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

Shahrzad Haddadanⓡ[*]
Brown University
The Data Science Initiative
shahrzad.haddadan@gmail.com

Yue Zhuangⓡ
Brown University
The Data Science Initiative
yue_zhuang1@brown.edu

Cyrus Cousins ⓡ
Brown University
Department of Computer Science
cyrus_cousins@brown.edu

Eli Upfal
Brown University
Department of Computer Science
eliezer_upfal@brown.edu

Abstract

We present a novel method for reducing the computational complexity of rigorously estimating the *partition functions* (normalizing constants) of Gibbs (Boltzmann) distributions, which arise ubiquitously in probabilistic graphical models.

A major obstacle to practical applications of Gibbs distributions is the need to estimate their partition functions. 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 estimation computations use MCMC as a black-box to draw approximate samples. 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 component in these algorithms; and (3) The improvement obtained by our method is particularly significant in the most challenging regime of high-precision estimation. 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 [25], Gibbs distributions are now ubiquitous in a variety of other disciplines, such as chemistry [24, 31], economics [1, 54] 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

[*] ⓡ indicates randomized ordering and equal contribution.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

in the system are modeled by a *Hamiltonian* function, and the probability distribution is inversely proportional to exponent of the Hamiltonian scaled by the *temperature* (see eq. (1) § 1.1).

Gibbs distributions provide potent statistical inference tools in many machine learning applications. They appear in probabilistic graphical models [41, 49, 51], including restricted Boltzmann machines [43, 66], Markov random fields [40, 47], and Bayes networks [32], and are applied in the analysis of images and graphical data [21, 23, 44, 65], topic modeling (LDA) [27, 53, 57, 62], and more [2, 13, 18, 19, 26, 30, 48, 56, 58, 68].

A major obstacle in applying the Gibbs distribution in practice is the need to compute, or estimate, its *partition function* (normalizing constant), henceforth written GPF. The partition function is defined over the Cartesian product of supports of a (typically large) number of variables, making exact computation intractable. Furthermore, due to interdependence of variables, exact sampling is not practically feasible, thus Markov-chain Monte-Carlo (MCMC) solutions for this problem have been extensively studied [6, 22, 29, 34, 37, 39, 42, 52, 63, 67].

Like other MCMC methods, here various heuristics are used. The most well-known heuristics are the *annealed importance sampling* [39, 52, 64] or *convergence diagnostics methods* [10, 11, 20, 61]. Unfortunately, these methods are often error-prone, as their correctness is only proven asymptotically, without rigorous mathematical analysis to bound their estimation error with *finite samples*. In fact, theoretical findings have shown that with no prior knowledge of relevant measures, such as the variance of importance weights in annealed importance sampling, or upper bounds on mixing or relaxation times for diagnostic methods, these methods are either unreliable or computationally intractable (see [52, section 4] or [8, 33]).

On the other hand, theoreticians study this problem by designing Fully Polynomial Randomized Approximation Schemes (FPRAS) (see problem 1). The state of the art FPRAS for estimating the GPF is a multi-stage algorithm involving a sequence of functions at various temperatures, such that the expectation of the product of these functions, or the product of the expectations of said functions, is the GPF. FPRAS's are proven to produce (approximate) solutions w.h.p., but their performance guarantees rely on available upper-bounds on various measures such as variances of estimators or mixing times of Markov chains. In static algorithms, these upper-bounds are given *a priori*, and adaptive[2] algorithms estimate them dynamically, while increasing the sample size until desired properties are mathematically guaranteed. Thus, adaptive algorithms are less sensitive to looseness of known upper-bounds, more robust, often faster, and more easily applied to various settings.

Most of the research on designing FPRAS's for the GPF is focused on designing adaptive algorithms to produce sequences (*cooling schedules*) with minimum length while keeping the variances of estimators small (thus removing the need to have a-priori known bounds on variances). In contrast, the computation of the sequence of mean estimates, which dominates the total computation cost, is done by black-box MCMC estimators, with *a priori* known upper bounds on the mixing times of the chains. These upper bounds are often loose, and improving them for particular models is a challenging active area of research [4, 5, 9, 12, 28, 63]. In order to complement the adaptive cooling schedule and reduce dependence on *a priori* bounds on Markov chains' mixing times, it seems necessary to design an *adaptive* procedure with theoretical guarantees for MCMC-mean estimation.

In this work we develop a *doubly adaptive* FPRAS, combining the adaptive cooling schedule with adaptive MCMC mean estimator that dynamically adapts 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 [5, 7, 15].

1.1 Preliminaries and Prior Work

Assume a *sample space* Ω, *Hamiltonian function* $H : \Omega \to \{0\} \cup [1, \infty)$, and *inverse temperature* parameter $\beta \in \mathbb{R}$, referred to as *inverse temperature*. The *Gibbs distribution* on Ω, $H(\cdot)$, and β is

[2] The usage of the word "adaptive" here refers to algorithms which draw samples progressively and adapt their sample complexity based on empirical estimates until desired conditions are met, as it has been used in [34, 42] (see § 1.1), and should not confused with the work of [60].

then characterized by probability law

$$\forall x \in \Omega : \; \pi_\beta(x) \doteq \frac{1}{Z(\beta)} \exp\big(-\beta H(x)\big) \ . \tag{1}$$

Here $Z(\beta)$ is the *normalizing constant* or *Gibbs partition function* (GPF) of the distribution, with

$$Z(\beta) \doteq \sum_{x \in \Omega} \exp\big(-\beta H(x)\big) \ . \tag{2}$$

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

Problem 1. *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 produce an estimate $\hat{Z}(\beta)$, such that with probability at least $1 - \delta$, we have $(1 - \varepsilon)Z(\beta) \leq \hat{Z}(\beta) \leq (1 + \varepsilon)Z(\beta)$, in time polynomial in ${}^{1}/_{\varepsilon}$, $\ln({}^{1}/_{\delta})$, and all other problem parameters (e.g., the number of vertices in an Ising model, or neurons in an RBM).*

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 in terms of number of the Markov chain steps.

TPA-Based Adaptive Cooling Schedules Building on extensive earlier work [6, 22, 67], the current state of the art is due to Huber and Schott [35], with Kolmogorov's sharper analysis [42]. They introduce the *paired product estimator* (PPE), see definition 1.1, and apply the *tootsie-pop algorithm* (TPA) to adaptively compute a near-optimal *cooling schedule*, i.e., a sequence of inverse temperatures $\beta_0 < \beta_1 < \cdots < \beta_{\ell-1} < \beta_\ell$ satisfying $\beta_\ell = \beta$, and that $Z(\beta_0)$ is easy to compute, e.g., $\beta_0 = 0$ is often convenient, since $Z(0) = |\Omega|$. We thus define $Q \doteq {}^{Z(\beta)}/_{Z(\beta_0)}$ and estimate it using the paired product estimator.

