Bayesian Optimization for Adaptive MCMC

by

Nimalan Mahendran

B. Math, University of Waterloo, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES
(Computer Science)

The University of British Columbia
(Vancouver)

January 2011

Abstract

A new randomized strategy for adaptive Markov chain Monte Carlo (MCMC) using Bayesian optimization, called Bayesian-optimized MCMC, is proposed. This approach can handle non-differentiable objective functions and trades off exploration and exploitation to reduce the number of function evaluations. Bayesian-optimized MCMC is applied to the complex setting of sampling from constrained, discrete and densely connected probabilistic graphical models where, for each variation of the problem, one needs to adjust the parameters of the proposal mechanism automatically to ensure efficient mixing of the Markov chains. It is found that Bayesian-optimized MCMC is able to match or surpass manual tuning of the proposal mechanism by a domain expert.

Table of Contents

List of Tables

List of Figures

Glossary

MCMC	Markov chain Monte Carlo
ARD	Automatic Relevance Determination
MH	Metropolis-Hastings
EI	Expected Improvement
SIR	Sampling-Importance Resampling
IM	Intracluster Move
RBM	Restricted Boltzmann machine

Acknowledgments

I would like to thank my readers, Dr. Nando de Freitas and Dr. Arnaud Doucet, for many insightful comments and suggestions that improved this thesis by leaps and bounds. I am grateful to Nando, my thesis supervisor, for taking me on as a graduate student - it was an amazing opportunity to work on original research in a field that interests me for its own sake, but is also so applicable to many other fields that I have always found fascinating. His patient guidance was a huge boon when working in this massive and often intimidating area.

There were also many others in the machine learning group that helped me in one way or another. Bo Chen and David Duvenaud are the first to come to mind, as the three of us pushed ourselves through all of those machine learning-related courses in our first year as graduate students. I highly value their openness and willingness to help. I am also grateful to Eric Brochu, Mark Schmidt, Matt Hoffman, Paul Vanetti, Kevin Swersky, Mohammed Emtiyaz Khan and others for valuable discussions and critique, code that I needed and the help to get it running and for taking the time to explain hard concepts to me.

The desire to do research in artificial intelligence has been with me from the start of my undergraduate career, but the path was often obscured. I am thankful to everyone that opened up that path to me, especially Dr. James Cherry, Dr. Chrysanne DiMarco, Dr. Kate Larson and Dr. Kevin Leyton-Brown.

I am thankful for the friendships I have shared with innumerable fellow graduate students, that made my stay at UBC very enjoyable, regardless of whether or not we worked directly together or shared research interests.

I would like to thank my parents and siblings for their love and support. It means very much to me that they are proud to have me as their son and brother.

Chapter 1

Introduction

Anybody can twiddle a bunch of knobs.
— The Chemical Brothers

A common line of attack for solving problems in physics, statistics and machine learning is to draw samples from probability distributions $\pi(\cdot)$ that are only known up to a normalizing constant. Markov chain Monte Carlo (MCMC) algorithms are often the preferred method for accomplishing this sampling task, see *e.g.* [5, 17]. Unfortunately, these algorithms typically have parameters that must be tuned in each new situation to obtain reasonable mixing times. These parameters are often tuned by a domain expert in a time-consuming and error-prone manual process. Adaptive MCMC methods have been developed to automatically adjust the parameters of MCMC algorithms.

Adaptive MCMC methods based on stochastic approximation have garnered the most interest out of the various types of adaptive MCMC methods for two reasons. Firstly, they can be shown to be theoretically valid, in the sense that although the Markov chain is made inhomogenous by the dependence of the parameter updates upon the history of the Markov chain, its ergodicity can be ensured [2, 3, 20]. Secondly, they have produced very impressive results in the case of the random walk Metropolis algorithm [10, 24]. However, there are limitations to the stochastic approximation approach. Some of the most successful samplers rely on knowing either the optimal acceptance rate or the gradient of some objective function of interest. Another disadvantage is that these stochastic approximation methods may require many iterations in some domains.

This work aims to overcome some of these limitations. It proposes the use of Bayesian optimization [7] to tune the parameters of the Markov chain. The proposed approach, Bayesian-optimized MCMC, has a few advantages over adaptive methods based on stochastic approximation.

Bayesian optimization does not require that the objective function be differentiable, enabling one to be much more flexible in the design of the adaptation mechanisms. The area under the auto-correlation function up to a specific lag is used as the objective function in this work. This objective function has been suggested previously in [3]. However, the computation of gradient estimates for this objective function is very involved and far from trivial [3]. This is believed to be one of the main reasons that practitioners have not embraced this approach. Bayesian optimization is shown here to easily optimize this objective function and to endow the designer with greater freedom in the design of adaptive strategies.

Bayesian optimization also has the advantage that it is explicitly designed to trade off exploration and exploitation and is implicitly designed to minimize the number of evaluations of the objective function [7], which may be very expensive.

Another important property of Bayesian-optimized MCMC is that it uses a distribution over the parameter settings of the proposal distribution, with probabilities estimated during the adaptation process, rather than a specific parameter setting. It was found that these randomized policies mix faster than specific parameter settings, for the models considered in this work.

Bayesian optimization has been used with MCMC in [15] with the intent to approximate the posterior with a surrogate function to minimize the cost of hybrid Monte Carlo evaluations. The intent in this work is instead to adapt the parameters of the Markov chain to improve mixing.

It is conjectured here that Bayesian optimization for MCMC has convergence properties similar to those of stochastic approximation for MCMC, given existing consistency results for Bayesian optimization [22, 23]. However, the type of Bayesian optimization studied in this work relies on latent Gaussian processes defined over the chain of samples. It becomes prohibitively expensive to invert the covariance matrix induced by a Gaussian process for very large chains. Thus, for practical reasons, a two-stage adaptation mechanism was adopted instead.

Chapter 2

Adaptive MCMC

rain and snow differ
like statistical models -
adaptation helps

The Metropolis-Hastings (MH) algorithm is the key building block for most MCMC methods [5]. It draws samples from a target distribution $\pi(\cdot)$ by proposing a move from $\mathbf{x}^{(t)}$ to $\mathbf{y}^{(t+1)}$ according to a parameterized proposal distribution $q_\theta(\mathbf{y}^{(t+1)}|\mathbf{x}^{(t)})$ and either accepting it ($\mathbf{x}^{(t+1)} = \mathbf{y}^{(t+1)}$) with probability equal to the acceptance ratio

$$\alpha(\mathbf{x}^{(t)} \rightarrow \mathbf{y}^{(t+1)}) = \min\left\{\frac{\pi(\mathbf{y}^{(t+1)})q_\theta(\mathbf{x}^{(t)}|\mathbf{y}^{(t+1)})}{\pi(\mathbf{x}^{(t)})q_\theta(\mathbf{y}^{(t+1)}|\mathbf{x}^{(t)})}, 1\right\}$$

or rejecting it ($\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$) otherwise.

The parameters of the proposal, $\theta \in \mathbf{\Theta} \subseteq \mathbb{R}^d$, can have a large influence on sampling performance. For example, the experiments in Section 4.2 consider constrained discrete probabilistic models, where changes to the connectivity patterns among the random variables will require different parameter settings. An approach that can adjust these parameters automatically for all possible connectivity patterns is very desirable.

