Adaptive Markov Chain Monte Carlo: A Review Report[*]

Aritra Majumdar[†] Arkajyoti Bhattacharjee[‡] Sanket Agrawal[§]

Department of Mathematics & Statistics

Indian Institute of Technology Kanpur

May 13, 2021

Abstract

We briefly review the adaptive Markov chain Monte Carlo techniques using some examples. We also talk about the theoretical properties like stationarity and ergodicity of such algorithms.

Contents

[*] As part of the Department elective course MTH707A - Markov Chain Monte Carlo

[†] 191025, M.Sc. Statistics (Final year)

[‡] 201277, M.Sc. Statistics (First year)

[§] 191124, M.Sc. Statistics (Final year)

1 Introduction

Markov chain Monte Carlo (MCMC) algorithms such as Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970) are widely used in sampling from complicated high-dimensional distributions. In particular, developments of MCMC techniques have revolutionized the field of Bayesian inference where one often deals with high-dimensional posterior distributions available only up to a proportionality constant (see Brooks et al., 2011; Meyn and Tweedie, 2012; Robert and Casella, 2013).

For best performances, tuning of associated parameters such as the proposal variance are necessary for any MCMC algorithm. This guarantees an efficient mixing of the chain and hence better estimates of the target. Particularly for proposal variance, it is well known that it should neither be too large nor too small (see Gelman et al., 1996; Roberts et al., 1997; Roberts and Rosenthal, 2001). Small values lead to a very slow moving chain. Then, even though the acceptance rate is very high, the space is explored very slowly. High values for proposal variance generates an active chain but also results in too many rejections. This leads to long periods of immobility in the chain. Hence, the proposal variance should be neither too less nor too high. Or equivalently, one may say that the acceptance rate of an MH algorithm should be neither too close to 0 nor too close to 1. Finding the actual optimal value, however, can be very difficult.

Roberts et al. (1997) show, for random walk type Gaussian proposals and identically decomposable target densities, that the proposal variance should be tuned such that the acceptance rate is roughly 0.234. The conditions on target and proposal distributions have since been relaxed by Roberts and Rosenthal (1998, 2001); Neal et al. (2006); Bédard (2008); Sherlock et al. (2009); Zanella et al. (2017); Yang et al. (2020) among others. So, now we have a tuning guideline which is robust, i.e. tune your proposal variance so that the acceptance rate is roughly 0.234. However, finding such proposal variance is non-trivial and often requires knowledge about the target distribution which is usually not available. In one dimensional settings, a naive approach is to use trial and error and manual tuning. That is, increase the variance when acceptance rate is higher than optimal and vice versa. The problem escalates as the number of dimensions become large. Even in 10 dimensions, there are 55 free variables to tune and more than one configuration may yield the optimal value. Such situations render trial and error approach useless. Another approach is to let the algorithm tune itself on the fly and achieve the optimal acceptance rate. Such algorithms are called *adaptive MCMC* algorithms and they are characterized by modified transition kernels at each step. Haario et al. (2001) proposes an adaptive version of the Metropolis algorithm which at each step, uses the sample estimate of the target covariance matrix as the proposal covariance matrix. They prove, for a version of this algorithm, the ergodicity of the process and show that as the number of iterations become large, the algorithm attains optimal acceptance rate.

Besides tuning for the optimal acceptance rate, Craiu et al. (2009) show that adaptive MCMC perform better than ordinary MCMC in case of multi-modal targets or when the target has *local properties*. Adaptive MCMC can also be significant in outperforming ordinary MCMC in cases of targets with weird supports. For example, a high dimensional problem with highly correlated structure. In such situations, a fixed kernel will result in lots of wasted moves. An adaptive MCMC will, on the other hand, try to learn the structure of the target and adapt the proposal kernel to make intelligent moves (Roberts and Rosenthal, 2009; Mallik and Jones, 2017).

So, the fundamental difference between an adaptive MCMC algorithm and an ordinary MCMC algorithm is that the parameters associated with the proposal kernels are updated at each step for the former while they remain fixed for the latter. Different update mechanisms lead to different algorithms. An update mechanism may utilize information on the history of the chain so far, in which case it is called an *internal adaptation*. Often in internal adaptation, the chain and the parameters together form a Markov chain. On the other hand, update mechanism may depend on parallel chains; then it is called *external adaptation*. Adaptive MCMC are known to not always converge (Rosenthal, 2004; Roberts and Rosenthal, 2007). Haario et al. (2001) was the first to show ergodicity of a particular adaptive algorithm. His results were later generalized by Atchadé et al. (2005); Andrieu et al. (2006); Roberts and Rosenthal (2007) among others.

The present report aims to give a brief introduction to adaptive MCMC and its ergodicity properties. For this purpose, we consider only internal adaptation algorithms. The rest of the report is organized as follows. We describe the notations and terminologies associated with adaptive MCMC in Section 2. In Section 3, we present examples of such adaptive algorithms. We illustrate two different adaptation schemes and discuss their performances on respective targets. Finally in Section 4, we talk about the ergodicity of adaptive MCMC. We illustrate two counter-examples to motivate the need to study ergodicity and present sufficient conditions to ensure the same.