Definition 1.1 (PPE [34]). *Assume a* cooling schedule $\beta_0, \beta_1, \ldots, \beta_\ell$. *For each pair (β_i, β_{i+1}) in the schedule, we define two random variables, $X_i \sim \pi_{\beta_i}$ and $Y_i \sim \pi_{\beta_{i+1}}$, all independent, and we then define $f_{\beta_i,\beta_{i+1}} \doteq \exp\big(-\frac{\beta_{i+1}-\beta_i}{2} H(X_i)\big)$ and $g_{\beta_i,\beta_{i+1}} \doteq \exp\big(\frac{\beta_{i+1}-\beta_i}{2} H(Y_i)\big)$. 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 $\mathbb{E}[g_{\beta_i,\beta_{i+1}}] = Z(\frac{\beta_i+\beta_{i+1}}{2})/Z(\beta_{i+1})$. We then define $F \doteq \prod_{i=1}^{k} f_{\beta_i,\beta_{i+1}}$, $G \doteq \prod_{i=1}^{k} g_{\beta_i,\beta_{i+1}}$. Letting $\hat{\mu}$ and $\hat{\nu}$ denote empirical estimates of $\mathbb{E}[F]$ and $\mathbb{E}[G]$, respectively, the* paired product estimator *(PPE) is $\hat{Q} \doteq {}^{\hat{\mu}}/_{\hat{\nu}}$.*

Denote by $\mathbb{V}_{\mathrm{rel}}[X] \doteq {}^{\mathbb{E}[X^2]}/_{\mathbb{E}[X]^2} - 1 = {}^{\mathbb{V}[X]}/_{\mathbb{E}[X]^2}$ the *relative variance* of a random variable X. *The TPA schedule* [35, 36] is generated by an adaptive algorithm, which, by a proper setting of parameters, outputs a cooling schedule guaranteeing constant $\mathbb{V}_{\mathrm{rel}}[F]$ and $\mathbb{V}_{\mathrm{rel}}[G]$ (see alg. 3 in the supplementary material). Kolmogorov [42] presents a tighter analysis of Huber's TPA method, and proves that with slight modifications (see alg. 4. in the Appendix) the schedule has a shorter length, while preserving constant relative variance for the paired product estimators (see thm. 1.1). In this paper, we use Kolmogorov's algorithm, and we denote it by TPA(k, d). For completeness, both of Huber's and Kolmogorov's versions of TPA are presented in the Appendix.

We will use the following result in our analysis:

Theorem 1.1 ([42]). *Let $H_{\max} \doteq \max_{x \in \Omega} H(x)$, using* TPA$(k, d)$, *$k = \Theta(\log H_{\max})$ and $d = 16$ to generate cooling schedule $(\beta_0, \beta_1, \ldots, \beta_\ell)$. W.h.p., we have $\ell = \Theta(\log(Q)\log(H_{\max}))$ and $\mathbb{V}_{\mathrm{rel}}[F]+1 = \prod_{i=1}^{\ell}(\mathbb{V}_{\mathrm{rel}}[f_{\beta_i,\beta_{i+1}}]+1) = \Theta(1)$ and $\mathbb{V}_{\mathrm{rel}}[G]+1 = \prod_{i=1}^{\ell}(\mathbb{V}_{\mathrm{rel}}[g_{\beta_i,\beta_{i+1}}]+1) = \Theta(1)$.*

Kolmogorov [42] 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 [35] assume unit-cost for exact sampling from each π_{β_i}, and Kolmogorov [42] extends their analysis to include the complexity of generating *approximate samples* with standard MCMC processes, assuming *a priori* upper-bounds on their mixing times. The main contribution of our paper is a specialized, adaptive, *multiplicative* MCMC-mean estimator for the TPA-based PPE. Our method is significantly more efficient than using standard black-box MCMC sampling for this problem, thus we improve the best-known method for estimating the GPF.

Let $\mathcal{M}$ be an ergodic Markov chain with state space S and stationary distribution π. Let $\tau_{\mathrm{mix}}(\varepsilon)$ denote the ε*-mixing time* of $\mathcal{M}$, and define $\tau_{\mathrm{mix}} \doteq \tau_{\mathrm{mix}}({}^1\!/\!{}_4)$. Letting λ denote the *second largest absolute eigenvalue* of $\mathcal{M}$'s transition matrix, the *relaxation time* of $\mathcal{M}$ is $\tau_{\mathrm{rx}} \doteq (1-\lambda)^{-1}$, and it is related to the mixing time τ_{mix}, by $\big(\tau_{\mathrm{rx}}(\mathcal{M})-1\big)\ln(2) \leq \tau_{\mathrm{mix}}(\mathcal{M}) \leq \big\lceil \tau_{\mathrm{rx}}(\mathcal{M}) \ln\big({}^2\!/\!\sqrt{\pi_{\min}}\big)\big\rceil$ [45]. Let T be an upper bound on $\max\{\tau_{\mathrm{rx}}(\mathcal{M}), \tau_{\mathrm{mix}}(\mathcal{M})\}$.

Consider any i.i.d. sampling concentration bound like Chebyshev's, Hoeffding's, or Bernstein's inequalities [50], with, say, sample complexity m_ε. Using MCMC as a black-box sampling tool, we obtain the same precision estimation guarantees, with a computational cost of $m_\varepsilon \cdot \tau_{\mathrm{mix}}(\varepsilon/m_\varepsilon)$, which is equal to $m_\varepsilon \log(m_\varepsilon \cdot \varepsilon^{-1}) \cdot T$ in the absence of exact values for τ_{mix}.

Other concentration bounds compute the average over the entire trace of a Markov chain, and their complexity is dependent on *known upper-bounds* on the *relaxation time* [14, 38, 46, 50, 55], or function specific mixing time [59]. Note that since $\log(\frac{1}{2\varepsilon})(\tau_{\mathrm{rx}}-1) \leq \tau_{\mathrm{mix}}(\varepsilon) \leq \log(\frac{1}{\varepsilon\pi_{\min}})\tau_{\mathrm{rx}}$, using these bounds is often more efficient, saving at least $\log(m_\varepsilon)$ steps.

Recently, Cousins et al. [16] introduce a novel Markov chain statistical measure, the *inter-trace variance*. The inter-trace variance depends on both the *function being estimated* and the *dependency structure* between nearby samples in the chain, and unlike the mixing time, it can be efficiently estimated from data. By using progressive sampling, Cousins et al. show an *additive MCMC mean estimator* whose complexity is proved in terms of *inter-trace variance* and they show it it less sensitive to *prior* knowledge of the input parameters, such as relaxation time and trace variance. Unfortunately due to a few technical problems, their result can not directly be used with the TPA method. Thus, in order to obtain a doubly adaptive algorithm for problem 1, we tailor their techniques 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.
- 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 these *a priori* bounds, and decays gracefully as they become looser.
- The improvement of our method is particularly significant in the more challenging *high precision* regime, where the goal is to compute estimates with very small multiplicative error.
- Our method improves the computational cost of prior work by replacing standard black-box MCMC mean estimators with an *adaptive MCMC estimator*, specially tailored to 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 method through experiments on Ising and voting models.