Several methods have been proposed to adapt MCMC algorithms. In the interest of brevity, the reader is referred to the comprehensive reviews of [4, 6, 18]. One can adapt parameters other than those of the proposal distribution in certain situations, but for the sake of simplicity, we focus here on adapting the proposal distribution.

One of the most successful adaptive MCMC algorithms was introduced in [10] and several extensions were presented in [4]. This algorithm is restricted to the adaptation of the multivariate random walk Metropolis algorithm with Gaussian proposals. It is motivated by a theoretical result regarding the optimal covariance matrix of a restrictive version of this sampler [9]. This adaptive algorithm belongs to the family of stochastic approximation methods.

Some notation needs to be introduced to briefly describe stochastic approximation, but will be useful later, when the stochastic approximation method is replaced with Bayesian optimization. Let $\mathcal{X}_i = \{\mathbf{x}^{(t)}\}_{t=1}^{i}$ denote the full set of samples up to iteration i of the MH algorithm and $\mathcal{Y}_i = \{\mathbf{y}^{(t)}\}_{t=1}^{i}$ be the corresponding set of proposed samples. $\mathbf{x}^{(0)}$ is the initial sample. Let $g(\theta)$ be the mean field of the stochastic approximation that may only be observed noisily as $G(\theta_i, \mathbf{x}^{(0)}, \mathcal{X}_i, \mathcal{Y}_i)$. This mean field corresponds to the gradient of the objective function $h(\theta)$ being optimized, that is $g(\theta) = \nabla h(\theta)$. Adaptive MCMC methods based on stochastic approximation typically use the following Robbins-Monro update:

$$\theta_{i+1} = \theta_i + \gamma_{i+1} G\left(\theta_i, \mathbf{x}^{(0)}, \mathcal{X}_{i+1}, \mathcal{Y}_{i+1}\right), \tag{2.1}$$

where γ_{i+1} is the step-size. This recursive estimate converges almost surely to the roots of $g(\theta)$ as $i \to \infty$ under suitable conditions.

This work is concerned with the adaptation of discrete models, where the optimal acceptance rates are unknown and it is not clear what objective function should be optimized to adjust the parameters of the proposal distribution. One possible choice, stated in the introduction, is to use the area under the auto-correlation function up to a certain lag. This objective function intuitively seems to be suitable for the adaptation task because it is used in practice to assess convergence, but its gradient cannot be effciently estimated [3]. We introduce Bayesian optimization in the following section to overcome this difficulty.

Chapter 3

Bayesian Optimization for Adaptive MCMC

noise, no gradients
and only a Markov chain -
ominous clouds form

The proposed adaptive strategy consists of an adaptation phase and a sampling phase. Bayesian optimization is used to construct a randomized policy in the adaptation phase by performing the gradient-free optimization of a practitioner-supplied objective function. A mixture of MCMC kernels, selected according to the learned randomized policy, is used to explore the target distribution in the sampling phase. These two phases and the specific objective function used in the experiments in section 4.2 are discussed in more detail subsequently.

3.1 Auto-correlation-based objective function

The objective function used in the experiments in section 4.2 is based on the auto-correlation $r(l, \theta)$ of the Markov chain $\mathcal{X}^\theta = \{\mathbf{x}_\theta^{(1)}, \mathbf{x}_\theta^{(2)}, \ldots\}$ of samples generated with parameters θ, defined as

$$r(l, \theta) \triangleq \frac{1}{\delta_\theta^2}\mathbb{E}\left[(\mathbf{x}_\theta^{(t)} - \bar{\mathbf{x}}_\theta)^T(\mathbf{x}_\theta^{(t+l)} - \bar{\mathbf{x}}_\theta)\right],$$

where the expectation is taken with respect to the stationary distribution of the Markov chain $\mathcal{X}^\theta$, l is the auto-correlation time lag and $\bar{\mathbf{x}}_\theta$ and δ_θ^2 are the mean and the variance of $\mathcal{X}^\theta$, respectively. Faster mixing times are characterized by larger values of the objective function $h(\theta) = 1 - (l_{\text{max}}{}^{-1}) \sum_{l=1}^{l_{\text{max}}} |r(l,\theta)|$, where l_{max} is the largest lag to be considered.

3.1.1 Empirical estimation of $h(\theta)$

$r(l,\theta)$ cannot be evaluated analytically and must be approximated by the estimator $\widehat{r}(l,\mathcal{X}^\theta)$, defined as

$$\widehat{r}(l,\mathcal{X}^\theta) \triangleq \frac{1}{(L-l)\delta_\theta^2} \sum_{t=1}^{L-l} (\mathbf{x}_\theta^{(t)} - \bar{\mathbf{x}}_\theta)^T (\mathbf{x}_\theta^{(t+l)} - \bar{\mathbf{x}}_\theta),$$

where $\mathcal{X}^\theta = \{\mathbf{x}_\theta^{(1)}, \ldots, \mathbf{x}_\theta^{(L)}\}$ is a sequence of L samples generated with parameters θ and $\bar{\mathbf{x}}_\theta$ and δ_θ^2 are now the sample mean and variance of $\mathcal{X}^\theta$, respectively. $h(\theta)$ must in turn be estimated by $\widehat{a}(\mathcal{X}^\theta) = 1 - (l_{\text{max}}{}^{-1}) \sum_{l=1}^{l_{\text{max}}} |\widehat{r}(l,\mathcal{X}^\theta)|$, where l_{max} here is $L-1$.

The auto-correlation estimator $\widehat{r}(l,\mathcal{X}^\theta)$ and hence $\widehat{a}(\mathcal{X}^\theta)$ may be inaccurate because $\mathcal{X}^\theta$ is likely to contain outliers or changepoints, since L is often too small for the MCMC chain to reach its stationary regime. A more robust objective function estimator can be obtained by averaging $\widehat{a}(\cdot)$ over successively larger intervals of the last samples in $\mathcal{X}^\theta$. Therefore, the objective function estimator actually used is $\widehat{h}(\mathcal{X}^\theta) \triangleq \frac{1}{L-l_{\text{min}}+1} \sum_{i=l_{\text{min}}}^{L} \widehat{a}(\mathcal{E}_i)$, where $\mathcal{E}_i$ is the last i states in $\mathcal{X}^\theta$ (the states' energies in Section 4) and l_{min} is the smallest interval length to consider. Note that $\mathcal{E}_L$ corresponds to $\mathcal{X}^\theta$, the entire sequence associated with θ.

3.2 Adaptation phase

The noisy observation $z_i \triangleq \widehat{h}(\mathcal{X}^{\theta_i})$ can be obtained by running the Markov chain for L steps with the parameters θ_i. Bayesian optimization in the adaptive MCMC setting then proposes a new candidate θ_{i+1} by constructing a model of the objective function using the entire history of noisy observations and a prior distribution over functions. Gaussian processes are used here as the prior distribution.

The predictive distribution of the Gaussian process is obtained using these noisy observations. An expected utility function, also known as the acquisition function, is derived in terms of the sufficient statistics of this predictive distribution. The acquistion function is then optimized to select the next parameter value θ_{i+1}. The overall procedure is shown in Algorithm 1. The reader is referred to [7, 14] for in-depth reviews of Bayesian optimization.