2 Notations and Setup

In this section, we will describe the mathematical framework of adaptive MCMC algorithms with internal adaptations. The discussion in this section has been derived from Roberts and Rosenthal (2007).

Let π be a fixed "target" distribution of interest, i.e. the distribution from which we want to sample, defined on a state space $\mathcal{X}$ endowed with a $\sigma-$field $\mathcal{F}$. Usually, the target

π is high-dimensional and too complicated for other direct sampling methods. Hence, the goal of MCMC is to generate a Markov chain which has the same stationary distribution as π.

As described in the introduction, an adaptive MCMC algorithm tunes itself on the fly by modifying the transition kernel at each step. However, once the transition kernel for a particular step has been fixed, the update mechanism remains similar to that of ordinary MCMC algorithm. Hence, the update step of an adaptive MCMC algorithm can be considered as a combination of two steps. First is to choose a transition kernel for a particular step. This choice may or may not be influenced by the past history of the process. In fact, when the choice is independent of the past, it is called an *independent adaptation.* When the same kernel is chosen for all steps, the algorithm is same as the usual MCMC algorithm. Second, given the chosen kernel, update the value of the process using the mechanism of an ordinary MCMC algorithm.

Let us now formalize the above description. Firstly, we need a space of transition kernels on the space $\mathcal{X}$ from which the algorithm will choose at any given step. We assume that this space remains fixed i.e. at each step, all options are available at all steps. Further, to ensure stationarity, it makes more sense to consider those kernels which are stationary with respect to π and are also ergodic. Hence, we let $\{P_\gamma\}_{\gamma \in \mathcal{Y}}$ be a collection of Markov chain kernels on $\mathcal{X}$, such that for each $\gamma \in Y$,

$$\begin{aligned} (\pi P_\gamma)(\cdot) &= \pi(\cdot), && \text{(stationary)}, \\ \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| &\to 0 \text{ as } n \to \infty, && \text{(ergodic)}, \end{aligned}$$

were $\|\mu - \nu\|$ denotes the total variation distance between μ and ν. This implies that if we were to choose and fix any of the P_γ, we would get a valid MCMC algorithm and the resulting Markov chain will converge to π. An AMCMC algorithm, on the other hand, might choose a different γ at each step. That is, at each time point n the value of γ is given by a $\mathcal{Y}-$valued random variable Γ_n. Description of the random variable Γ_n depends on the choice of the adaptation scheme.

So, we have for $n = 0, 1, 2, \ldots$, an $\mathcal{X}-$valued random variable X_n and a $\mathcal{Y}-$valued random variable Γ_n. Let,

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \Gamma_0, \ldots, \Gamma_n),$$

be the filtration generated by the process $\{(X_n, \Gamma_n)\}$. Then, according to the setup described above,

$$P[X_{n+1} \in B | X_n = x, \Gamma_n = \gamma, \mathcal{G}_{n-1}] = P_\gamma(x, B), \qquad x \in \mathcal{X}, \gamma \in \mathcal{Y}, B \in \mathcal{F}.$$

The conditional distribution of Γ_n given $\mathcal{G}_{n-1}$ is determined by the adaptation rule used. Usually, the adaptation is so chosen that the joint process $\{(X_n, \Gamma_n)\}$ becomes Markovian. Now, fix $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$ and define,

$$\begin{aligned} T(x, \gamma, n) &= \|P[X_n \in \cdot | X_0 = x, \Gamma_0 = \gamma] - \pi(\cdot)\| \\ &= \sup_{B \in \mathcal{F}} |P[X_n \in B | X_0 = x, \Gamma_0 = \gamma] - \pi(B)|. \end{aligned}$$

So, starting from x with initial kernel P_γ, $T(x, \gamma, n)$ is the total variation distance between the target and the adaptive process at time n. Call the adaptive algorithm ergodic if $\lim_{n\to\infty} T(x, \gamma, n) = 0$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$ (Roberts and Rosenthal, 2007). Determining ergodicity for an adaptive MCMC algorithm is a non-trivial problem. We will take this discussion briefly in Section 4.

3 Examples

We will now discuss a few examples of adaptive MCMC algorithms. We begin with some special cases which are very common and a reader with introductory course on MCMC will certainly know about. These are,

- *Ordinary MCMC*, i.e. Γ_n takes a fixed single value for all n. Mathematically, it is written as, $\Gamma_n = y$ for some fixed $y \in \mathcal{Y}$ for all n.

- *Deterministic Scan Component-wise updates*, recall that in a component-wise update mechanism, only one coordinate is updated at one time. Hence, in a $d-$dimensional problem if P_i is the transition kernel that updates only the $i-$th coordinate, then a deterministic scan component wise update is also an adaptive MCMC algorithm with Γ_n being a deterministic variable such that $\Gamma_n = P_{n \bmod d}$.

- *Random Scan Component-wise updates*, recall that a random scan update randomly chooses 1 out of the d coordinates and update the chosen coordinate. Hence, a random scan algorithm is also an adaptive MCMC algorithm with a special adaptation scheme, i.e. each Γ_n is independently and identically distributed as Uniform$\{P_1, \ldots, P_d\}$.