2 Algorithms

In this section, we develop two *doubly-adaptive fully polynomial randomized approximation schemes* providing more efficient algorithmic solutions to problem 1. The proof of all of the lemmas and theorems are presented fully in the supplementary material.

Notation and Setting Parameters We use the following notation throughout: We use capital letters to denote upper-bounds. e.g., T denotes an upper-bound on $\max(\tau_{\mathrm{mix}}, \tau_{\mathrm{rx}})$, and Λ denotes a upper-bound on the second absolute eigenvalue λ. We use $\mathcal{G}_{H,\beta}$ to denote any Markov chain with Gibbs stationary distribution π_β, eq. (1). Having the Hamiltonian H, we denote its maximum and minimum values as $H_{\max}$ and $H_{\min}$, i.e., $H_{\max} \doteq \max_{x\in\Omega}\{H(x)\}$ and $H_{\min} \doteq \min_{x\in\Omega}\{H(x)\}$. Having a schedule $(\beta_0, \beta_1, \ldots, \beta_\ell)$, the paired product estimators $f_{\beta_i,\beta_{i+1}}$, $g_{\beta_i,\beta_{i+1}}$, $F = \bigotimes_{i=1}^{\ell} f_{\beta_i,\beta_{i+1}}$ and

$G = \bigotimes_{i=1}^{\ell} g_{\beta_i,\beta_{i+1}}$ are as in definition 1.1. When writing $(\beta_0, \beta_1, \dots, \beta_\ell) = \textsc{Tpa}(k, d)$, we mean the cooling schedule is obtained from running alg. 4 in the Appendix, and we always set $k = \log H_{\max}$ and $d = 64$, as these parameters are shown to produce a near-optimal schedule w.h.p. [42].

We first introduce a novel MCMC-based *multiplicative* mean estimation procedure RelMeanEst (see alg. 1), and analyze its computational complexity in terms of a new quantity, which we coin *the relative trace variance* (see definition 2.1). RelMeanEst receives as input a Markov chain $\mathcal{M}$, a function f, and precision parameters ε and δ, and it outputs a multiplicative estimate of the expected value of the function w.r.t. the stationary distribution of the Markov chain. For simplicity, we may refer to it as RelMeanEst$(\mathcal{M}, f)$, leaving out the precision parameters.

Letting $(\beta_0, \beta_1, \dots, \beta_\ell) = \textsc{Tpa}(k, d)$, we first present ParallelTraceGibbs, in which we invoke both RelMeanEst$(\mathcal{G}_{H,\beta_i}, f_{\beta_i,\beta_{i+1}})$ and RelMeanEst$(\mathcal{G}_{H,\beta_i}, g_{\beta_i,\beta_{i+1}})$ for each $i = 1, 2, \dots, \ell - 1$. We then present an often-more-efficient algorithm, SuperChainTraceGibbs, which invokes RelMeanEst once each on F and G on a *"super" product chain* (see definition 2.2). We prove correctness of both ParallelTraceGibbs and SuperChainTraceGibbs, and bound their complexity in terms of *the relative trace variance* of the estimators. Furthermore, we prove SuperChainTraceGibbs improves the computational complexity of the state of the art [42] (thm. 2.4 and corollary 2.6). Both of these algorithms have low dependence on tightness of mixing time: They receive as input an upper-bound on mixing or relaxation time T, but we show for $\varepsilon \geq \varepsilon_0$ their computation complexity is dominated by the *true relaxation time* τ_{rel} (of each Gibbs chain or the product chain).

2.1 Relative trace variance and RelMeanEst

In this section we introduce a new variance notion, the *relative trace variance*, which captures the computational complexity of MCMC-mean estimation with *multiplicative* precision guarantees. The *relative trace variance* depends on both the chain $\mathcal{M}$ and the function f, and it generalizes the *relative variance*, defined as $\mathbb{V}_{\mathrm{rel}}[f] \doteq \mathbb{V}[f]/\mathbb{E}[f]^2$, which depends only on f, and is used in i.i.d. regimes.

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*

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

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

The above definition is similar to what Cousins et al. coined as *the inter-trace variance*, denoted by $\mathrm{trv}^{(\tau)}(\mathcal{M}, f)$, which they showed it captures MCMC-mean estimation with *additive* precision guarantees [16]. In fact, the two terms are related as

$$\mathrm{Reltrv}_{\mathcal{M}}^{\tau}[f] = \frac{\mathrm{trv}^{(\tau)}(\mathcal{M}, f)}{(\mathbb{E}[\bar{f}(\vec{X}_{1:\tau})])^2} \ .$$

Note that the two terms are not easily convertible without knowing the *mean*, $\mathbb{E}[\bar{f}(\vec{X}_{1:\tau})]$.

Lemma 2.1. *For any τ we have*

$$\mathrm{Reltrv}_{\mathcal{M}}^{\tau}[f] \leq \mathbb{V}_{\mathrm{rel}}[f] \ . \tag{3}$$

Furthermore, for $\tau \geq \tau_{\mathrm{rx}}(\mathcal{M})$ we have,

$$\mathrm{Reltrv}_{\mathcal{M}}^{\tau}[f] = O\left(\frac{\tau_{\mathrm{rx}}(\mathcal{M})}{\tau} \mathrm{Reltrv}_{\mathcal{M}}^{\tau_{\mathrm{rx}}(\mathcal{M})}[f]\right) \ . \tag{4}$$

Lemma 2.1 enables us to compare the computational complexity of our algorithms with the state of the art [42]. In particular, using (3), we show our results improve the state of the art (which is in terms of $\mathbb{V}_{\mathrm{rel}}$), and using (4), we show that for high-precision estimations, the sample complexity of our algorithms only depends on τ_{rx}, which improves the state of the art (which is in term of T).

The relative trace variance is a better analysis tool for estimating the GPF, because, unlike the inter-trace variance, it leads directly to *relative error bounds*, rather than *absolute error bounds*.We now present some definitions which can also be found in standard MCMC textbooks, e.g., [45].