Algorithm 1 Adaptive MCMC with Bayesian Optimization

```
1: for $i = 1, 2, \ldots, I$ do
2:     Run Markov chain for $L$ steps with parameters $\theta_i$.
3:     Use the drawn samples to obtain a noisy evaluation of the objective function:
       $z_i = h(\theta_i) + \epsilon$.
4:     Augment the data $\mathcal{D}_{1:i} = \{\mathcal{D}_{1:i-1}, (\theta_i, z_i)\}$.
5:     Update the GP's sufficient statistics.
6:     Find $\theta_{i+1}$ by optimizing the acquisition function:
       $\theta_{i+1} = \arg\max_\theta u(\theta|\mathcal{D}_{1:i})$.
7: end for
```

The objective function $h(\cdot)$ is assumed to be distributed according to a Gaussian process with mean function $m(\cdot)$ and covariance function $k(\cdot, \cdot)$:

$$h(\cdot) \sim GP(m(\cdot), k(\cdot, \cdot)).$$

A zero mean function $m(\cdot) = 0$ and an anisotropic Gaussian covariance $k(\theta_j, \theta_k)$ that is essentially the popular Automatic Relevance Determination (ARD) kernel [16] are adopted:

$$k(\theta_j, \theta_k) = \exp\left(-\frac{1}{2}(\theta_j - \theta_k)^T \mathrm{diag}(\psi)^{-2}(\theta_j - \theta_k)\right),$$

where $\psi \in \mathbb{R}^d$ is a vector of hyper-parameters corresponding to the length-scales of each dimension of θ. The Gaussian process is a surrogate model for the true objective function, which typically involves intractable expectations with respect to the invariant distribution and the MCMC transition kernels. Noisy Gaussian measurements are assumed, since this objective function can only be sampled:

$$z_i = h(\theta_i) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma_\eta^2),$$

see for example [8].

Let $\mathbf{z}_{1:i} \sim \mathcal{N}(0, \mathbf{K})$ be the i noisy observations of the objective function obtained from previous iterations. (Note that the Markov chain is run for L steps for each discrete iteration i. The extra index to indicate this fact has been made implicit to improve readability.) $\mathbf{z}_{1:i}$ and h_{i+1} are jointly multivariate Gaussian:

$$\begin{bmatrix} \mathbf{z}_{1:i} \\ h_{i+1} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\eta^2 I & \mathbf{k}^T \\ \mathbf{k} & k(\theta, \theta) \end{bmatrix}\right),$$

where

$$\mathbf{K} = \begin{bmatrix} k(\theta_1, \theta_1) & \dots & k(\theta_1, \theta_i) \\ \vdots & \ddots & \vdots \\ k(\theta_i, \theta_1) & \dots & k(\theta_i, \theta_i) \end{bmatrix}$$

and

$$\mathbf{k} = [k(\theta, \theta_1) \ \dots \ k(\theta, \theta_i)]^T.$$

These assumptions about the form of the prior distribution and observation model are standard and less restrictive than they might first appear. The main assumption is that the objective function is smooth.

The predictive distribution for any value θ follows from the Sherman-Morrison-Woodbury formula, where $\mathcal{D}_{1:i} = (\theta_{1:i}, \mathbf{z}_{1:i})$:

$$\begin{aligned} p(h_{i+1}|\mathcal{D}_{1:i}, \theta) &= \mathcal{N}(\mu_i(\theta), \sigma_i^2(\theta)) \\ \mu_i(\theta) &= \mathbf{k}^T(\mathbf{K} + \sigma_\eta^{\mathbf{2}}\mathbf{I})^{-1}\mathbf{z}_{1:i} \\ \sigma_i^2(\theta) &= k(\theta, \theta) - \mathbf{k}^T(\mathbf{K} + \sigma_\eta^{\mathbf{2}}\mathbf{I})^{-1}\mathbf{k} \end{aligned}$$

The next query point θ_{i+1} is chosen to maximize an acquisition function, $u(\theta|\mathcal{D}_{1:i})$, that trades-off exploration (where $\sigma_i^2(\theta)$ is large) and exploitation (where $\mu_i(\theta)$ is high). Expected Improvement (EI) over the best candidate was adopted as this acquisition function and it was optimized using the DIRECT algorithm, following [7, 21]. This optimization step is fairly fast and efficient. The reader is referred to [7] for details.

3.3 Sampling phase

The adaptation phase results in a Gaussian process on the I noisy observations of the performance criterion $\mathbf{z}_{1:I}$, taken at the corresponding locations in parameter space $\theta_{1:I}$. A discrete stochastic policy $p(\theta|\mathbf{z}_{1:I})$, defined over the parameter space $\boldsymbol{\Theta}$ and proportional to this Gaussian process, is constructed using the simple Sampling-Importance Resampling (SIR) approach [13, 19] given in Algorithm 2. The mean function is exponentiated and used as the unnormalized target distribution, since the Gaussian process can take on negative values. The generation of the candidate points might require optimization in large-dimensional spaces, but Latin hypercubes [25] suffice for the MCMC algorithms dealt with in this work. There are

Algorithm 2 SIR-based policy construction

```
1: Generate a set of candidate points $\widetilde{\theta}_{1:N} \triangleq \{\widetilde{\theta}_1, \dots, \widetilde{\theta}_N\}$, using either Latin
   hypercubes or optimization methods.
2: Obtain the weights $\widetilde{w}_i = \exp(\mu(\widetilde{\theta}_i))$ for $i = 1 : N$ by evaluating the Gaussian
   process mean function at each point in $\widetilde{\theta}_{1:N}$ and exponentiating it.
3: Normalize the weights: $w_i = \frac{\widetilde{w}_i}{\sum_{j=1}^{N} \widetilde{w}_j}$.
4: Resample, with replacement, $M$ samples $\{\theta_i | i = 1, \dots, M\}$ from the
   weighted discrete measure $\{(\widetilde{\theta}_i, w_i) | i = 1, \dots, N\}$.
```

several ways to proceed in the sampling phase once this normalized discrete measure proportional to the Gaussian process has been obtained. Here, it was choosen to run the Markov chain with parameter settings drawn at each step from this discrete stochastic policy, or equivalently, a mixture of M MCMC transition kernels is adopted, where each kernel uses one of the M parameters obtained in the SIR step in Algorithm 2. The distribution of the samples generated in the sampling phase will approach the target distribution $\pi(\cdot)$ as the number of iterations tends to ∞, provided that the kernels in this finite mixture are ergodic.

One should not erroneously draw the conclusion that running an unadapted chain for more iterations would give similar performance to running an adapted chain. The unadapted chain can mix extremely slowly for poor choices of the parameters, making the extra computation involved in the adaptation and sampling phases of an adapted chain worthwhile.

Chapter 4

Application to Constrained Discrete State-space Distributions

squishy human or
metallic automaton?
cold machine conquers

The Intracluster Move (IM) sampler, an MH algorithm, was recently proposed to generate samples from the notoriously-hard constrained Boltzmann machines in [11]. This sampler has two parameters, one continuous and the other discrete, that the authors state to be difficult to tune in some settings. The proposed Bayesian-optimized MCMC method was applied to this problem.

