A Comparative Study of MCMC Adaptive Schemes with Stopped and Diminishing Adaptation

Bowen Liu[1][0000−0003−0567−5528], Edna Milgo[1,2][0000−0002−7874−6609], Nixon Ronoh[1,2][0000−0001−8699−7494], and Bernard Manderick[1][0000−0002−9020−0510]

[1] Artificial Intelligence Lab, Vrije Universiteit Brussel, Belgium
[2] Moi University, Eldoret, Kenya

Abstract. Markov chain Monte Carlo (MCMC) techniques play a vital role in sampling from complex, high-dimensional target distributions. However, the optimization of the proposal distribution for efficient sampling poses a challenging task, which is where adaptation becomes significant. This paper presents a comparative analysis of three adaptive sampling strategies: Metropolis Gaussian Adaptation (MGaA), Metropolis Covariance Matrix Adaptation Evolution Strategy (MCMA), and Adaptive Metropolis (AM). It is noteworthy that incorrect implementation of adaptation can compromise the ergodicity of MCMC samplers, which is essential for generating unbiased samples and converging to the target distribution. To address this concern, two strategies, Stopped Adaptation (SA) and Diminishing Adaptation (DA), are introduced within the adaptive sampler framework to uphold ergodicity. Through a comprehensive evaluation across diverse test distributions, this research assesses the performance of MGaA, MCMA, and AM samplers in various scenarios. By comparing their strengths and capabilities, the study provides valuable insights into effective approaches for sampling from complex distributions.

Keywords: Adaptive MCMC · MGaA · MCMA · Stopped Adaptation · Diminishing Adaptation

1 Introduction

Markov Chain Monte Carlo (MCMC) algorithms are extensively employed in Bayesian inference to sample from the target distributions [9, 15]. The core principle of MCMC is to construct a Markov chain where the target distribution serves as the invariant distribution, irrespective of the initial state. After achieving convergence, samples can be generated to estimate the important statistics like the distribution's mean and covariance.

Metropolis-Hastings (MH) algorithm stands out as the most prevalent MCMC technique, with many others being extensions of MH [3]. Tuning an appropriate proposal is crucial for the effectiveness of the MH algorithm. Even though a finely-tuned proposal distribution can significantly enhance the algorithm's performance, finding the ideal proposal in MH often proves to be a complex

task. As a solution, adaptive MCMC methods have been introduced to autonomously fine-tune the proposal distribution.

A notable example is Adaptive Metropolis (AM) [6], which dynamically adjusts the proposal distribution's covariance through empirical covariance calculations involving past chain samples. Furthermore, stochastic optimization techniques like Gaussian Adaptation (GaA) and Covariance Matrix Adaptation Evolution Strategy (CMAES) also adapt the covariance of the search distribution within the optimization process, aligning with the adaptation in adaptive MCMC samplers. Thus, the conversion of GaA and CMAES into MCMC sampling stands out as a feasible approach for refining proposal distribution, termed MGaA [13] and MCMA [12]. Actually, the adaptation mechanisms differ between AM, MGaA, and MCMA. AM updates the covariance based on the entire historical sample set. MGaA adjusts the covariance by maximizing the entropy of the search distribution. Conversely, MCMA tailors the covariance to heighten the likelihood of discovering favorable samples in ensuing iterations.

Moreover, adaptation schemes can frequently fall short in maintaining the stationarity of the target distribution. To uphold the stationary distribution, two methods come into play: Stopped Adaptation (SA) and Diminishing Adaptation(DA) [1, 16, 2]. Regarding the already established adaptive MCMC sampling techniques, DA has been proficiently employed in ensuring the convergence of AM sampling. However, concerning the current MGaA and MCMA algorithms, the integration of both SA and DA remains relatively unexplored. In this research, we introduce an innovative method by fusing SA and DA into the MGaA and MCMA sampling techniques. This study undertakes a performance comparison of AM sampling against MGaA and MCMA sampling using a range of benchmark target distributions to gauge the efficacy of adaptation. As for MGaA and MCMA sampling, we compare the performance of different adaptation schemes. We first look at the standard MGaA alongside its four variants namely: MGaA with SA and MGaA with three different rates of DA. Additionally, we also compare the standard MCMA with its four variants namely: MCMA with SA and MCMA with three different rates of DA.