Definition 2.2 (Product Chain and Tensor Product Function)**.** *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 *$\mathcal{M}_{1:k}^{\otimes}$ is defined on the Cartesian product of S_i as follows: at any step $\mathcal{M}_{1:k}^{\otimes}$ 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 $\bigotimes_{1:k} f_i$, is defined as $\big(\bigotimes_{1:k} f_i\big)(x_1, x_2, \ldots, x_k) = \prod_{i=1}^k f_i(x_i)$.*

RELMEANEST Let T denote an upper bound on the relaxation time of a Markov chain $\mathcal{M}$. RELMEANEST receives T, $\mathcal{M}$, f and precision parameters ε and δ as input. Before it starts collecting samples, it runs the chain for a *warm start* (§ 2.1 of alg. 1). Starting from a minimum sample size $m^{\downarrow}$, it runs $\mathcal{M}$ for $T \cdot m^{\downarrow}$ steps, and collect samples $X_1, X_2, \ldots, X_{T \cdot m^{\downarrow}}$. It then computes for $j = 1, 2, \ldots, m^{\downarrow}$, $\bar{f}_j \doteq \sum_{i=(j-1)\cdot T+1}^{j \cdot T} f(X_i)$; using them, it calculates an empirical estimate of the mean, $\hat{\mu}$, and an empirical estimation for the *trace variance* of $\mathcal{M}$ and f, $\hat{v}$. Based on these estimates, we derive an upper-bound on the current trace variance u_i and relative error $\hat{\varepsilon}_i^{\times}$, and check whether is smaller than the user-specified error ε (lines 18-19). If so, we return the current mean estimate, otherwise we double the sample size and repeat.

Algorithm 1 RELMEANEST

```
 1: procedure RELMEANEST
 2:   Input: Markov chain $\mathcal{M}$, upper-bound on relaxation time $T$, real-valued function $f$ with range $[a, b]$, letting $R = b - a$,
      multiplicative precision $\varepsilon$, error probability $\delta$.
 3:   Output: Multiplicative approximation $\hat{\mu}$ of $\mu = \mathbb{E}_\pi[f]$.
 4:   $T \leftarrow \left\lceil \frac{1+\Lambda}{1-\Lambda} \ln \sqrt{2} \right\rceil$; $\Lambda' \leftarrow \Lambda^T$      ▷ Choose $T$ to be an upperbound on relaxation time
 5:   $I \leftarrow 1 \vee \left\lfloor \log_2 \left( \frac{bR}{2a^2} \cdot \frac{(1-\varepsilon)^2}{(1+\varepsilon)\varepsilon} \right) \right\rfloor$; $\alpha \leftarrow \frac{(1+\Lambda')R \ln \frac{3I}{\delta}(1+\varepsilon)}{(1-\Lambda')b\varepsilon}$; $m_0 \leftarrow 0$      ▷ Initialize sampling schedule
 6:   $T_{\mathrm{unif}} \leftarrow \lceil T \cdot \ln(1/\pi_{\min}) \rceil$; $(\vec{X}_{0,1}, \vec{X}_{0,2}) \leftarrow \mathcal{M}^{T_{\mathrm{unif}}}(\perp)$      ▷ Warm-start two chains for $T_{\mathrm{unif}}$ steps from arbitrary $\perp \in \Omega$
 7:   for $i \in 1, 2, \ldots, I$ do
 8:     $m_i \leftarrow \lceil \alpha r^i \rceil$      ▷ Total sample count at iteration $i$; $r$ is the geometric ratio (constant, usually 2) size
 9:     for $j \in (m_{i-1} + 1), \ldots, m_i$ do
10:       $(\vec{X}_{j,1}, \vec{X}_{j,2}) \leftarrow$ ($T$ steps of $\mathcal{M}$ starting at $\vec{X}_{j-1,1}, \vec{X}_{j-1,2}$)      ▷ Run two independent copies of $\mathcal{M}$ for $T$ steps
11:       $\bar{f}(\vec{X}_{j,1}) \leftarrow \frac{1}{T} \sum_{t=1}^{T} f\big(\vec{X}_{j,1}(t)\big)$; $\bar{f}(\vec{X}_{j,2}) \leftarrow \frac{1}{T} \sum_{t=1}^{T} f\big(\vec{X}_{j,2}(t)\big)$      ▷ Average $f$ over $T$-traces
12:     end for
13:     $\hat{\boldsymbol{\mu}}_i \leftarrow \frac{1}{2m_i} \sum_{i=1}^{m_i} \big(f(\vec{X}_{j,1}) + f(\vec{X}_{j,2})\big)$; $\hat{v}_i \leftarrow \frac{1}{2m_i} \sum_{i=1}^{m_i} \big((f(\vec{X}_{j,1}) - f(\vec{X}_{j,2})\big)^2$      ▷ Compute empirical mean; trace variance
14:     $u_i \leftarrow \hat{v}_i + \frac{(11 + \sqrt{21})(1 + \Lambda'/\sqrt{21})R^2 \ln \frac{3I}{\delta}}{(1 - \Lambda')m_i} + \sqrt{\frac{(1 + \Lambda')R^2 \hat{v}_i \ln \frac{3I}{\delta}}{(1 - \Lambda')m_i}}$      ▷ Variance upper bound
15:     $\hat{\boldsymbol{\varepsilon}}_i^{+} \leftarrow \frac{10R \ln \frac{3I}{\delta}}{(1 - \Lambda')m_i} + \sqrt{\frac{(1 + \Lambda')u_i \ln \frac{3I}{\delta}}{(1 - \Lambda')m_i}}$      ▷ Apply Bernstein bound
16:     $\hat{\boldsymbol{\mu}}_i^{\times} \leftarrow \frac{(\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\varepsilon}}_i^{+}) \vee a + (\hat{\boldsymbol{\mu}}_i + \hat{\boldsymbol{\varepsilon}}_i^{+}) \wedge b}{2}$      ▷ Optimal mean estimate
17:     $\hat{\boldsymbol{\varepsilon}}_i^{\times} \leftarrow \frac{((\hat{\boldsymbol{\mu}}_i + \hat{\boldsymbol{\varepsilon}}_i^{+}) \wedge b - (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\varepsilon}}_i^{+}) \vee a}{2\hat{\boldsymbol{\mu}}_i^{\times}}$      ▷ Empirical relative error bound
18:     if $(i = I) \vee (\hat{\boldsymbol{\varepsilon}}_i^{\times} \leq \varepsilon)$ then      ▷ Terminate if accuracy guarantee is met
19:       return $\hat{\boldsymbol{\mu}}_i^{\times}$
20:     end if
21:   end for
22: end procedure
```

The following theorem, shows the correctness of RELMEANEST and bounds its complexity.