4.1 The IM sampler

Boltzmann machines are described in [1]. Let $x_i \in \{0, 1\}$ denote the i-th random variable in $\mathbf{x} \in \mathcal{S}$, where $\mathcal{S}$ is the state space. The Boltzmann distribution is

$$\pi(\mathbf{x}) \triangleq \frac{1}{Z(\beta)} e^{-\beta E(\mathbf{x})}, \tag{4.1}$$

where

$$Z(\beta) \triangleq \sum_{\mathbf{x}\in\mathcal{S}} e^{-\beta E(\mathbf{x})}$$

is the normalizing constant, β is a temperature parameter and

$$E(\mathbf{x}) \triangleq -\sum_{i,j} x_i J_{ij} x_j - \sum_i b_i x_i$$

is the energy function, where J and b are coupling parameters that are assumed to be known.

Let $\mathcal{S}_n(\mathbf{c})$ be the subset of the states that are at exactly Hamming distance n away from a reference state $\mathbf{c}$. The distribution $\pi_{n,\mathbf{c}}(\mathbf{x})$ is the restriction of $\pi(\mathbf{x})$ to $\mathcal{S}_n(\mathbf{c})$. $\pi_{n,\mathbf{c}}(\mathbf{x})$ has

$$Z_n(\beta,\mathbf{c}) \triangleq \sum_{\mathbf{x}\in\mathcal{S}_n(\mathbf{c})} e^{-\beta E(\mathbf{x})}$$

as its normalizing constant and is defined as

$$\pi_{n,\mathbf{c}}(\mathbf{x}) \triangleq \begin{cases} \frac{1}{Z_n(\beta,\mathbf{c})} e^{-\beta E(\mathbf{x})} & \text{if } \mathbf{x} \in \mathcal{S}_n(\mathbf{c}) \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

The rest of this work makes $\mathbf{c}$ implicit and uses the simplified notation $\mathcal{S}_n$, $\pi_n(\mathbf{x})$ and $Z_n(\beta)$. These constraints on the states arise in statistical physics and in regularized statistical models [11].

The IM sampler proposes a new state $\mathbf{y}^{(t+1)} \in \mathcal{S}_n$ from an original state $\mathbf{x}^{(t)} \in \mathcal{S}_n$ using self-avoiding walks (SAWs) and has parameters $\theta = (k, \gamma)$, where $k \in \mathcal{L} \triangleq \{1, 2, \ldots, k_{\max}\}$ is the length of each SAW and $\gamma \in \mathcal{G} \triangleq [0, \gamma_{\max}]$ is the energy-biasing parameter. k determines the size, in terms of the number of bits flipped, of the moves through $\mathcal{S}_n$. γ controls the degree to which higher energy states are favored.

4.2 Experimental setup

The experiments compare the performance of four different sampling methods on three different models. Five trials were conducted for each combination of sampling method and model.

4.2.1 Sampling methods

The sampling methods are all instances of the IM sampler that differ in the manner that γ and k are picked:

Kawasaki sampler transitions from state to state within $\mathcal{S}_n$ by uniformly sampling a bit to flip to produce a state in $\mathcal{S}_{n+1}$ or $\mathcal{S}_{n-1}$ and then uniformly sampling a bit to flip to return to $\mathcal{S}_n$ [12]. This is equivalent to running the IM sampler with γ fixed to 0 and k fixed to 1.

IMExpert is the IM sampler manually tuned by a domain expert [11], with γ fixed to γ_{expert} and k drawn uniformly from $\mathcal{L}$.

IMUnif is a completely naive approach that draws γ uniformly from $\mathcal{G}$ and k uniformly from $\mathcal{L}$.

IMBayesOpt is Bayesian-optimized MCMC applied to the IM sampler with L samples generated for each of the I adaptations of the parameters. Adaptation is restricted to parameters in $\mathcal{L} \times \mathcal{G}$.

The parameter sets $\mathcal{L}$ and $\mathcal{G}$ that were chosen for each model are shown in Table 4.1, where "Others" refers to IMUnif and IMBayesOpt. These two samplers have the same parameter sets because IMUnif is a baseline algorithm used to ensure that Bayesian-optimized IM performs better than a naive strategy. The parameter sets for IMUnif and IMBayesOpt were selected such that $\mathcal{L}$ is a much larger superset of the SAW lengths used for IMExpert and $\mathcal{G}$ is the contiguous interval from 0 to $2\gamma_{\text{expert}}$. The parameter sets for IMExpert come from [11]. The Kawasaki sampler does not have any parameters.

Table 4.1: Algorithm parameters for each model. IMExpert parameters are from [11]

Model	Algorithm	$\mathcal{L}$	$\mathcal{G}$
2DGrid	IMExpert	$\{90\}$	$\{0.44\}$
2DGrid	Others	$\{1,...,300\}$	$[0, 0.88]$
3DCube	IMExpert	$\{1,...,25\}$	$\{0.8\}$
3DCube	Others	$\{1,...,50\}$	$[0, 1.6]$
RBM	IMExpert	$\{1,...,20\}$	$\{0.8\}$
RBM	Others	$\{1,...,50\}$	$[0, 1.6]$

Bayesian-optimized MCMC parameters

IMBayesOpt has a number of additional parameters, and thus it may seem that the parameters of the sampler have simply been substituted with those of Bayesian-optimized MCMC. However, although the values of these parameters were picked by hand, they were either set to be the same across all models or differed across models, but were set using the same simple underlying rule. The sampler parameters would have had to be tuned for every individual model, whereas the Bayesian-optimized MCMC parameters were simply fixed to sensible default values and then used across all models. The Bayesian-optimized MCMC parameter settings are conjectured to work across a wide variety of statistical models, but verifying this claim is outside the scope of this work.

The following parameters of IMBayesOpt were the same for each model: the number of Baysian optimization steps, I, was set to 100, the number of samples used to estimate the objective function, L, was set to 100, the variance of the noisy Gaussian measurements, σ_η^2, was set to 0.1 and the minimum lag considered, l_{min}, was set to 25. l_{max} was fixed to $L-1$, by definition.

The remaining parameters were set differently for each model, but using the same underlying rule. The ARD kernel hyper-parameters ψ were set to 0.1ϕ, where ϕ is the length of the intervals $[\min \mathcal{L}, \max \mathcal{L}]$ and $[\min \mathcal{G}, \max \mathcal{G}]$. The candidate points $\widetilde{\theta}_{1:N}$ in algorithm 2 were generated by taking the cross-product of 100 evenly-spaced points within $\mathcal{G}$ with the elements of $\mathcal{L}$, so that the candidate points are arranged in a grid. This resulted in $N = 30000$, $N = 5000$ and $N = 5000$

Table 4.2: Model parameters from [11]. n refers to the Hamming distance from states in $\mathcal{S}_n$ to the reference state $\mathbf{c}$ with the column indicating the number of bits that are set to 1 out of the total number of bits. β^{-1} is the temperature of the model.

Model	β^{-1}	Size	n
2DGrid	2.27	60×60	1800 of 3600
3DCube	1.0	$9 \times 9 \times 9$	364 of 729
RBM	1.0	$v = 784$, $h = 500$	428 of 1284

candidate points for the 2DGrid, 3DCube and restricted Boltzmann machine (RBM) models, respectively. M, the sizes of the generated randomized policies for each model, were fixed to N. The average number of unique parameter settings in the randomized policies constructed by IMBayesOpt for the 2DGrid, 3DCube and RBM models, averaged over five trials, were 8486, 1682 and 2200 respectively.

