (thus $\sum_{i=1}^k \omega_i = 1$), and moves from $(x_1, x_2, \ldots, x_i, \ldots, x_k)$ to $(x_1, x_2, \ldots, y_i, \ldots, x_k)$, with the transition probability of moving from x_i to y_i in $\mathcal{M}_i$. The *tensor product* of $\{f_i\}_{i=1}^k$, denoted by $\bigotimes_{1:k} f$, is defined as $\bigotimes_{1:k} f(x_1, x_2, \ldots, x_k) = \prod_{i=1}^k f_i(x_i)$.

We now state the following lemma whose proof is fully presented in the supplementary material and can be obtained by algebraic manipulations on the definition of relative trace variance and properties of product chains (that can be found in any MCMC textbook e.g., [39]).

Lemma 2.3. *Consider* k *Markov chains* $\{\mathcal{M}_j\}_{j=1}^k$ *and let* $\mathcal{M}_{1:k}^{\times}$ *(with any arbitrary* ω*) and* $\bigotimes_{1:k} f$ *be defined as above. Letting* τ_{p} *be the relaxation time of the product chain, we have:* $\mathrm{Reltrv}_{\mathcal{M}_{1:k}^{\times}}^{\tau_{\mathrm{p}}}[\bigotimes_{1:k} f_i] + 1 = \prod_{i=1}^k (\mathrm{Reltrv}_{\mathcal{M}_i}^{\tau_{\mathrm{p}}}[f_i] + 1)$.

Algorithm 1 MEANESTIMATOR

procedure MEANESTIMATOR
  Input Markov chain $\mathcal{M}$, upper-bound on second largest eigenvalue Λ, real valued function f with range $[a, b]$, letting $R = b - a$, multiplicative precision $\varepsilon_\times$, error probability δ.
  Output Multiplicative approximation $\hat{\mu}$ of $\mu = \mathbb{E}_\pi[f]$.
  $T \leftarrow \left\lceil \frac{1+\Lambda}{1-\Lambda} \ln \sqrt{2} \right\rceil$ & $\Lambda' \leftarrow \Lambda^T$ ▷ Choose T to be an upperbound on relaxation time and update Λ
  $I \leftarrow 1 \vee \left\lfloor \log_2 \left(\frac{bR}{2a^2} \cdot \frac{(1-\varepsilon_\times)^2}{(1+\varepsilon_\times)\varepsilon_\times} \right) \right\rfloor$; $\alpha \leftarrow \frac{(1+\Lambda')R \ln \frac{3I}{\delta}(1+\varepsilon_\times)}{(1-\Lambda')b\varepsilon_\times}$; ▷ Initialize *sampling schedule*
  for $i \in 1, 2, \ldots, I$ **do**
    $m_i \leftarrow \lceil \alpha 2^i \rceil$ ▷ Total sample count at iteration i
    for $j \in (m_{i-1} + 1), \ldots, m_i$ **do**
      $\vec{X}_{j,1} \leftarrow (T$ steps of $\mathcal{M}$ starting at $\vec{X}_{j-1,1})$
      $\vec{X}_{j,2} \leftarrow (T$ steps of $\mathcal{M}$ starting at $\vec{X}_{j-1,2})$ ▷ Run two independent copies of $\mathcal{M}$ for T steps
      $\bar{f}(\vec{X}_{j,1}) \leftarrow (\frac{1}{T}) \sum_{t=1}^T f(\vec{X}_{j,1}(t))$ & $\bar{f}(\vec{X}_{j,2}) \leftarrow (\frac{1}{T}) \sum_{t=1}^T f(\vec{X}_{j,2}(t))$
    end for
    $\hat{\boldsymbol{\mu}}_i \leftarrow (\frac{1}{2m_i}) \sum_{i=1}^{m_i} (f(\vec{X}_{j,1}) + f(\vec{X}_{j,2}))$ ▷ Empirical mean
    $\hat{v}_i \leftarrow (\frac{1}{2m_i}) \sum_{i=1}^{m_i} (f(\vec{X}_{j,1}) - f(\vec{X}_{j,2}))^2$ ▷ Empirical trace variance
    $u_i \leftarrow \hat{v}_i + \frac{(11+\sqrt{21})(1+{}^{\Lambda'}/_{\sqrt{21}})R^2 \ln \frac{3I}{\delta}}{(1-\Lambda')m_i} + \sqrt{\frac{(1+\Lambda')R^2 \hat{v}_i \ln \frac{3I}{\delta}}{(1-\Lambda')m_i}}$ ▷ Variance upper bound
    $\hat{\boldsymbol{\varepsilon}}_i^+ \leftarrow \frac{10R \ln \frac{3I}{\delta}}{(1-\Lambda')m_i} + \sqrt{\frac{(1+\Lambda')u_i \ln \frac{3I}{\delta}}{(1-\Lambda')m_i}}$ ▷ Apply Bernstein bound
    $\hat{\boldsymbol{\mu}}_i^\times \leftarrow \frac{(\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\varepsilon}}_i^+) \vee a + (\hat{\boldsymbol{\mu}}_i + \hat{\boldsymbol{\varepsilon}}_i^+) \wedge b}{2}$ ▷ Optimal mean estimate
    $\hat{\boldsymbol{\varepsilon}}_i^\times \leftarrow \frac{((\hat{\boldsymbol{\mu}}_i + \hat{\boldsymbol{\varepsilon}}_i^+) \wedge b - (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\varepsilon}}_i^+) \vee a}{2}$ ▷ Empirical relative error bound
    if $(i = I) \vee (\hat{\boldsymbol{\varepsilon}}_i^\times \leq \epsilon_\times)$ **then** ▷ Terminate if accuracy guarantee is met
      return $\hat{\boldsymbol{\mu}}_i^\times$
    end if
  end for
end procedure