Theorem 2.2 (Efficiency and Correctness of RELMEANEST)**.** *With probability at least $1 - \delta$,* RELMEANEST *will output $\hat{\mu}$ satisfying $(1 - \varepsilon)\hat{\mu} \leq \mu \leq (1 + \varepsilon)\hat{\mu}$. Furthermore, with probability at least $1 - \frac{\delta}{3I}$, the total Markov chain steps of* RELMEANEST*, $\hat{m}$, obeys*

$$\hat{m} \in \mathcal{O}\left(\ln \left(\frac{\ln \frac{b}{a\varepsilon}}{\delta} \right) \left(\frac{T \cdot R}{\mu\varepsilon} + \frac{\tau_{\mathrm{rx}} \mathrm{Reltrv}^{\tau_{\mathrm{rx}}}}{\varepsilon^2} \right) \right). \tag{5}$$

2.2 Doubly adaptive algorithms: SuperChainTraceGibbs and ParallelTraceGibbs

Let $(\beta_0, \beta_1, \ldots, \beta_\ell) = \textsc{Tpa}(k, d)$, and consider a family of Gibbs chains $\mathcal{G}_{H,\beta_i}$, each corresponding to some β_i, and the paired product estimators $F = \bigotimes_{i=1}^{\ell} f_{\beta_i,\beta_{i+1}}$ $G = \bigotimes_{i=1}^{\ell} g_{\beta_i,\beta_{i+1}}$. The Tpa method is designed to ensure $\mathbb{V}_{\text{rel}}$ of the estimators are bounded, which can be employed by concentration bounds (e.g., Chebyshev's bound) to guarantee the multiplicative error is bounded with high probability for a given sample size.

In order to generalize the same machinery for samples generated from a Markov chain using RelMeanEst, we need to bound the two terms appearing in eq. (5), which dominate the computational complexity of RelMeanEst. We refer to the first term, $T \cdot R/\mu$, as the *range term*, and to the term $\tau_{\text{rx}}\text{Reltrv}^{\tau_{\text{rx}}}$ as the *trace variance term*. Note that as ε becomes smaller, the *trace variance* term dominates the sample complexity of RelMeanEst, thus dependence on loose bounds T and R is dominated by dependence on *true and a priori unknown* values τ_{rx} and $\text{Reltrv}^{\tau_{\text{rx}}}$.

In order to ensure that the ranges of estimators are small, we prove that the length of each inverse-temperature interval in the Tpa schedule is w.h.p. small. Having a schedule $(\beta_0, \beta_1, \ldots, \beta_\ell)$ we define and use the following notation: for $0 \leq i \leq \ell - 1$, *interval length* $\Delta_i \doteq \beta_{i+1} - \beta_i$, *maximum interval length* $\Delta_{\max} \doteq \max_i \Delta_i$, and *total length* $\Delta \doteq \beta_\ell - \beta_0$.

Lemma 2.3. *Let $z(\beta) \doteq \ln\big(Z(\beta)\big)$, and let β_i, β_{i+1} be two consecutive points generated by* $\textsc{Tpa}(k, d)$*. For arbitrary $\varepsilon \geq 0$, we have:*

1. $\mathbb{P}(z(\beta_i) - z(\beta_{i+1}) \leq \varepsilon) \geq (1 - \exp(-\varepsilon k/d))^d$.

2. $\mathbb{P}\left(\Delta_i \geq {}^{\varepsilon}/_{\mathbb{E}[H(x)]}\right) \leq d\exp(-\varepsilon k/d)$*, where $\mathbb{E}[H(x)]$ is taken w.r.t. $x \sim \pi_{\beta_{i+1}}$.*

SuperChainTraceGibbs Let $\mathcal{G}^{\otimes}$ the product of $\mathcal{G}_{H,\beta_i}$s with uniform weights i.e., $\omega_i = \frac{1}{\ell}, \forall i$ (see definition 2.2). SuperChainTraceGibbs calls $\textsc{RelMeanEst}(\mathcal{G}^{\otimes}, F)$ and $\textsc{RelMeanEst}(\mathcal{G}^{\otimes}, G)$, with appropriate parameters, and simply outputs the ratio of the two estimates (see alg. 2, left).

Algorithm 2 SuperChainTraceGibbs and ParallelTraceGibbs

```
procedure SuperChainTraceGibbs(...)
  $(\beta_0, \beta_1, \ldots, \beta_\ell) \leftarrow \textsc{Tpa}(k, d)^a$
  $\varepsilon' \leftarrow \frac{\varepsilon}{2+\varepsilon}; \delta' \leftarrow \frac{\delta}{2}$
  for $i \in 1, 2, \ldots, \ell$ do
    $f_i(x) \doteq \exp(-\frac{\beta_{i+1}-\beta_i}{2}H(x))$
    $g_i(x) \doteq \exp(\frac{\beta_i-\beta_{i-1}}{2}H(x))$
  end for
  $F \doteq \bigotimes_{i=1}^{\ell} f_i; G \doteq \bigotimes_{i=1}^{\ell} g_i$
  $\mathcal{G}^{\otimes} \leftarrow \bigotimes_{i=1}^{\ell} \mathcal{G}_{H,\beta_i}$, with $\omega_i = \frac{1}{\ell}, \forall i$
  $R_f \leftarrow \exp(-\frac{\beta-\beta_0}{2}H_{\min}) - \exp(-\frac{\beta-\beta_0}{2}H_{\max})$
  $R_g \leftarrow \exp(\frac{\beta-\beta_0}{2}H_{\max}) - \exp(\frac{\beta-\beta_0}{2}H_{\min})$
  $\hat{\mu} \leftarrow \textsc{RelMeanEst}(\mathcal{G}^{\otimes}, R_f, T, F, \varepsilon', \delta')$
  $\hat{\nu} \leftarrow \textsc{RelMeanEst}(\mathcal{G}^{\otimes}, R_g, T, G, \varepsilon', \delta')$
  return $\hat{Z} \leftarrow \frac{\hat{\nu}}{\hat{\mu}}$
end procedure

procedure ParallelTraceGibbs(...)
  $(\beta_0, \beta_1, \ldots, \beta_\ell) = \textsc{Tpa}(k, d)$
  $\varepsilon' \leftarrow \frac{\sqrt[\ell]{1+\varepsilon}-1}{\sqrt[\ell]{1+\varepsilon}+1}; \delta' \leftarrow \frac{\delta}{2\ell}$
  for $i \in 1, 2, \ldots \ell$ do
    $f_i(x) \doteq \exp(-\frac{\beta_{i+1}-\beta_i}{2}H(x))$
    $g_{i-1}(x) \doteq \exp(\frac{\beta_i-\beta_{i-1}}{2}H(x))$
    $R_f \leftarrow \exp(-\frac{\beta_{i+1}-\beta_i}{2}H_{\min}) - \exp(-\frac{\beta_{i+1}-\beta_i}{2}H_{\max})$
    $R_g \leftarrow \exp(\frac{\beta_{i+1}-\beta_i}{2}H_{\max}) - \exp(\frac{\beta_{i+1}-\beta_i}{2}H_{\min})$
    $\hat{\mu}_i \leftarrow \textsc{RelMeanEst}(\mathcal{G}_i, R_f, T_i, f_i, \varepsilon', \delta')$
    $\hat{\nu}_i \leftarrow \textsc{RelMeanEst}(\mathcal{G}_i, R_g, T_i, g_i, \varepsilon', \delta')$
  end for
  return $\hat{Z} \leftarrow \prod_{i=1}^{\ell} \frac{\hat{\nu}_i}{\hat{\mu}_i}$
end procedure
```

[a] $k = \Theta(\log H_{\max})$ and $d = 64$ as in [42]

We now show the correctness and efficiency of SuperChainTraceGibbs. Let τ_{prx} denote $\mathcal{G}^{\otimes}$'s true (and unknown) relaxation time and T a known upper-bound on it ($T \geq \tau_{\text{prx}}$), ε and δ are user specified precision parameters. For simlicity of presentation we use the following notation to refer to relative ranges: $\text{relR} = \text{Range}(F)/\mu + \text{Range}(G)/\nu$, where $\mu = \mathbb{E}[F]$ and $\nu = \mathbb{E}[G]$.