2 Adaptive MCMC

2.1 Metropolis-Hastings Algorithm

Let $\pi(\mathbf{x})$ denote the target distribution. The MH algorithm initiates by selecting a random initial sample $\mathbf{x}_0$. At each iteration n, MH generates a candidate $\mathbf{x}^*$ from the proposal distribution $q(\mathbf{x}; \mathbf{x}_n)$. Subsequently, MH determines whether to transition to the candidate state $\mathbf{x}^*$ or remain at the current state $\mathbf{x}_n$. This decision is based on the MH acceptance probability, ensuring that the resulting Markov chain is reversible. The MH acceptance probability can be expressed as follows:

$$\alpha(\mathbf{x}^*; \mathbf{x}_n) = \min\left\{1, \frac{\pi(\mathbf{x}^*)q(\mathbf{x}_n; \mathbf{x}^*)}{\pi(\mathbf{x}_n)q(\mathbf{x}^*; \mathbf{x}_n)}\right\} \tag{1}$$

In the case of a symmetric proposal distribution, i.e., $q(\mathbf{x}^*; \mathbf{x}_n) = q(\mathbf{x}_n; \mathbf{x}^*)$, the acceptance probability can be simplified as $\alpha(\mathbf{x}^*; \mathbf{x}_n) = \min\{1, \pi(\mathbf{x}^*)/\pi(\mathbf{x}_n)\}$.

When considering the proposal distribution $q(\mathbf{x}; \mathbf{x}_n)$, a commonly used choice is the Gaussian distribution. The Gaussian distribution is favored because of its symmetry, which simplifies the calculation of acceptance probabilities. Moreover, sampling from a Gaussian distribution is relatively straightforward. Unless stated otherwise, the default proposal distribution throughout the paper is assumed to be Gaussian distribution. If the target is also Gaussian distribution, an optimal proposal distribution can be determined theoretically [17]. Given a Gaussian proposal and Gaussian target distribution the optimal scale in a particularly large dimensional context is $\sigma_{opt} = (2.38/\sqrt{d})$ where d is the dimension of the state space [4]. While in most cases, the target distribution is non-Gaussian, and there is no one-size-fits-all optimal proposal distribution. As a result, adaptive MCMC algorithms have been proposed to automatically tune the proposal distribution and improve its efficiency [8, 6].

2.2 Generic Framework of Adaptive MCMC

Adaptive MCMC algorithms, as the extensions of the Metropolis-Hastings (MH) algorithm, enhance the sampling process by dynamically adjusting the proposal distribution based on the accepted samples. The general framework of adaptive MCMC sampling is depicted in Algorithm 1.

Algorithm 1: Generic Adaptive MCMC Sampling

Input: Target $\pi(\mathbf{x})$, Initial state $\mathbf{x}_0$, Initial proposal distribution $q(\mathbf{x}; \mathbf{x}_0)$
Output: Sequence of generated samples $(\mathbf{x}_n), n = 0, 1, \cdots, N$

for $n = 0, 1, 2, \cdots N$ **do**
- Generate candidate $\mathbf{x}^* \sim q(\mathbf{x}; \mathbf{x}_n)$
- Determine the acceptance ratio $\alpha(\mathbf{x}^*; \mathbf{x}_n) = \min\left\{1, \frac{\pi(\mathbf{x}^*)q(\mathbf{x}_n; \mathbf{x}^*)}{\pi(\mathbf{x}_n)q(\mathbf{x}^*; \mathbf{x}_n)}\right\}$
- **if** $u \leq \alpha(\mathbf{x}^*; \mathbf{x}_n)$ *with* $u \sim U(0, 1)$ **then**
  - $\mathbf{x}_{n+1} = \mathbf{x}^*$
- **else**
  - $\mathbf{x}_{n+1} = \mathbf{x}_n$
- Adapt the mean and covariance of the proposal distribution $q(\mathbf{x}; \mathbf{x}_n)$

return Sequence of samples $(\mathbf{x}_n), n = 0, 1, \cdots, N$

In Algorithm 1, the dynamic adaptation facilitates better exploration of the target distribution and more efficient sampling. Unlike fixed proposal distributions, adaptive MCMC algorithms excel at handling diverse target distributions, leading to the improved performance. A pivotal element of adaptive MCMC schemes lies in their ability to adapt the mean and covariance of the proposal distribution in real-time, guided by the accepted samples.