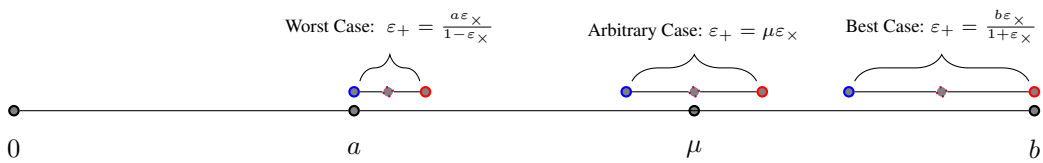
We now prove the correctness and bound the computational complexity of MEANESTIMATOR. As opposed to the *additive guarantee* of [13], here we show a *multiplicative guarantee* without *a priori*

knowledge of the true mean, we show MEANESTIMATOR requires a wider progressive sampling schedule. We present a proof sketch; full proof can be found in the supplementary material.

Theorem 2.4 (Efficiency and Correctness of MEANESTIMATOR). *With probability at least* $1-\delta$, MEANESTIMATOR *will output* $\hat{\mu}$ *satisfying* $(1-\varepsilon)\hat{\mu} \le \mu \le (1+\varepsilon)\hat{\mu}$. *Furthermore, with probability at least* $1-\frac{\delta}{3I}$, *the total Markov chain steps of* MEANESTIMATOR, $\hat{m}$, *obeys*

$$\hat{m} \in \mathcal{O}\left(\log\left(\frac{\log\frac{b}{a\varepsilon}}{\delta}\right)\left(\frac{R}{(1-\Lambda)\mu\varepsilon} + \frac{\tau_{\text{rx}}\text{Reltrv}^{\tau_{\text{rx}}}}{\varepsilon^2}\right)\right).$$

Proof Sketch. Suppose confidence interval $[a, b]$. The interval endpoints, multiplicative error $\varepsilon_\times$, and additive error ε_+ are related as $2\varepsilon_+ = a\frac{1+\varepsilon_\times}{1-\varepsilon_\times} - a = a\frac{2\varepsilon_\times}{1-\varepsilon_\times}$, depicted graphically below.



We derive a geometric progressive sampling schedule such that the algorithm draws sample sizes, ranging between optimistic and pessimistic (over unknown variance and mean) upper and lower bounds on the sufficient sample size.

Using the Markov chain Bennett inequality [32], the best-case complexity, assuming maximal expectation, and minimal variance, is

$$m^{\downarrow} \ge m_B(\Lambda, R, 0, \varepsilon_+, \frac{2\delta}{3I}) \ge \frac{(1+\Lambda)R\ln\frac{3I}{\delta}}{(1-\Lambda)\varepsilon_+} = \frac{(1+\Lambda)R\ln\frac{3I}{\delta}(1+\varepsilon_\times)}{b(1-\Lambda)\varepsilon_\times}.$$

The worst-case complexity, then assuming minimal expectation, and maximal variance, is

$$m^{\uparrow} \ge m_H(\Lambda, R, \varepsilon_+, \frac{2\delta}{3I}) \ge \frac{(1+\Lambda)R^2\ln\frac{3I}{\delta}}{2(1-\Lambda)\varepsilon_+^2} = \frac{(1+\Lambda)R^2\ln\frac{3I}{\delta}(1-\varepsilon_\times)^2}{2(1-\Lambda)a^2\varepsilon_\times^2},$$

via the Markov chain Hoeffding's inequality [38].

Consequently, a doubling schedule requires $I = \left\lfloor \log_2\left(\frac{m^{\uparrow}}{m^{\downarrow}}\right)\right\rfloor = \left\lfloor \log_2\left(\frac{bR}{2a^2}\cdot\frac{(1-\varepsilon_\times)^2}{(1+\varepsilon_\times)\varepsilon_\times}\right)\right\rfloor$ steps.