Theorem 2.4. *With probability at least $1 - \delta$, it holds that the total number $\hat{m}$ of Markov chain steps taken by* SuperChainTraceGibbs *is upper-bounded by*

$$\tilde{\mathcal{O}}\left(\ln\left(\frac{1}{\delta}\right)\left(\frac{T \cdot \text{relR}}{\varepsilon} + \frac{\tau_{\text{prx}} \cdot \left(\text{Reltrv}_{\mathcal{G}^{\otimes}}^{\tau_{\text{prx}}}(F) + \text{Reltrv}_{\mathcal{G}^{\otimes}}^{\tau_{\text{prx}}}(G)\right)}{\varepsilon^2}\right)\right).$$

Lemma 2.5. *Defining $\alpha_1 = \sqrt{\frac{Z(\beta_0)}{Z(\beta_0 - \Delta_{\max})}}$, we have:* $\frac{\text{Range}(F)}{\mu} \leq \alpha_1\sqrt{\frac{Q}{\exp(\Delta H_{\min})}}$ *and* $\frac{\text{Range}(G)}{\nu} \leq \alpha_1\sqrt{\frac{\exp(\Delta H_{\max})}{Q}}$.

ParallelTraceGibbs	SuperChainTraceGibbs	Tpa + PPE [42]
$\ell^2 \sum_{i=1}^{\ell} \tau_i \Big(\mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}(f_i) +$	$\tau_{\mathrm{prx}} \left(\mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[F] + \mathrm{Reltrv}^{\tau_{\mathrm{prx}}}[G] \right)$	$\ln \frac{q \ln H_{\max}}{\varepsilon} \sum_{i=1}^{\ell} T_i \cdot (\mathbb{V}_{\mathrm{rel}}(F) + \mathbb{V}_{\mathrm{rel}}(G))$
$\mathrm{Reltrv}^{\tau_i}_{\mathcal{G}_{H,\beta_i}}(g_i) \Big)$	$= O\left(\ell \max\{\tau_i\}_{i=1:\ell}\right)$	$= O\left(\ln \frac{q \ln H_{\max}}{\varepsilon} \sum_{i=1}^{\ell} T_i \right)$

Table 1: Comparison of the 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 $q = \ln Q$. Note that computational complexity of both ParallelTraceGibbs and SuperChainTraceGibbs only depends on true relaxation times, denoted by τ_i, and the Tpa + PPE method's complexity is dependent on their upper bounds, denoted by T_i.

Using lemma 2.5 and thm. 2.4, we identify ε_0 such that for $\varepsilon \leq \varepsilon_0$ the trace variance term in will the dominate computational complexity of SuperChainTraceGibbs. In order to make a fair comparison with the state of the art [42] we employ eq. (3) of lemma 2.1. Finally we use thm. 1.1 and conclude:

Corollary 2.6. *Let* α_1 *be as in lemma 2.5,* $\tau_{\max} \doteq \max_i \tau_i$ *and* $\varepsilon_0 \doteq (\tau_{\mathrm{prx}}/T) \cdot \left(\sqrt{\frac{\exp(\Delta H_{\min})}{Q}} + \sqrt{\frac{Q}{\exp(\Delta H_{\max})}} \right) \cdot \alpha_1$. *When* $\varepsilon \leq \varepsilon_0$*, the number of Markov chain steps of* SuperChainTraceGibbs *is dominated by* $\tilde{O}(\ell \tau_{\max})$.

ParallelTraceGibbs For $i = 1, 2, \ldots, \ell - 1$, ParallelTraceGibbs (alg. 2, right) runs RelMeanEst$(\mathcal{G}_{H,\beta_i}, f_{\beta_i,\beta_{i+1}})$ and RelMeanEst$(\mathcal{G}_{H,\beta_i}, g_{\beta_i,\beta_{i+1}})$ independently. We show the computational complexity of ParallelTraceGibbs in thm. 2.7.

For $i = 1, 2, \ldots, \ell$, assume τ_i is the true (unknown) relaxation time of $\mathcal{G}_{H,\beta_i}$ and T_i is a known bound on it. For simplicity of presentation we use the following notations: $\mathrm{relR}_i \doteq \mathrm{Range}(f_{\beta_i,\beta_{i+1}})/\mu_i + \mathrm{Range}(g_{\beta_{i-1},\beta_i})/\nu_i$, where $\mu_i = \mathbb{E}(f_{\beta_i,\beta_{i+1}})$ and $\nu_i = \mathbb{E}(g_{\beta_i,\beta_{i+1}})$.

Theorem 2.7 (Efficiency of ParallelTraceGibbs)**.** *With probability at least* $1 - \delta$*, it holds that the total number* $\hat{m}$ *of Markov chain steps taken by* ParallelTraceGibbs *is upper-bounded by*

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

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

ParallelTraceGibbs and SuperChainTraceGibbs make different computational complexity tradeoffs. ParallelTraceGibbs is usually slower than SuperChainTraceGibbs, because in each iteration $i = 1, 2, \ldots, \ell$, the mean estimator must acquire a higher-precision estimate so that *all estimators together* achieve an ε-δ relative-error guarantee. Relaxation times (true values and their upper-bounds) appear *in a sum* in the complexity of ParallelTraceGibbs, whereas they appear *in a maximum* in SuperChainTraceGibbs ($\sum_{i=1}^{\ell} \tau_i$ vs. $\max_{i=1,\ldots,\ell} \tau_i$). Furthermore, dominance of the trace variance terms in both of these algorithms occur at different values of ε. A comparison of the complexity of these algorithms, in the high-precision regime, with Kolmogorov's Tpa + PPE (which uses MCMC as a black box) is presented in table 1.

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 [42].