Observe that in all of the above three cases, the random variable Γ_n does not depend on the present or the history of the process. In general, we call an adaptive MCMC algorithm an *independent adaptation* if, for all n, Γ_n is independent of the current and past values of the process. The above three cases are thus particular examples of independent adaptation. Independent adaptations always preserve stationarity but may fail to preserve convergence (Roberts and Rosenthal, 2007).

Another special case of adaptive MCMC is to stop adaptation after a finite number of steps. The stopping time may be fixed or random, in which case it is assumed to be finite with probability 1. Clearly, then after the stopping time, the transition kernel remains fixed. Hence, intuitively, *finite adaptation* must always preserve convergence. This is in fact true and can be shown by conditioning on the process at the stopping time and then integrating over all possible values at the stopping time (Roberts and Rosenthal, 2007).

3.1 Adaptive Metropolis Algorithm

The adaptive Metropolis Algorithm of Haario et al. (2001) is an adaptive version of the Gaussian random walk MH algorithm and uses an internal adaptive scheme. At each step, the proposal kernel is updated using the entire history of the process.

Suppose π is the $d-$dimensional target distribution with associated Lebesgue density, also denoted by $\pi(\cdot)$. Suppose that at step n, we have sampled states $X_0, X_1, \ldots, X_n$. Then a value Y is proposed using a proposal distribution $Q_n(x, \cdot) = Q(x, \cdot|\Gamma_n(X_0, X_1, \ldots, X_n))$. Note that this proposal may now depend on the entire history of the process. The proposed value Y is accepted with probability,

$$\alpha(X_n, Y) = \min\left\{1, \frac{\pi(Y)}{\pi(X_n)}\right\},$$

in which case we set $X_{n+1} = Y$, otherwise $X_{n+1} = X_n$. Note that the acceptance probability is similar to the the acceptance probability for usual MH algorithm with symmetric proposals. However, here the proposals are usually not symmetric. As a result, the generated process loses its stationarity as well as its Markovian nature and the exact convergence of the process needs to be studied separately. We will talk about these issues in a later section.

The proposal distribution $Q(x, \cdot|\Gamma_n(X_0, X_1, \ldots, X_n))$ utilized in the adaptive MH algorithm of Haario et al. (2001) is the Gaussian distribution centered at the current point X_n and covariance $\Gamma_n(X_0, X_1, \ldots, X_n)$ such that,

$$\Gamma_n(X_0, X_1, \ldots, X_n) = \begin{cases} \Gamma_0, & n \leq n_0, \\ s_d \text{cov}(X_0, \ldots, X_n) + s_d \epsilon I_d, & n > n_0, \end{cases}$$

where s_d is a parameter that depends only on the dimension d and $\epsilon > 0$ is a constant. I_d is the identity matrix of order d. Γ_0 is some initial strictly positive definite, covariance matrix (may be based on our prior knowledge) and n_0 is the initial period of non adaptation. The choice of n_0 is free and in a sense reflects our trust in the initial covariance Γ_0 (Haario et al., 2001). Finally, $\text{cov}(x_0, x_1, \ldots, x_k)$ denotes the sample