All tail bounds on variances and means are hold simultanously with probability at least $1-\delta$ (by union bound), and the doubling schedule never overshoots the sufficient sample size by more than a constant factor, which yields the stated guarantees. □

2.2 Doubly adaptive algorithms: SUPERCHAINTRACEGIBBSand PARALLELTRACEGIBBS

Let $(\beta_0, \beta_1, \ldots \beta_l)$ be a cooling schedule generated by TPA(k, d) (k and d chosen as in [35]), F, G the paired product estimators corresponding to this schedule, and $\mu = \mathbb{E}[F]$, $\nu = \mathbb{E}[G]$. Let $\mathcal{G}^\times$ be the Markov chain which is product of $\mathcal{G}_{H,\beta_i}$s with uniform weights ($\omega_i = 1/l, \forall i$), Λ a known upper bound on its second largest eigenvalue and τ_{prx} the true relaxation time of the product chain. The following whose full proof is presented in the supplementary material hold:

Theorem 2.5. *With probability at least* $1-\delta$, *it holds that total number of Markov chain steps of* SUPERCHAINTRACEGIBBS ,$\hat{m}$, *is upper bounded by*

$$\tilde{\mathcal{O}}\left(\log(\delta^{-1})\left(\frac{\text{Range}(F)/\mu + \text{Range}(G)/\nu}{(1-\Lambda)\varepsilon} + (\varepsilon^{-2})\cdot\tau_{\text{prx}}\cdot\left(\text{Reltrv}_{\mathcal{M}}^{\tau_{\text{prx}}}(F) + \text{Reltrv}_{\mathcal{M}}^{\tau_{\text{prx}}}(G)\right)\right)\right).$$

Lemma 2.6. *Defining* $\Delta = \beta_{\max} - \beta_{\min}$ *and* $\alpha_1 = \sqrt{\frac{Z(\beta_{\min})}{Z(\beta_{\min}-\Delta_{\max})}}$, *and letting* $H_{\max} \doteq \max_{x\in\Omega} H(x)$ *and* $H_{\min} \doteq \min_{x\in\Omega} H(x)$ *we have:*

$$\text{Range}(f)/\mu \le \sqrt{\frac{Q}{\exp(\Delta H_{\min})}}\,\alpha_1 \;\&\quad \text{Range}(g)/\nu \le \sqrt{\frac{\exp(\Delta H_{\max})}{Q}}\,\alpha_1.$$

Proof sketch. The proof requires approximations using the Taylor series, fundamental theorem of calculus, that the first derivative is negative and that the second derivative is positive. □

Finally, we use the following remark shown in [35].

Remark 2.7. *[35] Using* TPA(k, d), $k = \Theta(\log H_{\max})$ *and* $d = \Theta(1)$ *to generate cooling schedule* $(\beta_0, \beta_1, \dots \beta_l)$ *w.h.p. we have:* $l = \Theta(\log Q \log H_{\max})$ *and* $\mathbb{V}_{\mathrm{rel}}[F] + 1 = \prod_{i=1}^{l}(\mathbb{V}_{\mathrm{rel}}[f_{\beta_i,\beta_{i+1}}] + 1) = \Theta(1)$ *and* $\mathbb{V}_{\mathrm{rel}}[G] + 1 = \prod_{i=1}^{l}(\mathbb{V}_{\mathrm{rel}}[g_{\beta_i,\beta_{i+1}}] + 1) = \Theta(1)$.

These results culminate in the following bound.

Corollary 2.8. *Let* α_1, Δ, $H_{\min}$ *and* $H_{\max}$ *be as in lemma 2.6 and* $\tau_{\max} \doteq \max_i \tau_i$. *When* $\varepsilon \leq \frac{\tau_{\mathrm{prx}}}{(1-\Lambda)^{-1}} \left(\sqrt{\frac{\exp(\Delta H_{\min})}{Q}} + \sqrt{\frac{Q}{\exp(\Delta H_{\max})}} \right) / \alpha_1$ *the number of Markov chain steps of* SUPERCHAINTRACEGIBBS *is dominated by* $\tilde{O}\left(l\tau_{\max} \prod_{i=1}^{l}(\mathrm{Reltrv}_i / l + 1) - 1 \right) = \tilde{O}(l\tau_{\max})$.

Finally in this section we state the following theorem which shows the complexity of PARALLELTRACEGIBBS. PARALLELTRACEGIBBS runs MEANESTIMATOR each chain $\mathcal{G}_{H,\beta_i}$ and each of paired estimators independently and it is fully discussed in the supplementary material.

Theorem 2.9 (Efficiency of PARALLELTRACEGIBBS)**.** *For each* $1 \leq i \leq l$, *let* $\mathrm{RelR}_i \doteq \mathrm{Range}(f_{\beta_i,\beta_{i+1}})/\mu_i + \mathrm{Range}(g_{\beta_{i-1},\beta_i})/\nu_i$ *and* Λ_i *known bound on second largest eigenvalue of each* $\mathcal{G}_{H,\beta_i}$, *with true relaxation time* τ_i. *With probability at least* $1 - \delta$, *it holds that total Markov chain steps of* PARALLELTRACEGIBBS, $\hat{m}$, *is upper bounded by*

$$\tilde{\mathcal{O}}\left(\log\left(\frac{l}{\delta}\right) \left(\frac{l \cdot \sum_{i=1}^{l} \mathrm{RelR}_i}{(1-\Lambda_i)\varepsilon} + \frac{l^2}{\varepsilon^2} \sum_{i=1}^{l} \tau_i \cdot \left(\mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}(f_{\beta_i,\beta_{i+1}}) + \mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}(g_{\beta_{i-1},\beta_i}) \right) \right) \right) .$$