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

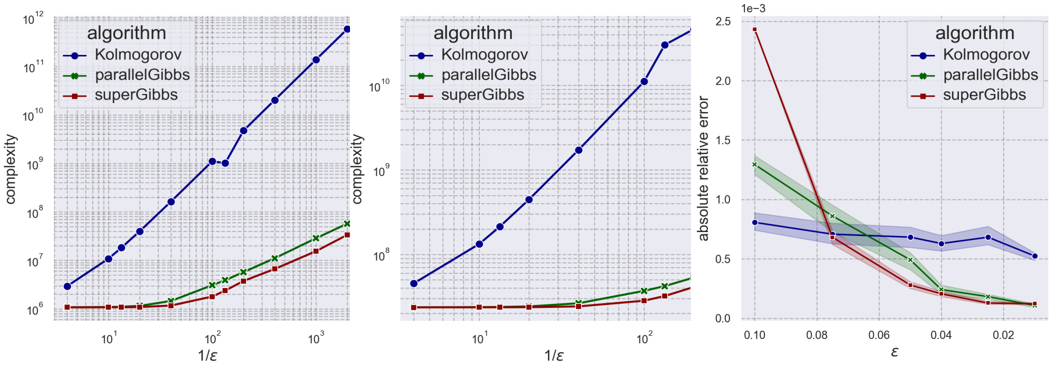
(A) The Ising model on 2D lattices. Having a 2-dimension lattice of size $n \times n$, the Hamiltonian is defined on n^2 random variables having values ± 1 and their dependency is represented by the Hamiltonian: $H(x) = -\sum_{(i,j) \in E} \mathbb{1}(x(i) = x(j))$. We run the algorithms on lattices of 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 known mixing time bounds for high temperature Ising models [3] (see fig. 1 and A.6. of supplementary material).

(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. [17] and use *hierarchy width* to derive upper bounds on mixing times. To make a fair comparison, we always run the TPA algorithms once, and with the parameters given in [42]. At each iteration of RELMEANEST, the sample size is extended with geometric ratio 1.1 (see alg. 1 line 8). All code is available at `https://github.com/zysophia/Doubly_Adaptive_MCMC`.



(a) $\beta = .01$, 3×3 lattice

(b) $\beta = .002$, 6×6 lattice

(c) relative errors

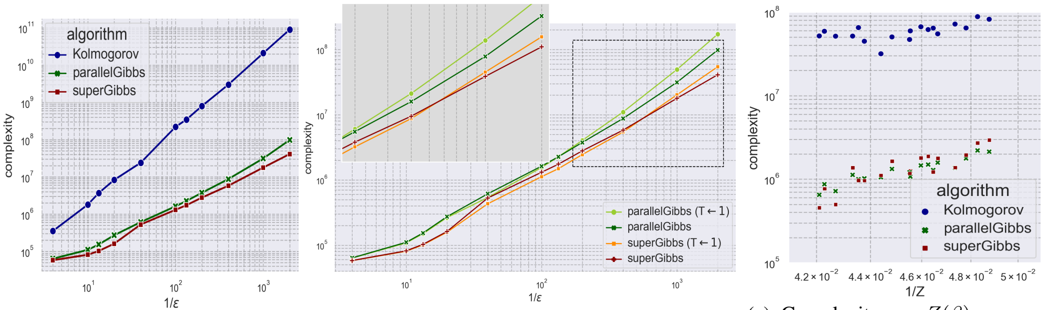
Figure 1: Comparison of sample complexity and precision $\frac{1}{\varepsilon}$ on Ising models. See also the A.6. of the supplementary material

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 on 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 [42] as $\varepsilon \to 0$ (fig. 1c).

(2) To demonstrate the advantage of using the relative *trace* variance, in contrast to the relative variance, we run both of our algorithms using a simpler mean estimator which only uses progressive sampling, and we compare the results. This is done by setting $T \leftarrow 1$ in line 4 of RELMEANEST. 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 trace variance term 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.



(a) Complexity vs. ε comparison against TPA + PPE [42].

(b) Comparison of our algorithms and the effect of trace variance.

(c) Complexity vs. $Z(\beta)$; $\varepsilon = 0.025$ comparison against TPA + PPE [42].

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

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 estimate. The complexity of the adaptive MCMC process depends on the (smaller) *trace*, rather than *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 $\Theta(\ell/\varepsilon^2)$ approximately independent samples, where ℓ is the length of cooling schedule. This requires tight convergence (total variance distance of $O(\varepsilon^2/\ell)$ from stationary) for each sample, which adds a multiplicative $\ln\frac{\ell}{\varepsilon^2}$, with $\ell = \Theta(\ln Q \ln H_{\max})$, to its complexity (see column 3 of table 1 and [42], theorem 9). In contrast, our doubly adaptive method *only* depends on relaxation times, which do not depend on ε.

Limitations. While significantly improving the state of the art, 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 $\ell \max\{\tau_i\}_{i=1}^{\ell}$, as opposed to $\sum_{i=1}^{\ell} \tau_i$. While PARALLELTRACEGIBBS circumvents these issues by estimating each factor of the telescoping product independently, it fails to beat SUPERCHAINTRACEGIBBS's efficiency in general, due both to the union bound and the higher-precision guarantees required for each subproblem. Improving performance further will likely 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.

Acknowledgements. Shahrzad Haddadan is supported by NSF Award CCF-1740741. Cyrus Cousins and Eli Upfal are supported by NSF grant RI-1813444 and DARPA/AFRL grant FA8750. The authors are thankful to anonymous reviewers of NeurIPS 2021 for several valuable inputs.

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. *SIAM Journal on Computing*, (0):FOCS20–1, 2021.

[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] Nayantara Bhatnagar, Andrej Bogdanov, and Elchanan Mossel. The computational complexity of estimating MCMC convergence time. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 424–435. Springer, 2011.

[9] 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.

[10] Stephen P Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7(4):434–455, 1998.

[11] Stephen P Brooks and Gareth O Roberts. Assessing convergence of Markov chain Monte Carlo algorithms. *Statistics and Computing*, 8(4):319–335, 1998.

[12] Zongchen Chen, Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Rapid mixing for colorings via spectral independence. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1548–1557. SIAM, 2021.

[13] 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.

[14] Kai-Min Chung, Henry Lam, Zhenming Liu, and Michael Mitzenmacher. Chernoff-Hoeffding bounds for Markov chains: Generalized and simplified. *arXiv:1201.0559*, 2012.

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

[16] Cyrus Cousins, Shahrzad Haddadan, and Eli Upfal. Making mean-estimation more efficient using an MCMC trace variance approach: DynaMITE. *CoRR*, abs/2011.11129, 2020. URL `https://arxiv.org/abs/2011.11129`.

[17] C. De 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.

[18] Chris De Sa, Vincent Chen, and Wing Wong. Minibatch Gibbs sampling on large graphical models. In *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. URL `http://proceedings.mlr.press/v80/desa18a.html`.

[19] 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, pages 1567–1576, 2016.