Despite the advantages of adaptive MCMC in tuning the proposal distribution, it can compromise the convergence due to its dependence on the accepted samples. In order for the convergence to be maintained, the adaptive proposal must incorporate Stopped Adaptation (SA) and Diminishing Adaptation(DA), which will be discussed later in Section 4.

3 Adaptation Schemes based on stochastic optimization

As previously highlighted, the methodologies behind the adaptation mechanisms in AM, MGaA, and MCMA vary considerably. AM updates the covariance by taking into account the entire history of sample sets. In contrast, MGaA modifies the covariance aiming to optimize the entropy of the search distribution. Meanwhile, MCMA refines the covariance with an objective to enhance the probability of identifying favorable samples in future iterations.

In this section, we will provide a deeper examination of the nuances that distinguish these three adaptation strategies. Furthermore, given that the updates to $\mathbf{C}_n$ in both the AM and MGaA algorithms entail a considerably high computational overhead, we suggest a more resource-efficient method to update $\mathbf{C}_n$, leveraging the Cholesky decomposition.

3.1 Adaptive Metropolis Sampling

AM is the first example of adaptive MCMC, which dynamically updates the proposal distribution using all previous information gathered so far in the sampling process [6]. It determines whether to accept or reject a proposed sample by evaluating the acceptance probability $\alpha(\mathbf{x}^*; \mathbf{x}_n)$. One of the key features of the AM algorithm is its ability to dynamically adjust the shape of the proposal distribution to improve the exploration of the target distribution. This adaptive nature sets it apart from the traditional MH algorithm, which employs a fixed proposal distribution. As a result, the AM algorithm offers significant advantages in terms of sampling performance.

The AM sampling proceeds as follows: The candidate $\mathbf{x}^*$ is sampled from the proposal distribution $\mathcal{N}(\mathbf{x}_n, \mathbf{\Sigma}_n)$, then We set $\mathbf{x}_{n+1} = \mathbf{x}^*$ if the acceptance probability $\alpha(\mathbf{x}^*; \mathbf{x}_n)$ in Eq(1) is met, otherwise $\mathbf{x}_{n+1} = \mathbf{x}_n$. The proposal distribution is centered around the current state of the Markov chain, denoted as $\mathbf{x}_n$, and the covariance matrix is determined as:

$$\mathbf{\Sigma}_n = \sigma_n \cdot \text{cov}(\mathbf{x}_1, \cdots, \mathbf{x}_n) + \sigma_n \cdot \epsilon \cdot I_d$$

Here, σ_n is the scale parameter, $\epsilon = 0.05$ is a very small constant, the rationale of using ϵ is to avoid the algorithm getting stuck with a singular covariance matrix. And $I_d \in \mathbb{R}^{d \times d}$ represents the d-dimensional identity matrix [7]. The empirical covariance matrix $\text{cov}(\mathbf{x}_1, \cdots, \mathbf{x}_n)$ is computed as:

$$\text{cov}(\mathbf{x}_1, \cdots, \mathbf{x}_n) = \frac{1}{n}\left(\sum_{i=0}^{n} \mathbf{x}_i \mathbf{x}_i^{\mathrm{T}} - (n+1)\bar{\mathbf{x}}_n \bar{\mathbf{x}}_n^{\mathrm{T}}\right)$$

To simplify the notation, let $\mathbf{C}_n = \text{cov}(\mathbf{x}_1, \cdots, \mathbf{x}_n)$. Then, in order to save the computational cost, the covariance matrix is updated as the following recursion formula:

$$\mathbf{C}_{n+1} = \frac{n-1}{n}\mathbf{C}_n + \frac{1}{n+1}\left(\mathbf{x}_{n+1} - \overline{\mathbf{x}}_n\right)\left(\mathbf{x}_{n+1} - \overline{\mathbf{x}}_n\right)^T \tag{2}$$

Here, $\overline{\mathbf{x}}_n = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ represents the sample mean of all accepted samples. The scale parameter σ_n is always set to the optimal scale σ_{opt}.

The AM update involves calculating the empirical covariance of all the preceding samples. This update limits the capability of the proposal distribution to explore the space, particularly when dealing with complex distributions such as multimodal ones. Consequently, stochastic optimization methods like GaA and CMAES have been introduced to adapt the proposal distribution and augment its exploration capacity.