Furthermore, for all $1 \leq i \leq l$, $\mathrm{Range}(f_{\beta_i,\beta_{i+1}})/\mu_i \leq l^{1/\log(n)}$ *and* $\mathrm{Range}(g_{\beta_{i-1},\beta_i})/\nu_i \leq l^{\alpha_0(i)/\log n}$, *where* $\alpha_0(i) = (H_{\max}/2\mathbb{E}[H(x)]) - 1,\ x \sim \pi_{\beta_i}$.

Algorithm 2 doubly adaptive Gibbs partition function estimators

procedure SUPERCHAINTRACEGIBBS$(\dots)$
  $(\beta_0, \beta_1, \dots, \beta_l) =$ TPA$(k, d)^a$
  $\varepsilon' \leftarrow \frac{\varepsilon}{2+\varepsilon}$, & $\delta' \leftarrow \delta/2$.
  for $i \in 1, 2, \dots l$ **do**
    $f_i(x) = \exp(-\frac{\beta_{i+1}-\beta_i}{2} H(x))$
    $g_i(x) = \exp(\frac{\beta_i-\beta_{i-1}}{2} H(x))$
  end for
  $f = \bigotimes_{i=1}^{l} f_i$; $g = \bigotimes_{i=1}^{l} g_i$
  $\mathcal{G}^{\times} = \prod_{i=1}^{l} \mathcal{G}_{H,\beta_i}$, with $\omega(i) = 1/l, \forall i$
  $R_f = \exp(-\frac{\beta-\beta_0}{2} H_{\min}) - \exp(-\frac{\beta-\beta_0}{2} H_{\max})$
  $R_g = \exp(\frac{\beta-\beta_0}{2} H_{\max}) - \exp(\frac{\beta-\beta_0}{2} H_{\min})$
  $\mu =$ MEANESTIMATOR$(\mathcal{G}^{\times}, R_f, T, f, \varepsilon', \delta')$
  $\nu =$ MEANESTIMATOR$(\mathcal{G}^{\times}, R_g, T, g, \varepsilon', \delta')$
  return $Z = \frac{\nu}{\mu}$
end procedure

procedure PARALLELTRACEGIBBS$(\dots)$
  $(\beta_0, \beta_1, \dots, \beta_l) =$ TPA(k, d)
  $\varepsilon' \leftarrow \frac{\sqrt[l]{1+\varepsilon}-1}{\sqrt[l]{1+\varepsilon}+1}$ & $\delta' \leftarrow \delta/2l$
  for $i \in 1, 2, \dots l$ **do**
    $f_i(x) = \exp(-\frac{\beta_{i+1}-\beta_i}{2} H(x))$
    $g_{i-1}(x) = \exp(\frac{\beta_i-\beta_{i-1}}{2} H(x))$
    $R_f = \exp(-\frac{\beta_{i+1}-\beta_i}{2} H_{\min}) - \exp(-\frac{\beta_{i+1}-\beta_i}{2} H_{\max})$
    $R_g = \exp(\frac{\beta_{i+1}-\beta_i}{2} H_{\max}) - \exp(\frac{\beta_{i+1}-\beta_i}{2} H_{\min})$
    $\mu_i =$ MEANESTIMATOR$(\mathcal{G}_i, R_f, T_i, f_i, \varepsilon', \delta')$
    $\nu_i =$ MEANESTIMATOR$(\mathcal{G}_i, R_g, T_i, g_i, \varepsilon', \delta')$
  end for
  return $Z = \prod_{i=1}^{l} \frac{\nu_i}{\mu_i}$
end procedure

[a] k and d are as chosen in [35], thus $k = \Theta(\log H_{\max})$ and $d = 64$.

3 Experimental Results

In this section we report our experiment results, comparing the performance of the two versions of our *doubly adaptive* method (alg. 2), to the performance of the state of the art algorithm in [35].

Setup. We run the experiments using the single site Gibbs sampler (known also as Glauber dynamics) on two factor graph models:

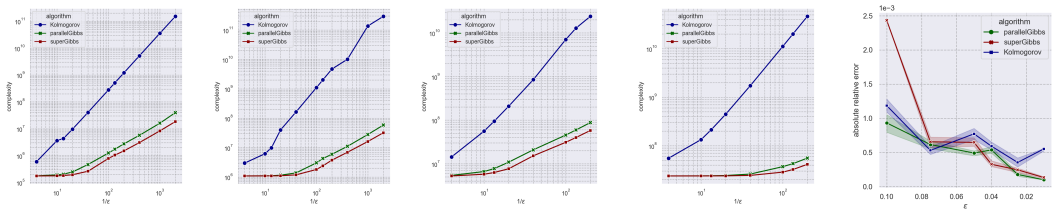
(A) the Ising model on 2D lattices. Having a 2-dimension lattice of size $n \times n$, the Hamiltonian is defined on n^2 random variables having values ± 1 and their dependency is represented by the Hamiltonian: $H(x) = -\sum_{(i,j)\in E} \mathbb{1}(x(i) = x(j))$. We run the algorithms on lattices of sizes 2×2,

3×3, 4×4, and 6×6. For each lattice, the parameter $\beta \geq 0$ is chosen below the critical inverse temperature at which it undergoes a phase transition. We use the bounds known for high temperature Ising models [3] (see fig. 1).