4.2.2 Models

The three models studied in [11] are considered. The model parameters are given in Table 4.2. Note that n refers to the Hamming distance from states in $\mathcal{S}_n$ to the reference state $\mathbf{c}$ and that β^{-1} is the temperature of the model. The reference state $\mathbf{c}$ was the ground state, where none of the bits are 1.

Ferromagnetic 2D grid Ising model

The ferromagnetic 2D grid Ising model consists of nodes arranged in a planar rectangular grid with edges between the nodes on one boundary to the nodes on the other boundary for each dimension (i.e. periodic boundaries), also known as a toroidal grid. Hence, each node has exactly four neighbours. The interaction weights, J_{ij}, are all 1 and the biases, b_i, are all 0.

Frustrated 3D cube Ising model

The frustrated 3D cube Ising model consists of nodes arranged in a topology that is the three-dimensional analogue of the two-dimensional grid, with periodic boundaries. Hence, each node has exactly six neighbours. The interaction weights, J_{ij}, are uniformly sampled from $\{-1, 1\}$ and the biases, b_i, are all 0.

Restricted Boltzmann machine (RBM)

The RBM has a bipartite graph structure, with h hidden nodes in one partition and v visible nodes in the other. The interaction weights J_{ij} and biases b_i are exactly the same as in [11] and correspond to local Gabor-like filters that capture regularities in perceptual inputs.

4.2.3 Details of sampler run lengths

Each sampler was run five times with 9×10^4 steps for each run. IMBayesOpt had an additional 10^4 steps for an adaptation phase consisting of 100 adaptations of 100 samples each. IMBayesOpt was not penalized for the computational overhead involved in these additional steps because it is seen as being far cheaper than having the IM sampler parameters tuned manually. All of the algorithms have a burn-in phase consisting of the first 10^4 samples generated in the corresponding sampling phase. The burn-in phase was not included in the computation of the auto-correlation functions in Figures 4.1, 4.4 and 4.7. IMBayesOpt begins its sampling phase in the same starting state as all of the other samplers, even though it would most likely be in a low energy state at the end of its adaptation phase, to ensure fairness.

4.3 Results and discussion

4.3.1 Ferromagnetic 2D grid Ising model

IMExpert and IMBayesOpt have very similar mean auto-corrrelation functions, as indicated in Figure 4.1. Figure 4.2 shows that IMUnif suffers from long strings of consecutive proposal rejections. This is evident from the many intervals where the sampled state energy does not change.

Figure 4.3 suggests that γ is much more important to the performance of the IM sampler than the SAW lengths for this model, especially at large SAW lengths. One of the highest peaks in the Gaussian process mean function corresponds to the parameters chosen by [11] ($\gamma = 0.44, k = 90$).

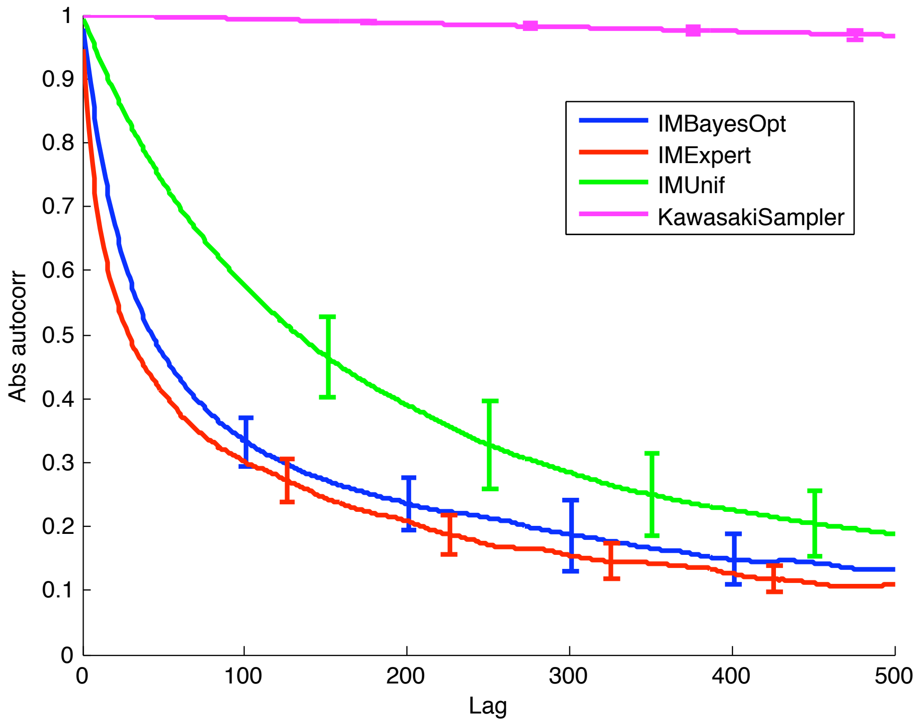

Figure 4.1: Mean and standard deviation of the auto-correlation function of the energies of the sampled states drawn from the 2D grid Ising model by each of the four sampling methods, taken over five trials.

4.3.2 Frustrated 3D cube Ising model

Figures 4.4 and 4.5 show that IMUnif now performs far worse than the IMExpert and IMBayesOpt, implying that the extremely rugged energy landscape of the 3D cube Ising model makes manual tuning a non-trivial and necessary process. IMBayesOpt performs similarly to IMExpert, but is automatically tuned.

The Gaussian process mean function in Figure 4.6 suggests that SAW lengths should not be longer than $k = 25$, as found in [11]. Both IMExpert and IMBayesOpt are essentially following the same strategy and performing well, while the performance of IMUnif confirms that tuning is important for the 3D cube Ising model.

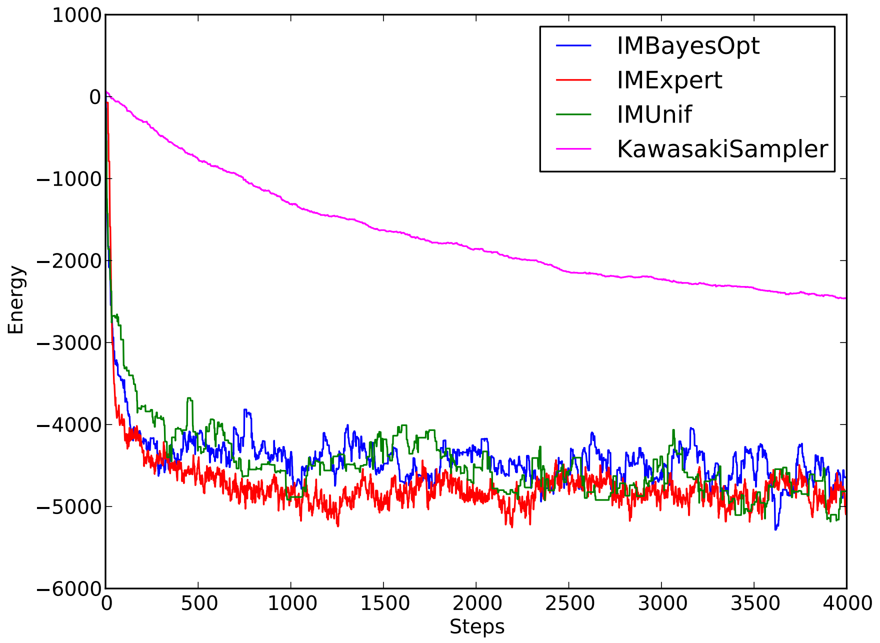

Figure 4.2: Sampled states' energies from the start of the sampling phase for the 2D grid Ising model, from the first trial.