3.2 Metropolis Gaussian Adaptation Sampling

The idea of incorporating stochastic optimization into the concept of adaptive MCMC was originally introduced by Muller and Sbalzarini [13], where GaA was initially employed as the stochastic optimization method to fine-tune the proposal distribution. GaA is a stochastic optimization algorithm specifically tailored to handle noisy and discontinuous objective functions, particularly in situations where gradients or higher-order derivatives are not readily available [14]. GaA explores the search space by sampling potential solutions from a Gaussian distribution and iteratively updating the distribution's first and second moments. The adaptation strategy embedded in GaA aligns effectively with the adjustment process of the proposal distribution in adaptive MCMC. As a result, the direct use of GaA to refine the proposal distribution constitutes a straightforward approach, often referred to as MGaA sampling.

In MGaA sampling, the candidate $\mathbf{x}^*$ is sampled from the proposal distribution $\mathcal{N}(\mathbf{x}_n, \mathbf{\Sigma}_n)$, then accept or reject $\mathbf{x}^*$ by the probability $\alpha(\mathbf{x}^*; \mathbf{x}_n)$ in Eq(1). As for the adaptation of the covariance matrix $\mathbf{\Sigma}_n = \sigma_n^2\mathbf{C}_n$, when the candidate $\mathbf{x}^*$ is accepted, the scale σ_n is increased as $\sigma_{n+1} = f_e\sigma_n$, where $f_e > 1$ is the expansion factor. In case $\mathbf{x}^*$ is rejected, σ_n is reduced as $\sigma_{n+1} = f_c\sigma_n$, where $f_c < 1$ is the contraction factor. $\mathbf{C}_n$ is only updated when $\mathbf{x}^*$ is accepted:

$$\mathbf{C}_{n+1} = (1-\lambda_C)\mathbf{C}_n + \lambda_C(\mathbf{x}_{n+1} - \mathbf{x}_n)(\mathbf{x}_{n+1} - \mathbf{x}_n)^T \tag{3}$$

where $\lambda_C = \ln(d+1)/(d+1)^2$ weighs the influence of the accepted sample on the covariance adaptation, and d is the dimension of the state space. Table 1 gives the values of the parameters in MGaA, which are recommended in the reference [13].

3.3 A Trick for Updating the Covariance

In both the AM and MGaA algorithms, updating the complete $d \times d$ covariance matrix $\mathbf{C}_n$ involves a computational complexity of $\Omega(d^2)$. Furthermore, generating the $\mathbf{x}^*$ entails a covariance decomposition with cubic time complexity.

Table 1: MGaA Parameters

Name	Definition
Target acceptance rate	$\alpha^* = e^{-1}$
Learning rate of covariance matrix	$\lambda_C = ln(d+1)(d+1)^{-2}$
Scale expansion factor	$f_e = 1 + \lambda_C(1-\alpha^*)$
Scale contraction factor	$f_c = 1 - \lambda_C\alpha^*$

To mitigate these inefficiencies, we introduce a more efficient update approach for $\mathbf{C}_n$ based on the Cholesky decomposition. The Cholesky decomposition has proven effective in the context of CMAES [11], and we will adopt it for MCMA sampling as well.

Consider a d-dimensional Gaussian distribution $\mathcal{N}(\mathbf{x}_n, \mathbf{\Sigma}_n)$ as the proposal distribution, where $\mathbf{x}_n$ is the current state. The covariance matrix $\mathbf{\Sigma}_n$ is factorized as $\mathbf{\Sigma}_n = \sigma_n^2\mathbf{C}_n$. Because of the positive definiteness of $\mathbf{C}_n$, there exists an unique Cholesky decomposition $\mathbf{C}_n = \mathbf{L}_n\mathbf{L}_n^\top$, where $\mathbf{L}_n$ is a lower triangular matrix. The candidate $\mathbf{x}^*$ sampling from $\mathcal{N}(\mathbf{x}_n, \mathbf{\Sigma}_n)$ is equivalent to:

$$\mathbf{x}^* = \mathbf{x}_n + \sigma_n\mathbf{L}_n\mathbf{z}_n \tag{4}$$

where $\mathbf{z}_n \sim \mathcal{N}(0, \mathbf{I}_d)$, $\mathbf{I}_d$ is the d-dimensional identical matrix.