(B) the logical voting model. For a parameter n, we have $2n + 1$ random variables: the query variable $Q \in \{-1, 1\}$, and the voter variables $T_1, T_2, \ldots T_n$ and $F_1, F_2, \ldots F_n$ all in $\{0, 1\}$. The factors have $2n + 1$ weights, $\omega, \omega_{T_i}, \omega_{F_i}, i = 1, \ldots n$. The Hamiltonian is:

$$H(Q, T, F) = \omega Q \max_i T_i - \omega Q \max_i F_i + \sum_{i=1}^{n} \omega_{T_i} T_i + \sum_{i=1}^{n} \omega_{F_i} F_i \text{ , where } \omega, \omega_{T_i}, \omega_{F_i} \in [-1, 1]$$

The parameters are reported in fig. 2. We follow De Sa et al. [52] and use hierarchy width to derive upper bounds on mixing times.



(a) $\beta = .05$, 2×2 lattice (b) $\beta = .01$, 3×3 lattice (c) $\beta = .02$, 4×4 lattice (d) $\beta = .002$, 6×6 lattice (e) relative errors

Figure 1: Sample complexity for different algorithms on Ising models

To make a fair comparison, we always run the TPA algorithms once, and with the parameters given in [35]. At each iteration of MEANESTIMATOR, the sample size is extended at rate 1.1.

Reproducibility. The code is available from `git@github.com:XXXX/Doubly_Adaptive_MCMC.git`.

Results: Our experiments demonstrate the practical advantages of our *doubly adaptive method*, validating our theoretical analysis.

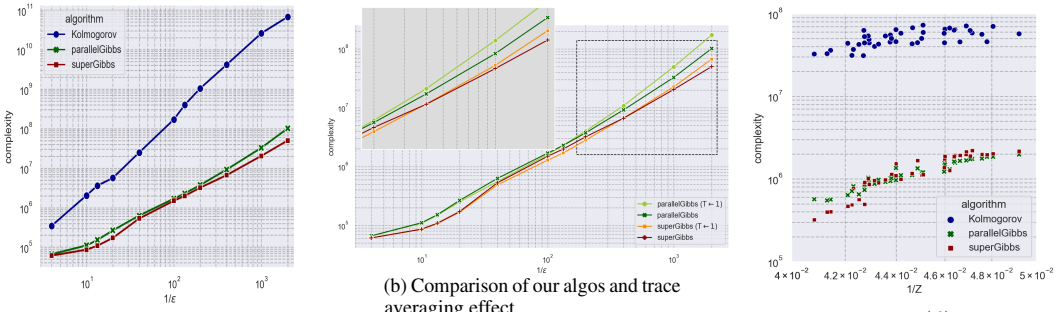
(1) We first compare the complexity of our algorithms to Kolmogorov's algorithm. Our experiments show the superiority of both versions of our methods on different models and various sets of parameters. Figure 1 demonstrates the superiority of our methods in the Ising model for various sets of parameters, and in figs. 2a and 2c for the voting model, when ε is fixed and $Z(\beta)$ is varying (fig. 2c) and when Z is fixed and ε is varying (fig. 2a). All of these hold while the precision of our algorithms beats [35] as $\varepsilon \to 0$ (fig. 1e).

(2) To demonstrate the advantage of using trace averaging (in contrast to just progressive sampling), and therefore relaying on relative trace variance instead relative variance, we run both of our algorithms using a simpler mean estimator which only uses progressive sampling. This is done by setting $T \leftarrow 1$ in line 4 of MEANESTIMATOR and we compare the results. In Figure 2b, we show the effectiveness of *trace averaging*, since both SUPERCHAINTRACEGIBBS and PARALLELTRACEGIBBS beat their simplified versions ($T \leftarrow 1$) after $1/\varepsilon$ passes a certain threshold. This is consistent for different parameters of the voting model.

(3) Comparing the performance of SUPERCHAINTRACEGIBBS and PARALLELTRACEGIBBS, we observe that in all of our experiments SUPERCHAINTRACEGIBBS has better performance than PARALLELTRACEGIBBS. In fig. 2b, we show the effect of trace averaging in PARALLELTRACEGIBBS becomes dominant earlier as $1/\varepsilon$ grows, thus it performs better in this perspective. This is consistent with our theoretical findings, because the ranges of estimators in PARALLELTRACEGIBBS are smaller than the ranges used in SUPERCHAINTRACEGIBBS.