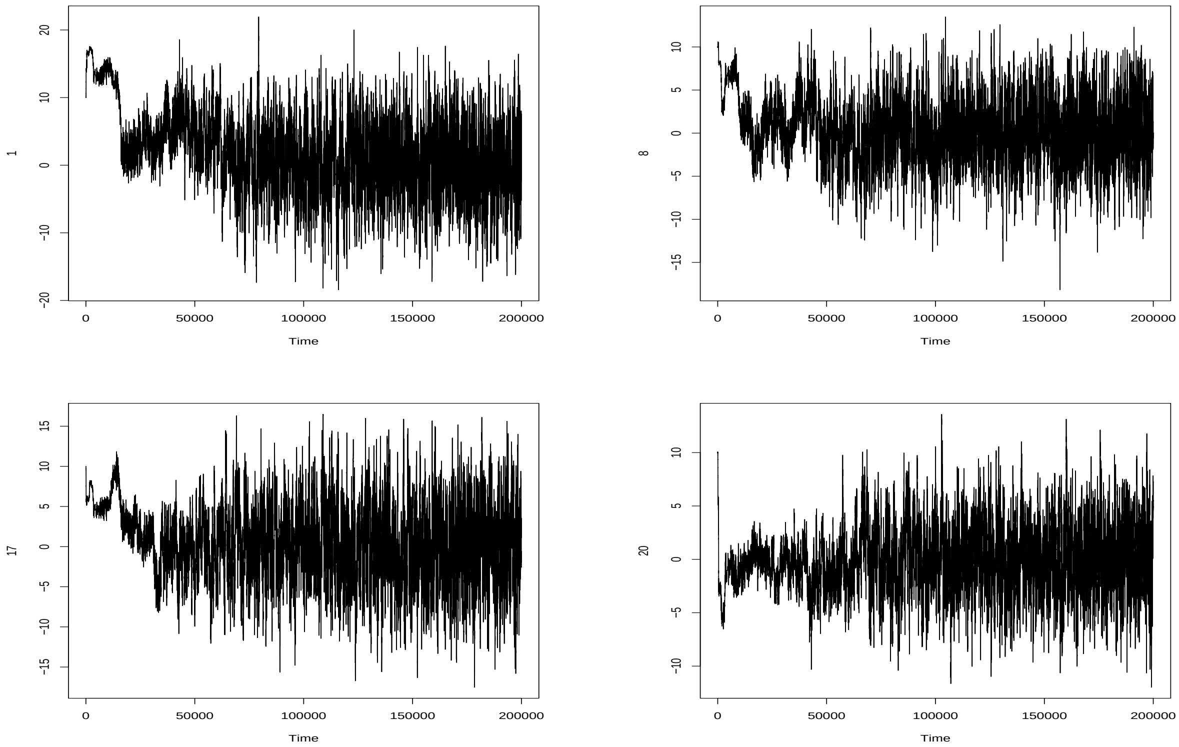

Figure 1: Trace plots for different coordinates of the AM algorithm against number of iterations. (In clockwise direction) Coordinate 1, 8, 17, 20.

covariance matrix determined by points $x_0, \ldots x_k \in \mathbb{R}^d$ as,

$$\operatorname{cov}(x_0, x_1, \ldots, x_k) = \frac{1}{k}\left(\sum_{i=0}^{k} x_i x_i^T - (k+1)\bar{x}_k \bar{x}_k^T\right),$$

where $\bar{x}_k = (\sum_{i=0}^{k} x_k)/(k+1)$. Haario et al. (2001) further assumes that the target distribution is supported on a bounded subset S of $\mathbb{R}^d$ and also chooses $s_d = (2.38)^2/d$ from Roberts et al. (1997). With restricted support, proposals outside S are outright rejected and only those which lie in S are considered for acceptance.

We now present a different version of the above algorithm from Roberts and Rosenthal (2009). They do not require the target to be supported on a bounded set. In this case, define $\Gamma_n(X_0, X_1, \ldots, X_n) = \operatorname{cov}(X_0, \ldots, X_n)$. Then the proposal distribution $Q_n(x, \cdot)$ is given as,

$$Q_n(x, \cdot) = \begin{cases} N(x, (0.1)^2 I_d/d) & n \leq 2d, \\ (1-\beta)N(x, (2.38)^2\Gamma_n/d) + \beta N(x, (0.1)^2 I_d/d) & n > 2d, \end{cases}$$

for some $\beta > 0$. Notice that in both versions of the above algorithm, we take the proposal covariance to be $(2.38)^2/d$ times the empirical covariance matrix (plus some adjustment).

This is due to the result that in high-dimensional settings, a Gaussian proposal is optimal when the covariance matrix is same as $(2.38)^2\Sigma/d$ where Σ is the target covariance matrix (see Roberts et al., 1997; Roberts and Rosenthal, 2001). Since Σ is not known, empirical covariance matrix is employed in an effort to approximate this.

For our implementation, we let $\pi = N(0, MM^T)$ where M is a $d \times d$ matrix, elements of which are randomly generated as independently and identically distributed $N(0,1)$ random variates. This ensures that the target covariance is highly erratic and consequently difficult to sample from in high dimensions. In particular, we let $d = 20$.

Figure 1 shows the trace plot of 200000 iterations for the above algorithm for 4 different coordinates [1]. To emphasize the role of adaptation, we choose the starting values for each coordinate to be away from the center at 10. In all of the four plots, it can be seen that the algorithm begins settling down around the center and starts mixing rapidly after about 50000 steps. One may notice in initial stages, that the algorithm tries to find its way from a low probability region to a higher probability region by taking very small steps. As it comes closer to 0, it also increases fluctuations as it begins to learn the true covariance. Even still, for most of the initial part of the process, target covariance matrix is vastly underestimated by the algorithm. It takes about 80000 iterations to learn the true target covariance matrix and settle down to rapid mixing. This number does sounds large, but is still good when compared to the number of parameters the algorithm is learning, which is 210. From this, we conclude that the AM algorithm does converges to a well-mixing process sampling from π.

Note that we limit our observations to trace plots of the algorithm only. There are other technical ways too to assess the performance of adaptation. Typically, these are based on a quantitative comparison between the sample covariance matrix and the target covariance matrix (see Roberts and Rosenthal, 2009).