4.3.3 Restricted Boltzmann machine (RBM)

The rapid dropoff experienced by IMExpert in [11] is exaggerated by the inclusion of the burn-in phase. Figure 4.7 shows a much more modest dropoff when the burn-in phase is left out of the auto-correlation function computation. However, it still corroborates the claim in [11] that IMExpert performs far better than the Kawasaki sampler.

Figures 4.7 and 4.8 both show that IMExpert does not perform much better than IMUnif. The variance of IMExpert's auto-correlation function is also much higher than any of the other methods. IMBayesOpt performs significantly better than any of the other methods, including manual tuning by a domain expert.

The Gaussian process mean function in Figure 4.9 suggests that SAW lengths greater than 20 can be used and are at least as effective as shorter ones, whereas [11] only picks SAW lengths between 1 and 20. This discrepancy is an instance where

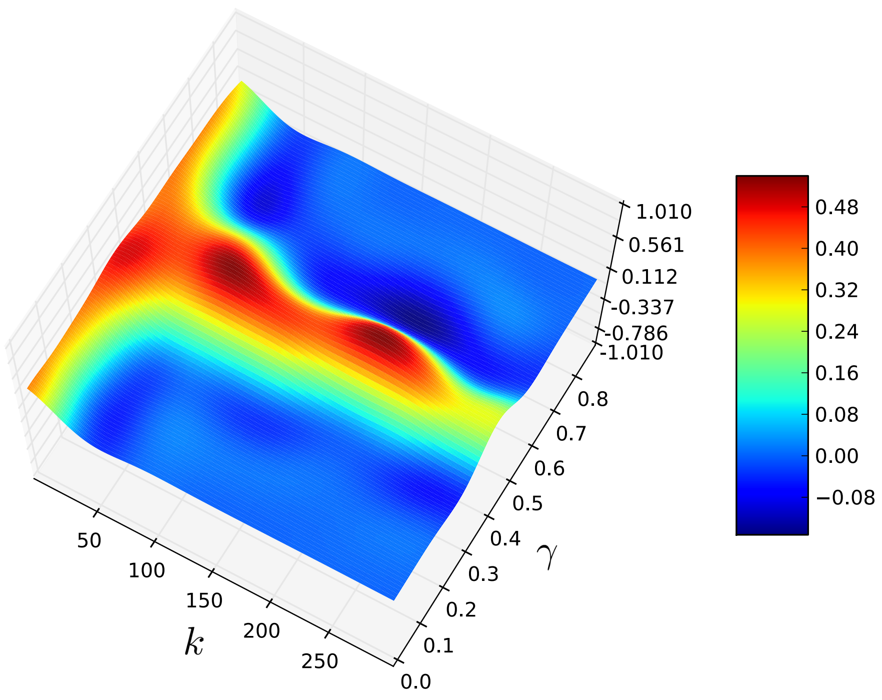

Figure 4.3: The mean function of the Gaussian processes over $\boldsymbol{\Theta}$, learned by IMBayesOpt for the 2D grid Ising model. An average over the five trials is shown.

Bayesian-optimized MCMC has found a better strategy for selecting parameters than a domain expert.

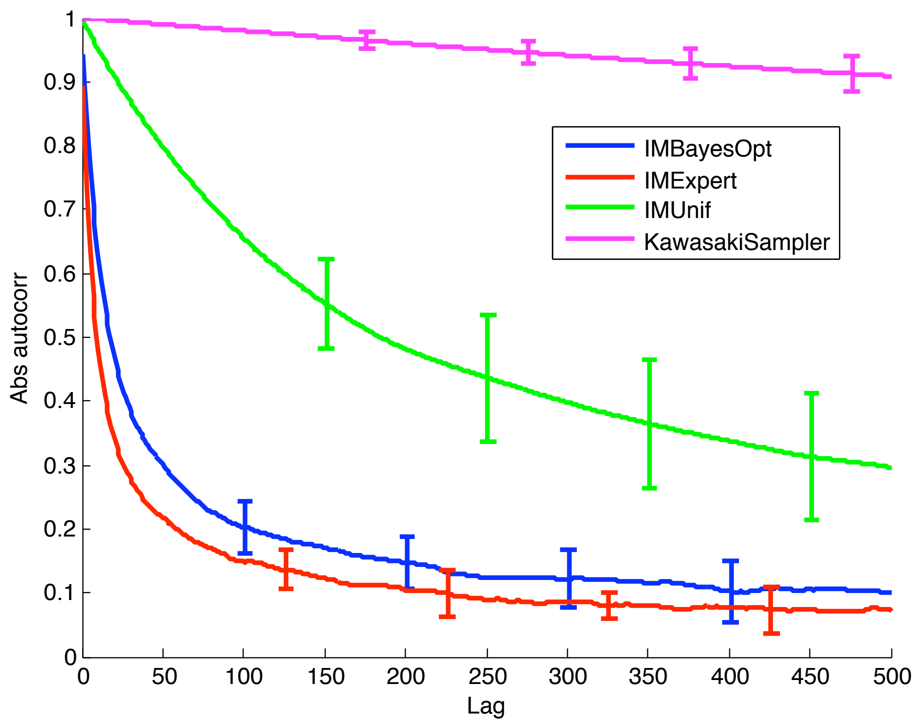

Figure 4.4: Mean and standard deviation of the auto-correlation function of the energies of the sampled states drawn from the 3D cube Ising model by each of the four sampling methods, taken over five trials.

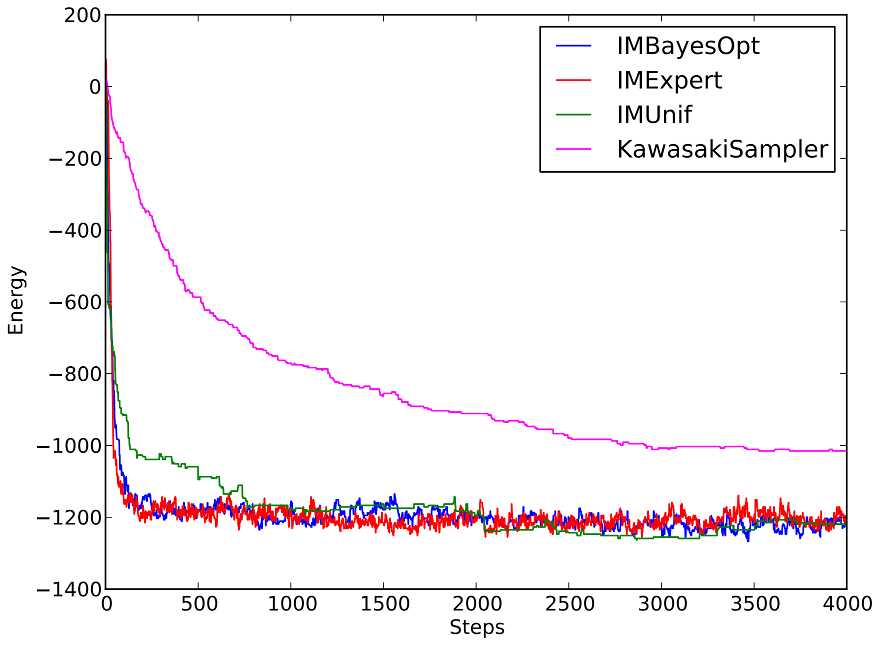

Figure 4.5: Sampled states' energies from the start of the sampling phase for the 3D cube Ising model, from the first trial.