4 Conclusions: advantages and limitations of proposed algorithms

We develop a doubly-adaptive MCMC-based estimator for the partition function of Gibbs distributions, which resolves a major impediment of prior methods that use MCMC as a black-box sampler. We show, both theoretically and experimentally, that our method requires substantially fewer MCMC steps than the state-of-the-art method. The better performance is due to several factors, which all stem from the use of an *adaptive MCMC mean estimator* instead of a standard "black-box" MCMC



(a) complexity vs. ε comparison against [35]

(b) Comparison of our algos and trace averaging effect

(c) complexity vs. $Z(\beta)$; $\varepsilon = 0.025$ comparison against [35]

Figure 2: Experiments on voting models. In (a) and (b) the parameters are $\beta = 0.1, n = 3, \omega = 0.9, \omega_T = \langle 0.2, 0.5, 0.1\rangle, \omega_F = -\langle 0.8, 0.2, 0.9\rangle$. In (c) we have $n = 5$ the weights and β are picked randomly to generate various values of $Z(\beta)$.

estimate. The complexity of the adaptive MCMC process depends on the tighter *trace* instead of the *stationary* relative variances, and on *relaxation times* instead of *mixing times*. It is also less sensitive to weak upper-bounds on mixing and relaxation times,

In particular, Kolmogorov's method requires $\Theta(l/\varepsilon^2)$ approximately independent samples, where l is the length of cooling schedule. This requires tight convergence (total variance distance of $O(\varepsilon^2/l)$ from stationary) for each sample, which adds a multiplicative $\ln l, (l = \Theta\left(\ln Q \ln H_{\max}\right))$ to its complexity (see column 3 of table 1 and [35], theorem 9). In contrast, our doubly adaptive method *only* depends on relaxation times, which do not depend on ε.

PARALLELTRACEGIBBS	SUPERCHAINTRACEGIBBS	TPA+ standard methods [35]
$l^2 \sum_{i=1}^{l} \tau_i \left(\mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}[f_i] + \mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}[g_i]\right)$	$\tau_{\mathrm{prx}}\left(\mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[F] + \mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[G]\right) = O\left(l\tau_{\max}\right)$	$\ln \frac{q \ln H_{\max}}{\varepsilon} \sum_{i=1}^{l} T_i \cdot \left(\mathbb{V}_{\mathrm{rel}}(F) + \mathbb{V}_{\mathrm{rel}}(G)\right) = O\left(\ln \frac{q \ln H_{\max}}{\varepsilon} \sum_{i=1}^{l} T_i\right)$

Table 1: Comparison number of Markov chain steps, when ε is adequately small. In all columns a multiplicative factor of $1/\varepsilon^2$ is omitted to ease presentation and $q = \ln Q$. The second line in the middle column follows from $\tau_{\mathrm{prx}} = l\tau_{\max}$ and that using the TPA schedule, $\mathbb{V}_{\mathrm{rel}}(F) + \mathbb{V}_{\mathrm{rel}}[G]$ is constant, and by lemma 2.2 we have $\mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[F] + \mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[G] = O\left(\mathbb{V}_{\mathrm{rel}}[F] + \mathbb{V}_{\mathrm{rel}}[G]\right)$.

Limitations. While providing significant improvement over the state of the art solution, our methods suffer from a several limitations. In SUPERCHAINTRACEGIBBS, the major limitation is the dependence on the *relative ranges* of F and G, which can be large, especially when the Hamiltonian range is large. Another issue is that the product chain's mixing time is dominated by $l \max\{\tau_i\}_{i=1}^{l}$, as opposed to $\sum_{i=1}^{l} \tau_i$. While PARALLELTRACEGIBBS circumvents these issues by estimating each the components of the products, it fails to beat SUPERCHAINTRACEGIBBS's efficiency in general, which is due to the union bound that require higher estimates guarantees for each component. Achieving an even better performance will probably require new estimators with smaller ranges and relative trace variances.

Statement of Broader Impact While probabilistic graphical models as other machine learning methods that rely on MCMC estimations continue to grow in importance and popularity. But running the MCMC to theoretical convergence guarantees is often prohibitively expensive, while running it to *apparent convergence* is methodologically unsound, particularly in the modern context, where public confidence in machine learning systems is continuously eroded by ethical, accuracy, and safety failures. Our work attempts to bridge the gap between the definite, elegant and theoretically sound analytic methods, and efficiency-focused practical utility, as we seek to reduce *proof-burden*, while maintaining theoretical guarantees of accuracy, with adaptive methods that bound efficiency in terms of (potentially unknown) convergence rate metrics and variances.

References

[1] Permit allocation in emissions trading using the boltzmann distribution. *Physica*, A 391:4883–4890, 2012.

[2] H. Afshar, S. Sanner, and Christfried Webers. Closed-form gibbs sampling for graphical models with algebraic constraints. In *AAAI*, 2016.

[3] David Aldous, Geoffrey R Grimmett, C Douglas Howard, Fabio Martinelli, J Michael Steele, and Laurent Saloff-Coste. *Probability on discrete structures*, volume 110. Springer Science & Business Media, 2013.