3.2 A banana-shaped distribution

Let us now consider an irregularly shaped distribution and see how AM algorithm performs for this as a target. The "banana-shaped" distribution with the density function

$$f_B(x_1, \ldots, x_d) \propto \exp\left\{-x_1^2/200 - \frac{1}{2}(x_2 + Bx_1^2 - 100B)^2 - \frac{1}{2}(x_3^2 + x_4^2 + \cdots + x_d^2)\right\},$$

[1] Codes for all plots and tables are available at https://github.com/ArkaB-DS/Adaptive-MCMC

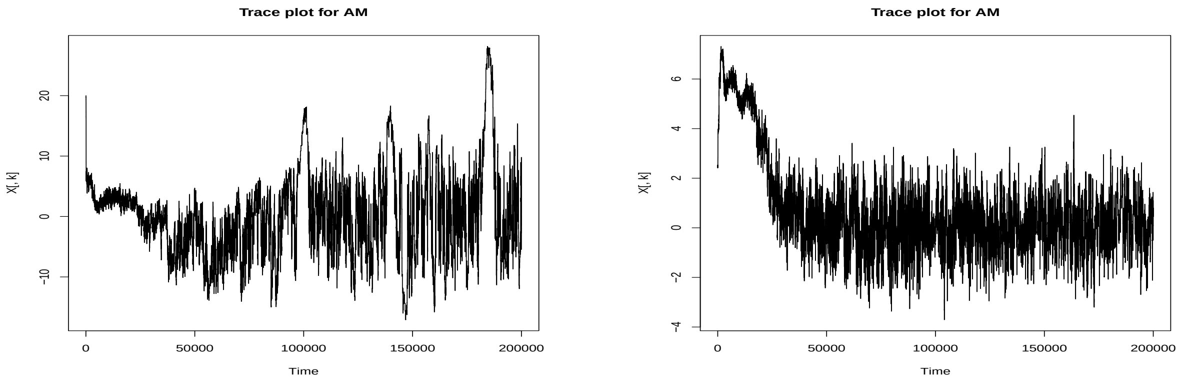

Figure 2: Trace plots for first (left) and third (right) coordinates in the banana-shaped example.

was proposed in Haario et al. (2001). $B > 0$ is called the "bananicity" constant. The above distribution is obtained by applying a measure preserving transformation

$$\phi_B(x_1, \dots, x_d) = (x_1, x_2 + Bx_1^2 - 100B, x_3, ..., x_d),$$

to a zero mean Gaussian distribution with covariance matrix $\Sigma = \text{diag}(100, 1, \dots, 1)$.

For our implementation, we take $d = 20$ and $B = 0.1$ and run the algorithm for 200000 iterations. Figure 2 shows trace plots for the first and third components of the process. Here again, we start away from the center. Clearly, adaptation helps the algorithm find its way back to the center improving mixing as the number of iterations increase. The true variance of the third coordinate is 1 while that of the first coordinate is 100. This difference is also visible in the trace plots of the two coordinates. The third coordinate stabilizes after 60000 iterations and starts mixing well. The first coordinate, although concentrates around the true center, fails to achieve as good mixing as the third coordinate. This is due to irregular structure of the covariance matrix. In fact, the true variance of the first component is again vastly underestimated by the algorithm even after 200000 steps. But even though the mixing is not great, it still shows a significant improvement over time.

3.3 Adaptive Metropolis within Gibbs

In both of the previous examples, we employed the same adaptation scheme of updating the covariance matrix with the sample covariance matrix. Such adaptation is aimed at chasing the target covariance matrix, or in fact an optimum proposal covariance matrix. In this example, we will show a different adaptation scheme that chases the optimal acceptance rate instead.

The example is from Roberts and Rosenthal (2009) and the setup is that of a Bayesian one-way random effects model,

$$Y_{i,j} \sim N(\theta_i, V); \quad j = 1, \ldots, r_i, i = 1, \ldots, K,$$

with prior specifications as follows,

$$\theta_i \sim \text{Cauchy}(\mu, A),\ i = 1, \ldots, K; \quad \mu \sim N(0, 1); \quad A \sim IG(1, 1); \quad V \sim IG(1, 1),$$

where $IG(a, b)$ is the inverse gamma distribution with parameters a, b. Given the data Y, we obtain a posterior distribution π on the $(K + 3)$-dimensional random vector $(A, V, \mu, \theta_1, \ldots, \theta_K)$.

We employ a Metropolis-within-Gibbs algorithm to sample from π; sampling from full conditional of each of the $K + 3$ variables turn by turn. When it comes to V, we notice that the full conditional distribution of V comes out to be a tractable IG distribution. So for V, we sample directly from its full conditional. For the remaining $K+2$ variables, we update their current values by adding a $N(0, \sigma^2)$ component via an MH update step. This iteration is repeated a large number of times and with a hope to converge to the actual posterior distribution π. Now, the question of interest is what should be the optimal choice of σ^2 and should it be different for different variables. An adaptive algorithm helps answer these questions.

Recall that we mentioned earlier that in this example our adaptation strategy will be to chase the optimal acceptance rate. For one-dimensional proposals, the optimal acceptance rate (under some conditions) is roughly 0.44 (Gelman et al., 1996; Roberts et al., 1997). Hence, our approach will be to increase the proposal variance for a given variable if the corresponding acceptance rate is greater than 0.44 and vice-versa. Mathematically, for each variable i of the $K + 2$ variables (V does not requires adaptation as it is sampled directly from its full condiitonal), we create an associated variable ls_i which is the logarithm of the standard deviation to be used to propose a normal increment to the variable i. Initial values of ls_i are set to be 0 for all i. Adaptation is done after every 50 iterations. So after the n^{th} "batch" of 50 iterations, ls_i is increased by $\min(0.01, n^{-1/2})$ if the proportion of acceptances for variable i is greater than 0.44. Otherwise, it is decreased by the same amount. Notice that the amount of adaptation goes to 0 as $n \to \infty$ for each variable.