[20] Anand Dixit and Vivekananda Roy. MCMC diagnostics for higher dimensions using Kullback Leibler divergence. *Journal of Statistical Computation and Simulation*, 87(13):2622–2638, 2017. doi: 10.1080/00949655.2017.1335313. URL `https://doi.org/10.1080/00949655.2017.1335313`.

[21] H. Elliott, H. Derin, R. Cristi, and D. Geman. Application of the Gibbs distribution to image segmentation. 9:678–681, 1984. doi: 10.1109/ICASSP.1984.1172637.

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

[23] 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. doi: 10.1109/TPAMI.1984.4767596.

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

[25] Josiah Willard Gibbs. *Elementary Principles in Statistical Mechanics*. Scribner, 1902.

[26] 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. URL `http://proceedings.mlr.press/v15/gonzalez11a.html`.

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

[28] Shahrzad Haddadan and Peter Winkler. Mixing of permutations by biased transpositions. *Theory of Computing Systems*, 63(5):1068–1088, 2019.

[29] David G. Harris and Vladimir Kolmogorov. Parameter estimation for Gibbs distributions. *CoRR*, abs/2007.10824, 2020. URL `https://arxiv.org/abs/2007.10824`.

[30] 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.

[31] 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.

[32] Tomas Hrycej. Gibbs sampling in Bayesian networks. *Artificial Intelligence*, 46(3):351–363, 1990. ISSN 0004-3702. doi: https://doi.org/10.1016/0004-3702(90)90020-Z. URL `https://www.sciencedirect.com/science/article/pii/000437029090020Z`.

[33] Daniel Hsu, Aryeh Kontorovich, David A. Levin, Yuval Peres, Csaba Szepesvári, and Geoffrey Wolfer. Mixing time estimation in reversible Markov chains from a single sample path. *The Annals of Applied Probability*, 29(4):2439–2480, 2019. doi: 10.1214/18-AAP1457. URL `https://doi.org/10.1214/18-AAP1457`.

[34] Mark Huber. Approximation algorithms for the normalizing constant of Gibbs distributions. *The Annals of Applied Probability*, 25(2):974–985, 2015. ISSN 10505164. URL `http://www.jstor.org/stable/24519939`.

[35] Mark Huber and Sarah Schott. Random construction of interpolating sets for high-dimensional integration. *Journal of Applied Probability*, 51(1):92–105, 2014. doi: 10.1239/jap/1395771416.

[36] Mark Huber, Sarah Schott, et al. Using TPA for Bayesian inference. *Bayesian Statistics*, 9:257–282, 2010.

[37] Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

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

[39] Georgios Karagiannis and Christophe Andrieu. Annealed importance sampling reversible jump MCMC algorithms. *Journal of Computational and Graphical Statistics*, 22(3):623–648, 2013.

[40] 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.

[41] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. ISBN 0262013193.

[42] Vladimir Kolmogorov. A faster approximation algorithm for the Gibbs partition function. In *Conference On Learning Theory*, pages 228–249. PMLR, 2018.

[43] Oswin Krause, Asja Fischer, and Christian Igel. Algorithms for estimating the partition function of restricted Boltzmann machines. *Artificial Intelligence*, 278:103–195, 10 2019. doi: 10.1016/j.artint.2019.103195.

[44] 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. ISSN 0277-6715. doi: 10.1002/sim.6757. URL `https://doi.org/10.1002/sim.6757`.

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

[46] Pascal Lezaud. Chernoff-type bound for finite Markov chains. *The Annals of Applied Probability*, 8, 08 1998. doi: 10.1214/aoap/1028903453.

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

[48] D. Lunn, D. Spiegelhalter, A. Thomas, and N. Best. The BUGS project: Evolution, critique and future directions. *Stat Med.*, 28(25):3049-67, 2009 Nov 10. doi: 10.1002/sim.3680.

[49] Andrew McCallum, Karl Schultz, and Sameer Singh. FACTORIE: Probabilistic programming via imperatively defined factor graphs. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL `https://proceedings.neurips.cc/paper/2009/file/847cc55b7032108eee6dd897f3bca8a5-Paper.pdf`.

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

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

[52] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[53] David Newman, Padhraic Smyth, Max Welling, and Arthur Asuncion. Distributed inference for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008. URL `https://proceedings.neurips.cc/paper/2007/file/2dea61eed4bceec564a00115c4d21334-Paper.pdf`.

[54] 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. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2004.04.024. URL `https://www.sciencedirect.com/science/article/pii/S0378437104004327`. News and Expectations in Thermostatistics.

[55] Daniel Paulin. Concentration inequalities for Markov chains by Marton couplings and spectral methods. *Electron. J. Probab.*, 20, 2015.

[56] M. Plummer. JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. 2003.

[57] 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, pages 569–577, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581934. doi: 10.1145/1401890.1401960. URL `https://doi.org/10.1145/1401890.1401960`.

[58] Adarsh Prasad, Vishwak Srinivasan, Sivaraman Balakrishnan, and Pradeep Ravikumar. On learning Ising models under Huber's contamination model. *Advances in neural information processing systems*, 33, 2020.

[59] Maxim Rabinovich, Aaditya Ramdas, Michael Jordan, and Martin Wainwright. Function-specific mixing times and concentration away from equilibrium. *Bayesian Analysis*, 15, 05 2016. doi: 10.1214/19-BA1151.

[60] Gareth O Roberts and Jeffrey S Rosenthal. Examples of adaptive MCMC. *Journal of computational and graphical statistics*, 18(2):349–367, 2009.

[61] Jeffrey Rosenthal et al. Quantitative convergence rates of Markov chains: A simple account. *Electronic Communications in Probability*, 7:123–128, 2002.

[62] Alexander Smola and Shravan Narayanamurthy. An architecture for parallel topic models. *Proceedings of the VLDB Endowment*, 3(1–2):703–710, September 2010. ISSN 2150-8097. doi: 10.14778/1920841.1920931. URL `https://doi.org/10.14778/1920841.1920931`.

[63] 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. doi: 10.1109/FOCS.2007.67.

[64] Andreas S Stordal and Ahmed H Elsheikh. Iterative ensemble smoothers in the annealed importance sampling framework. *Advances in Water Resources*, 86:231–239, 2015.

[65] 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, pages 1124–1132, Red Hook, NY, USA, 2012. Curran Associates Inc.

[66] Christopher Tosh. Mixing rates for the alternating Gibbs sampler over restricted Boltzmann machines and friends. In *ICML*, 2016.

[67] Daniel Štefankovič, Santosh Vempala, and Eric Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3), May 2009. ISSN 0004-5411. doi: 10.1145/1516512.1516520. URL `https://doi.org/10.1145/1516512.1516520`.

[68] Ruqi Zhang and C. De Sa. Poisson-minibatching for Gibbs sampling with convergence rate guarantees. In *NeurIPS*, 2019.