[4] Y. Alimohammadi, Nima Anari, Kirankumar Shiragur, and T. Vuong. Fractionally log-concave and sector-stable polynomials: Counting planar matchings and more. *ArXiv*, abs/2102.02708, 2021.

[5] Nima Anari, Kuikui Liu, and Shayan Oveis Gharan. Spectral independence in high-dimensional expanders and applications to the hardcore model. *arXiv preprint arXiv:2001.00303*, 2020.

[6] I. Bezáková, Daniel Stefankovic, V. Vazirani, and Eric Vigoda. Accelerating simulated annealing for the permanent and combinatorial counting problems. In *SODA 2006*, 2006.

[7] Nayantara Bhatnagar, Allan Sly, and Prasad Tetali. Reconstruction threshold for the hardcore model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 434–447. Springer, 2010.

[8] A. Blanca, P. Caputo, Z. Chen, D. Parisi, Daniel Stefankovic, and Eric Vigoda. On mixing of markov chains: Coupling, spectral independence, and entropy factorization. *ArXiv*, abs/2103.07459, 2021.

[9] Z. Chen, Andreas Galanis, Daniel Stefankovic, and Eric Vigoda. Rapid mixing for colorings via spectral independence. *ArXiv*, abs/2007.08058, 2020.

[10] H. Cheng, L. Qu, D. Garrick, and R. Fernando. A fast and efficient gibbs sampler for bayesb in whole-genome analyses. *Genetics, Selection, Evolution : GSE*, 47, 2015.

[11] Kai-Min Chung, Henry Lam, Zhenming Liu, and Michael Mitzenmacher. Chernoff-hoeffding bounds for markov chains: Generalized and simplified. *arXiv preprint arXiv:1201.0559*, 2012.

[12] Barry A Cipra. An introduction to the ising model. *The American Mathematical Monthly*, 94(10):937–959, 1987.

[13] Cyrus Cousins, Shahrzad Haddadan, and Eli Upfal. Making mean-estimation more efficient using an MCMC trace variance approach: Dynamite. *CoRR*, abs/2011.11129, 2020.

[14] Chris De Sa, Vincent Chen, and Wing Wong. Minibatch gibbs sampling on large graphical models. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1165–1173. PMLR, 10–15 Jul 2018.

[15] Christopher De Sa, Kunle Olukotun, and Christopher Ré. Ensuring rapid mixing and low bias for asynchronous gibbs sampling. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1567–1576. JMLR.org, 2016.

[16] H. Elliott, H. Derin, R. Cristi, and D. Geman. Application of the gibbs distribution to image segmentation. 9:678–681, 1984.

[17] G. S. Fishman. Choosing sample path length and number of sample paths when starting in steady state. *Oper. Res. Lett.*, 16:209–219, 1994.

[18] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984.

[19] Charles R Gibbs. Characterization and application of ferrozine iron reagent as a ferrous iron indicator. *Analytical Chemistry*, 48(8):1197–1201, 1976.

[20] Josiah Willard Gibbs. *Elementary Principles in Statistical Mechanics*. Charles Scribner's Sons., 1902.

[21] Joseph Gonzalez, Yucheng Low, Arthur Gretton, and Carlos Guestrin. Parallel gibbs sampling: From colored fields to thin junction trees. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 324–332, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[22] Joseph Gonzalez, Yucheng Low, Arthur Gretton, and Carlos Guestrin. Parallel gibbs sampling: From colored fields to thin junction trees. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 324–332, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[23] Tom Griffiths. Gibbs sampling in the generative model of latent dirichlet allocation. Technical report, 2002.

[24] David G. Harris and Vladimir Kolmogorov. Parameter estimation for gibbs distributions. *CoRR*, abs/2007.10824, 2020.

[25] Bryan He, Christopher De Sa, Ioannis Mitliagkas, and Christopher Ré. Scan order in gibbs sampling: Models in which it matters and bounds on how much. *Advances in neural information processing systems*, 29, 2016.

[26] Arnim Hellweg and Frank Eckert. Brick by brick computation of the gibbs free energy of reaction in solution using quantum chemistry and cosmo-rs. *AIChE Journal*, 63(9):3944–3954, 2017.

[27] Tomas Hrycej. Gibbs sampling in bayesian networks. *Artificial Intelligence*, 46(3):351–363, 1990.

[28] M. Huber and S. Schott. Using tpa for bayesian inference. 2010.

[29] Mark Huber. Approximation algorithms for the normalizing constant of gibbs distributions. *The Annals of Applied Probability*, 25(2):974–985, 2015.

[30] Mark Huber and Sarah Schott. Random construction of interpolating sets for high-dimensional integration. *Journal of Applied Probability*, 51(1):92–105, 2014.

[31] Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

[32] Bai Jiang, Qiang Sun, and Jianqing Fan. Bernstein's inequality for general markov chains. *arXiv preprint arXiv:1805.10721*, 2018.