To test the algorithm, we take $K = 10$ and let r_i vary between 5 to 500. Towards observed data, we generate $Y_{ij} \sim N(i - 1, 10^2)$ independently. Table 1 summarizes the acceptance rate for all 12 variables after 100000 batches of 50 iterations each. Observe that for each variable, the acceptance rate is very close to the optimum value 0.44. Hence, we can conclude from here that adaptation is very effective in helping the algorithm attain

A	μ	θ_1	θ_2	θ_3	θ_4
0.4373	0.4401	0.4392	0.4398	0.4400	0.4399
θ_5	θ_6	θ_7	θ_8	θ_9	θ_{10}
0.4402	0.4398	0.4399	0.4401	0.4404	0.4395

Table 1: Acceptance rates of each variable for adaptive Metropolis-within-Gibbs sampler

the optimal proportion of acceptances.

4 Ergodicity

We mentioned in the Introduction that adaptive MCMC algorithms do not always preserve the convergence of the generated chain. However, if some conditions are satisfied, ergodicity of the chain can be guaranteed. We will briefly discuss those conditions in this section.

What makes the analysis of adaptive MCMC algorithms challenging is that now we have a changing transition kernel at each step. And also, since the transition kernels are allowed to depend on the current and all the past values of the chain, the Markovian property is lost. Hence, the usual results for Markov chains do not directly apply in this case. There are however special situations such as independent adaptation and finite adaptation (see Section 3), where the theory of Markov chains can be directly applied to ensure convergence of the corresponding algorithms. But a more rigorous analysis (usually via coupling arguments) is required to study the general framework.

Recall the setup from Section 2 and that an adaptive algorithm $\{X_n, \Gamma_n\}$ is said to be ergodic if for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$,

$$\lim_{n\to\infty} T(x,\gamma,n) = \lim_{n\to\infty} \|P[X_n \in \cdot\,|X_0 = x, \Gamma_0 = \gamma] - \pi(\cdot)\| = 0.$$

Haario et al. (2001) proved the convergence of a version of adaptive Metropolis algorithm (described in Section 3.1) using a mixingale approach. Following the same mixingale approach, Atchadé et al. (2005) extended the result to general adaptive MCMC algorithms and on unbounded support $\mathcal{X}$ under some assumptions. Most important of them was that each kernel $P_\gamma, \gamma \in \mathcal{Y}$ is ergodic for some invariant distribution π_γ (note that we assume in Section 2 that $\pi_\gamma = \pi$ for all $\gamma \in \mathcal{Y}$). Further, the rate of convergence of each kernel P_γ to π_γ was required to be simultaneously uniform-in-time (geometric or subgeometric) i.e. convergence rate did not depend on γ. Finally, it was also required that as the algorithm proceeds, transition kernels become more and more stable i.e. the amount of adaptation decreases. They also proposed another adaptation scheme based on random walk Metropolis algorithm which finds the optimal scale parameter (using results from Roberts et al. (1997)) and proved ergodicity for it. Andrieu et al. (2006) further extended the result to unbounded parameter space $\mathcal{Y}$ and proved a law of large numbers using different proof technique utilizing results from martingale limit theory. They also proved a central limit theorem result. Although, the two papers provided a generalization to Haario et al. (2001) result removing many restrictions and limitations, they also imposed other technical hypotheses difficult to verify in practice. Roberts and Rosenthal (2007) then obtained somewhat simpler and intuitive conditions which could still ensure ergodicity of algorithms for the specified target distribution. They used a coupling construction to prove their result. Before talking about those conditions, we will first look at two simple counter examples where an adaptive MCMC algorithm fails to converge or preserve stationarity. These should serve as a motivation to study ergodicity separately.

4.1 Counterexamples

The first one is from Andrieu et al. (2006) and illustrates that an adaptive algorithm can destroy stationarity of the target instead of improving the convergence. Let $\mathcal{X} = \{1,2\}$ and consider for $\gamma, \gamma(1), \gamma(2) \in \mathcal{Y} = (0,1)$, the following two transition probability matrices,

$$P_\gamma = \begin{bmatrix} 1-\gamma & \gamma \\ \gamma & 1-\gamma \end{bmatrix}, \qquad \bar{P} = \begin{bmatrix} 1-\gamma(1) & \gamma(1) \\ \gamma(2) & 1-\gamma(2) \end{bmatrix}.$$

Then, it is easy to check that for all $\gamma \in \mathcal{Y}$, $\pi = (1/2, 1/2)$ is stationary, i.e. it satisfies $\pi P_\gamma = \pi$. However, if we let Γ_n to be a function of the current state given by $\gamma : \{1,2\} \to (0,1)$ i.e. if we take $\Gamma_n = \gamma(X_n)$, then we obtain $\bar{P}$ as the associated transition kernel having

$$\bar{\pi} = \left(\frac{\gamma(2)}{\gamma(1)+\gamma(2)}, \frac{\gamma(1)}{\gamma(1)+\gamma(2)} \right),$$

as the invariant distribution. This is not same as π unless $\gamma(1) = \gamma(2) = 1/2$. Hence, one recovers π when the dependence of γ on the current state X_n is removed or vanishes with the number of iterations (Andrieu et al., 2006).