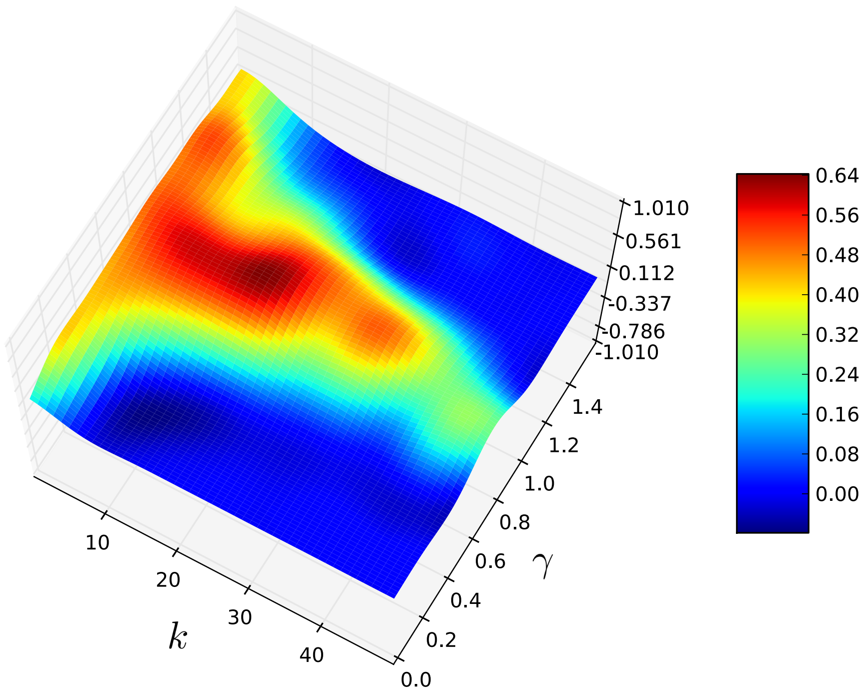

Figure 4.6: The mean function of the Gaussian processes over $\mathbf{\Theta}$, learned by IMBayesOpt for the 3D cube Ising model. An average over the five trials is shown.

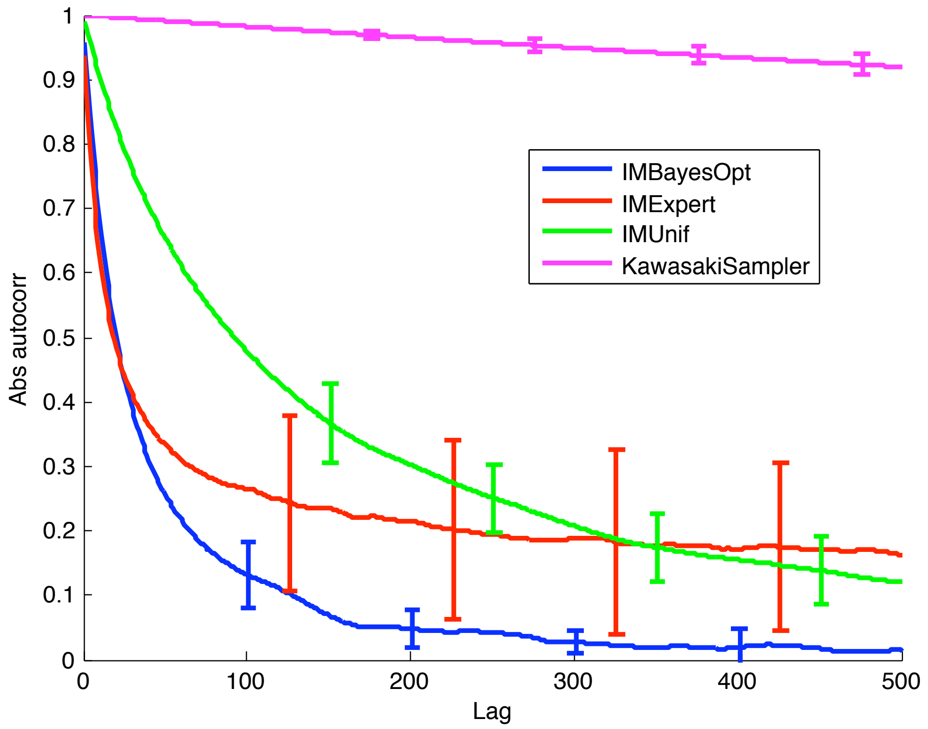

Figure 4.7: Mean and standard deviation of the auto-correlation function of the energies of the sampled states drawn from the RBM by each of the four sampling methods, taken over five trials.

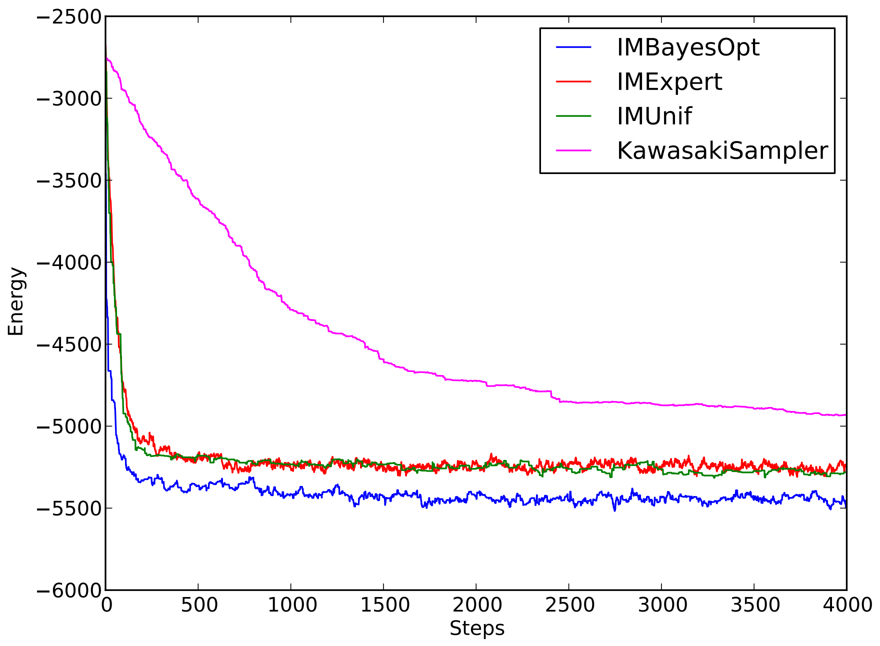

Figure 4.8: Sampled states' energies from the start of the sampling phase for the RBM, from the first trial.