[33] John G Kemeny, J Laurie Snell, and Anthony W Knapp. *Denumerable Markov chains: with a chapter of Markov random fields by David Griffeath*, volume 40. Springer Science & Business Media, 2012.

[34] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[35] V. Kolmogorov. A faster approximation algorithm for the gibbs partition function. *ArXiv*, abs/1608.04223, 2018.

[36] Oswin Krause, Asja Fischer, and Christian Igel. Algorithms for estimating the partition function of restricted boltzmann machines. *Artificial Intelligence*, 278:103195, 10 2019.

[37] Patricio S La Rosa, Terrence L Brooks, Elena Deych, Berkley Shands, Fred Prior, Linda J Larson-Prior, and William D Shannon. Gibbs distribution for statistical analysis of graphical data with a sample application to fcmri brain images. *Statistics in medicine*, 35(4):566—580, February 2016.

[38] Carlos Leon and François Perron. Optimal hoeffding bounds for discrete reversible markov chains. *The Annals of Applied Probability*, 14, 05 2004.

[39] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[40] Pascal Lezaud. Chernoff-type bound for finite markov chains. *The Annals of Applied Probability*, 8, 08 1998.

[41] Xianghang Liu and Justin Domke. Projecting markov random field parameters for fast mixing. NIPS'14, page 1377–1385, Cambridge, MA, USA, 2014. MIT Press.

[42] Thomas A Best N. Lunn D, Spiegelhalter D. The bugs project: Evolution, critique and future directions. *Stat Med.*, 28(25):3049-67, 2009 Nov 10.

[43] Andrew McCallum, Karl Schultz, and Sameer Singh. Factorie: Probabilistic programming via imperatively defined factor graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009.

[44] Michael Mitzenmacher and Eli Upfal. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[45] Joris M. Mooij and Soon Ong. libdai: A free/open source c++ library for discrete approximate inference methods, 2008.

[46] David Newman, Padhraic Smyth, Max Welling, and Arthur Asuncion. Distributed inference for latent dirichlet allocation. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

[47] Marco Patriarca, Anirban Chakraborti, and Kimmo Kaski. Gibbs versus non-gibbs distributions in money dynamics. *Physica A: Statistical Mechanics and its Applications*, 340(1):334–339, 2004. News and Expectations in Thermostatistics.

[48] Daniel Paulin. Concentration inequalities for markov chains by marton couplings and spectral methods. *Electron. J. Probab.*, 20, 2015.

[49] M. Plummer. Jags: A program for analysis of bayesian graphical models using gibbs sampling. 2003.

[50] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, page 569–577, New York, NY, USA, 2008. Association for Computing Machinery.

[51] Maxim Rabinovich, Aaditya Ramdas, Michael Jordan, and Martin Wainwright. Function-specific mixing times and concentration away from equilibrium. *Bayesian Analysis*, 15, 05 2016.

[52] C. D. Sa, Ce Zhang, K. Olukotun, and C. Ré. Rapidly mixing gibbs sampling for a class of factor graphs using hierarchy width. *Advances in neural information processing systems*, 28:3079–3087, 2015.

[53] Alexander Smola and Shravan Narayanamurthy. An architecture for parallel topic models. *Proc. VLDB Endow.*, 3(1–2):703–710, September 2010.

[54] D. Stefankovic, S. Vempala, and E. Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 183–193, 2007.

[55] Lucas Theis, Jascha Sohl-Dickstein, and Matthias Bethge. Training sparse natural image models with a fast gibbs sampler of an extended state space. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1124–1132, Red Hook, NY, USA, 2012. Curran Associates Inc.

[56] Christopher Tosh. Mixing rates for the alternating gibbs sampler over restricted boltzmann machines and friends. In *ICML*, 2016.

[57] Daniel Štefankovič, Santosh Vempala, and Eric Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3), May 2009.

[58] Ruqi Zhang and C. D. Sa. Poisson-minibatching for gibbs sampling with convergence rate guarantees. In *NeurIPS*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We discuss some limitations of our main algorithm in the beginning of section 2 and later in conclusion (section 4) of the paper, we dedicated the second to last paragraph (the **Limitations** paragraph) to discussing limitations .
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] We see no negative impacts because our work suggests substituting heuristic methods with fully analyzed algorithms with proven performance guarantees, and we highlight positive impacts in the **Statement of Broader Impact** (last paragraph).
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] all of our assumptions are state fully in the paper and separately in each theorem or lemma
   (b) Did you include complete proofs of all theoretical results? [Yes] Due to lack of space intuitions or proof sketches are given in the main paper, and all of the proofs are provided in full detail in the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We attached the code to this submission, and will release the URL in the camera ready (due to anonymity). Implementation details and parameter values are presented in the experimental section.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] Our paper proposes a fpras and its errors bounds are proved theoretically (see theorems 2.4, 2.5 and 2.9) and experimentally (see figure 1.e).
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A] Our contribution is mostly theoretical, thus our experiments are run with small parameters which do not need specialized computational resources. We report the total amount of computation (the number of MCMC steps) in our experiments, which can be fairly compared across algorithms and computational architectures.