Another example is due to Roberts and Rosenthal (2007); an interactive Java applet is also available from Rosenthal (2004). Let $\mathcal{X} = \{1,2,3,4\}$ with $\pi(1) = \epsilon, \pi(2) = \epsilon^3$, and $\pi(3) = \pi(4) = (1-\epsilon-\epsilon^3)/2$ for some small $\epsilon > 0$. Let $\mathcal{Y} = \{1,2\}$ and for $\gamma \in \Gamma$, let P_γ be the MH transition kernel for π, with proposal distribution

$$Q_\gamma(x, \cdot) = \text{Uniform}\{x-\gamma, x-\gamma+1, \ldots, x-1, x+1, \ldots, x+\gamma\}.$$

Whenever the proposed value lies outside $\mathcal{X}$, it is rejected outright. The adaptive scheme is defined as follows. For $n = 1, 2, 3, \ldots$, given X_n and Γ_n, if the next proposal is accepted set $\Gamma_{n+1} = 2$. Otherwise, if the next proposal is rejected set $\Gamma_{n+1} = 1$. In words, this means that whenever a proposal is accepted, γ is increased to 2 and whenever it is rejected, γ is decreased to 1. Each of the P_γ is reversible with respect to π. It can also be shown that the algorithm has a positive and in particular $O(\epsilon)$ probability of ever reaching the configuration $\{x = \gamma = 1\}$. Further we have,

$$P[X_1 = 1, \Gamma_1 = 1 | X_0 = 1, \Gamma_0 = 1] = 1 - \epsilon^2/2,$$

i.e. the probability of leaving $\{x = \gamma = 1\}$ once the chain is just $O(\epsilon^2)$. Some more technical arguments lead us to the following,

$$\lim_{\epsilon \downarrow 0} \lim_{n \to \infty} T(x, \gamma, n) \geq 1.$$

In particular, for any $\delta > 0$, there exists $\epsilon > 0$ with $\lim_{n \to \infty} T(x, \gamma, n) \geq 1 - \delta$, so the algorithm does not converges to the specified target at all.

4.2 Diminishing adaptation and Containment

Now we know why a separate investigation of ergodicity of adaptive MCMC algorithms is necessary. Roberts and Rosenthal (2007) proved the asymptotic convergence in TV distance and a weak law of large numbers for bounded functions under two conditions described as follows. For any $\epsilon > 0$, write

$$M_\epsilon(x, \gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x, \cdot) - \pi(\cdot)\| \leq \epsilon\}$$

for the $\epsilon-$convergence time of the kernel P_γ when beginning in state $x \in \mathcal{X}$. Say, that an adaptive algorithm satisfies *Containment* condition if,

$$\{M_\epsilon(X_n, \Gamma_n)\}_{n=0}^{\infty} \quad \text{is bounded in probability,} \quad \epsilon > 0,$$

and a *Diminishing Adaptation* condition if,

$$\lim_{n \to \infty} \sup_{x \in \mathcal{X}} \|P_{\Gamma_{n+1}}(x, \cdot) - P_{\Gamma_n}(x, \cdot)\| = 0 \quad \text{in probability.}$$

Now, Containment (also known as *Bounded Convergence*) condition is a technical condition but is still intuitive. Having convergence times bounded in probability ensures that all kernels P_γ are usually close to π after a large number of steps, say N. Also, Diminishing adaptation condition ensures that the amount of adapting at the n^{th} iteration goes to 0 as $n \to \infty$. Together, both the conditions ensure that after a large number of steps transition kernels are usually close to the target and to each other. The proof is based on this observation and utilizes coupling constructions.

Diminishing adaptation is more natural and can be easily satisfied by appropriately constructing adaptation schemes. Both the counterexamples considered in this report fail to satisfy Diminishing adaptation. In fact, when $\mathcal{Y}$ is finite, and adaptation is not finite, the only way to satisfy the condition is to ensure that probability of adaptation goes to 0 as $n \to \infty$. Containment is more abstract and difficult to verify. Roberts and Rosenthal (2007) proved that Containment is satisfied whenever $\mathcal{X} \times \mathcal{Y}$ is finite, or is compact in some topology in which either the transition kernels or the proposal kernels (along with a continuous acceptance function) have jointly continuous densities. Both, Containment and Diminishing adaptation, are satisfied for algorithms considered in Section 3 (see Roberts and Rosenthal, 2009). Although, it is also possible to construct pathological counterexamples where the containment condition is not satisfied and the algorithm performs very poorly (see e.g. Łatuszyński and Rosenthal, 2014). Finally, it is important to note at this point that the above conditions are sufficient conditions. No comment about their necessity is being made.