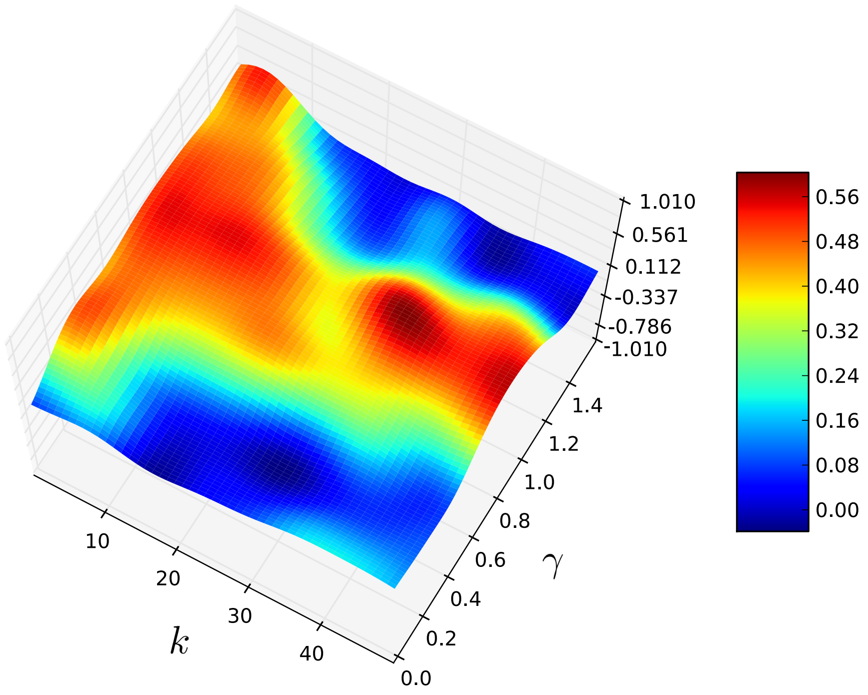

Figure 4.9: The mean function of the Gaussian processes over $\mathbf{\Theta}$, learned by IMBayesOpt for the RBM. An average over the five trials is shown.

Chapter 5

Conclusion

Experiments were conducted to assess Bayesian optimization for adaptive MCMC. These experiments show that the manual tuning done in [11] significantly improves the performance of the IM sampler for the 2D grid Ising model and the 3D cube Ising model, but Bayesian-optimized MCMC is able to realize the same gains without any human intervention and surpasses the human expert for the RBM model.

The Gaussian measurement noise model assumes that the noise is unbiased and has a constant variance σ_η^2 throughout the parameter space. The noisy measurements of $h(\theta)$ are dependent on the initial samples drawn with θ; elements of the parameter space with very slow mixing rates may have biased or high-variance noise, and hence break this assumption. Bayesian-optimized MCMC may not end up learning an appropriate stochastic policy, as the underlying Gaussian process may be a very poor model of the true objective function. This necessitates an additional assumption that the mixing rate is uniform throughout the parameter space, but this is an assumption that is also needed in stochastic approximation, which will not converge if the gradient estimates are biased or have high variance. The variance of the measurement error model could be estimated, at additional expense, by performing multiple runs with the same parameters, but this has been left for future work.

Parametric bandit algorithms could be used in Bayesian optimization, instead of Gaussian processes, to make it practical to adapt infinitely often, but the interesting challenge of showing convergence for a single adapted chain would arise.

Bayesian optimization is a general method for adapting the parameters of any MCMC algorithm. It has some advantages over stochastic approximation, as indicated and demonstrated in this paper. However, it presently only applies to parameter spaces of up to fifty dimensions. It should not be seen as a replacement for stochastic approximation, but rather as a complementary technique. In particular, Bayesian optimization should be adopted when the objective is non-differentiable or too expensive to evaluate.

Bibliography

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985. → pages 11

[2] C. Andrieu and E. Moulines. On the ergodicity properties of some adaptive MCMC algorithms. *The Annals of Applied Probability*, 16(3):1462–1505, 2006. → pages 1

[3] C. Andrieu and C. Robert. Controlled MCMC for optimal sampling. Technical Report 0125, Cahiers de Mathematiques du Ceremade, Universite Paris-Dauphine, 2001. → pages 1, 2, 4

[4] C. Andrieu and J. Thoms. A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4):343–373, 2008. → pages 3, 4

[5] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, January 2003. → pages 1, 3

[6] Y. Atchade, E. M. Gersende Fort, and P. Priouret. Adaptive Markov chain Monte Carlo: Theory and methods. Technical report, University of Michigan, 2009. → pages 3

[7] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-23, Department of Computer Science, University of British Columbia, November 2009. → pages 2, 7, 9

[8] P. J. Diggle, J. A. Tawn, and R. A. Moyeed. Model-based geostatistics. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 47(3): 299–350, 1998. → pages 8

[9] A. Gelman, G. O. Roberts, and W. R. Gilks. Efficient Metropolis jumping rules. In J. M. Bernado et al., editors, *Bayesian Statistics*, volume 5, page 599. OUP, 1996. → pages 4

[10] H. Haario, E. Saksman, and J. Tamminen. An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242, 2001. → pages 1, 4

[11] F. Hamze and N. de Freitas. Intracluster moves for constrained discrete-space MCMC. In *Uncertainty in Artificial Intelligence*, 2010. → pages v, 11, 12, 13, 14, 15, 16, 17, 18, 26

[12] K. Kawasaki. Diffusion constants near the critical point for time-dependent Ising models. i. *Phys. Rev.*, 145(1):224–230, 1966. → pages 13

[13] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5 (1):1–25, 1996. → pages 9

[14] D. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Edmonton, Alberta, Canada, 2008. → pages 7

[15] C. E. Rasmussen. Gaussian processes to speed up hybrid Monte Carlo for expensive Bayesian integrals. *Bayesian Statistics*, 7:651–659, 2003. → pages 2

[16] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. → pages 7

[17] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 1st edition, 1998. → pages 1

[18] G. O. Roberts and J. S. Rosenthal. Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367, 2009. → pages 3

[19] D. B. Rubin. Using the SIR algorithm to simulate posterior distributions. In J. M. Bernardo, M. H. DeGroot, D. V. Lindley, and A. F. M. Smith, editors, *Bayesian Statistics 3*, pages 395–402, Cambridge, MA, 1988. Oxford University Press. → pages 9

[20] E. Saksman and M. Vihola. On the ergodicity of the adaptive Metropolis algorithm on unbounded domains. *Annals of Applied Probability*, 20(6): 2178 – 2203, 2010. → pages 1

[21] M. Schonlau, W. J. Welch, and D. R. Jones. Global versus local search in constrained optimization of computer models. *Lecture Notes-Monograph Series*, 34:11–25, 1998. $\rightarrow$ pages 9

[22] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. International Conference on Machine Learning (ICML)*, 2010. $\rightarrow$ pages 2

[23] E. Vasquez and J. Bect. On the convergence of the expected improvement algorithm. Technical Report arXiv:0712.3744v2, arXiv.org, Feb 2008. $\rightarrow$ pages 2

[24] M. Vihola. Grapham: Graphical models with adaptive random walk Metropolis algorithms. *Computational Statistics and Data Analysis*, 54(1): 49 – 54, 2010. $\rightarrow$ pages 1

[25] K. Q. Ye. Orthogonal column Latin hypercubes and their application in computer experiments. *Journal of the American Statistical Association*, 93 (444):1430–1439, 1998. $\rightarrow$ pages 9