5 Conclusions

We gave a brief overview of adaptive MCMC in this report. The aim was to provide an introduction to adaptive MCMC to someone already familiar with MCMC algorithms and having some working knowledge of them. Beginning with the motivation for doing it we first described its mathematical framework. We illustrated two different adaptation schemes for three different targets. One of them was aimed at learning the true target covariance matrix while the other one chased the optimal acceptance rate. It is also possible to incorporate both approaches in one and simultaneously chase the optimal acceptance rate and learn the target covariance. We saw that adaptive algorithms converge to well-mixing processes and can be very efficient in high dimensions. They are also advantageous when starting in bad places and when targets are irregularly shaped. However, we limited ourselves to only internal adaptation schemes. One interesting way to adapt is to run two parallel chains X and Y, and use the information of Y to adapt kernels of X (see Atchade et al., 2011).

We saw that the adaptation scheme for Metropolis-within-Gibbs sampler considered in Section 3.3 is an effective scheme as it guides the algorithm towards the optimal acceptance rate. It will, in addition, also be interesting to see where the proposal variances for each variable settle. That is, an analysis of what values of proposal variance correspond to the optimal acceptance rates for different variables and whether they comply to the theoretical results of Gelman et al. (1996).

Although the algorithms discussed in Section 3 perform sufficiently well, in many cases it is also desirable that the algorithm adjust its proposal covariance depending on the current state of the process. This calls for state dependent adaptation schemes (see Roberts and Rosenthal (2009, Section 4)).

Finally, it will be worthwile to actually verify the ergodicity conditions for the algorithms considered in Section 3. In the first scheme, proposal covariance matrix is updated at each step and the amount of update is of the order $O(1/n)$ which goes to 0 as $n \to \infty$. In the second scheme also, we saw that the amount of adaptation vanishes. Hence, in both situations, diminishing adaptation is verified. Containment, on the other hand, is more technical. Roberts and Rosenthal (2009) argue that when the parameter space is bounded, Containment is satisfied for all those target densities which are log-concave outside an arbitrary bounded region. It can be shown that these conditions are satisfied for the three situations considered in this report. An interested reader is then encouraged to see for himself how these conditions guarantee Containment.

Acknowledgement

We express our sincerest gratitude towards Prof. Dootika Vats who gave us this wonderful opportunity to study such an interesting topic in the form of a course project. This report wouldn't have been possible without her constant support and guidance throughout the duration of the project.

References

Andrieu, C., Moulines, É., et al. (2006). On the ergodicity properties of some adaptive MCMC algorithms. *The Annals of Applied Probability*, 16(3):1462–1505.

Atchade, Y., Fort, G., Moulines, E., and Priouret, P. (2011). Adaptive Markov chain Monte Carlo: Theory and methods. *Bayesian time series models*, pages 32–51.

Atchadé, Y. F., Rosenthal, J. S., et al. (2005). On adaptive Markov chain Monte Carlo algorithms. *Bernoulli*, 11(5):815–828.

Bédard, M. (2008). Optimal acceptance rates for Metropolis algorithms: Moving beyond 0.234. *Stochastic Processes and their Applications*, 118(12):2198–2222.

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov chain Monte Carlo*. CRC press.

Craiu, R. V., Rosenthal, J., and Yang, C. (2009). Learn from thy neighbor: Parallel-chain and regional adaptive MCMC. *Journal of the American Statistical Association*, 104(488):1454–1466.

Gelman, A., Roberts, G. O., and Gilks, W. R. (1996). Efficient Metropolis jumping rules. *Bayesian statistics*, 5:599–608.

Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242.

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

Łatuszyński, K. and Rosenthal, J. S. (2014). The Containment condition and Adapfail algorithms. *Journal of Applied Probability*, 51(4):1189–1195.

Mallik, A. and Jones, G. L. (2017). Directional Metropolis-Hastings. *arXiv preprint arXiv:1710.09759*.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Meyn, S. P. and Tweedie, R. L. (2012). *Markov chains and Stochastic stability.* Springer Science & Business Media.

Neal, P., Roberts, G., et al. (2006). Optimal scaling for partially updating MCMC algorithms. *The Annals of Applied Probability*, 16:475–515.

Robert, C. and Casella, G. (2013). *Monte Carlo statistical methods*. Springer Science & Business Media.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268.

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical science*, 16(4):351–367.

Roberts, G. O. and Rosenthal, J. S. (2007). Coupling and Ergodicity of Adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probability*, 44(2):458–475.

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

Rosenthal, J. S. (2004). Adaptive MCMC Java Applet.

Sherlock, C., Roberts, G., et al. (2009). Optimal scaling of the random walk Metropolis on elliptically symmetric unimodal targets. *Bernoulli*, 15:774–798.

Yang, J., Roberts, G. O., and Rosenthal, J. S. (2020). Optimal scaling of random-walk Metropolis algorithms on general target distributions. *Stochastic Processes and their Applications*, 130(10):6094 – 6132.

Zanella, G., Bédard, M., and Kendall, W. S. (2017). A Dirichlet form approach to MCMC optimal scaling. *Stochastic Processes and their Applications*, 127:4053–4082.