When updating the covariance, we choose to update $\mathbf{L}_n$ instead of directly updating $\mathbf{C}_n$. This approach offers several advantages. Firstly, updating $\mathbf{L}_n$ reduces the computational cost since we only need to update $d(d+1)/2$ parameters, compared to updating all d^2 parameters in $\mathbf{C}_n$. Secondly, when sampling the candidate using Eq (4), we can avoid the costly covariance decomposition when directly sampling from $\mathcal{N}(\mathbf{x}_n, \mathbf{\Sigma}_n)$. Additionally, sampling using $\mathbf{L}_n$ ensures the positive definiteness of $\mathbf{\Sigma}_n$. The updating of $\mathbf{L}_n$ is implemented by the efficient rank-one-update method. If $\mathbf{C}_n$ is updated as $\mathbf{C}_{n+1} = \alpha\mathbf{C}_n + \beta\mathbf{v}\mathbf{v}^T$, where $\mathbf{v}$ is a d-dimensional column vector, then the update rule for $\mathbf{L}_n$ is based on the *rank-one-update* method:

$$\mathbf{L}_{n+1} = \textit{rank-one-update}(\mathbf{L}_n, \alpha, \beta, \mathbf{v}) \tag{5}$$

The rank-one-update can guarantee that $\mathbf{L}_{n+1}\mathbf{L}_{n+1}^\top = \mathbf{C}_{n+1}$. For a more in-depth understanding of the rank-one update technique, please refer to the work by Krause et al. [11].

3.4 Metropolis Covariance Matrix Adaptation Evolution Strategy

In contrast to GaA, CMAES represents a more potent stochastic optimization algorithm that combines evolutionary strategies with adaptive Gaussian adaptation. It particularly excels in addressing optimization challenges within continuous domains. CMAES is designed to enhance the likelihood of sampling improved candidate solutions by maintaining a population of candidate solutions and adjusting the mean and covariance matrix of a multivariate Gaussian

distribution based on their performance. This adaptive methodology empowers CMAES to effectively navigate and exploit the search space, resulting in efficient and resilient optimization. Utilizing CMAES for fine-tuning the proposal distribution has been proposed [12], we call it MCMA in this paper.

There are different strategies for CMAES, we only consider the (1+1)-CMAES in this paper. The core goal of CMAES is optimization, with its focus squarely on pinpointing the apex of the objective function. Conversely, MCMA sampling serves to produce samples drawn from the target distribution, operating as an adaptive variant within the MCMC methodologies. Both methods involve generating candidates from the proposal distribution, but the key difference lies in the acceptance criterion for these candidates. In CMAES, the acceptance of a candidate is based on the value of the objective function. In contrast, MCMA sampling uses the acceptance probability $\alpha(\mathbf{x}^*; \mathbf{x}_n)$ to determine the acceptance of a candidate.

In MCMA sampling, the scale σ_n is adapted at each iteration in two steps

$$\begin{cases} \bar{p}_{succ} = (1 - \lambda_\sigma)\bar{p}_{succ} + \lambda_\sigma \alpha_p \\ \sigma_{n+1} = \sigma_n \exp\left(\frac{1}{k_\sigma}\left(\frac{\bar{p}_{succ} - \alpha^*}{1 - \alpha^*}\right)\right) \end{cases} \tag{6}$$

Here, $\alpha_p = 1$ if the candidate $\mathbf{x}^*$ is accepted, otherwise $\alpha_p = 0$, λ_σ is the acceptance rate averaging parameter, α^* is the target acceptance rate and k_σ is the scale damping parameter of σ_n. Besides, $\mathbf{C}_n$ is only adapted when $\mathbf{x}^*$ is accepted. The evolution path $\mathbf{p}_{c,n}$ is crucial in the covariance update process. The update also depends on the comparison between the average success rate $\bar{p}_{succ}$ and the threshold p_{thresh}, where $p_{thresh} < 0.5$. For the default parameters, we recommend referring to [10], where the suggested values are listed in Table 2.

Table 2: MCMA Parameters

Name	Definition
Scale damping	$k_\sigma = 1 + d/2$
Target acceptance rate	$\alpha^* = 2/11$
Acceptance rate averaging	$\lambda_\sigma = 1/12$
Learning rate of evolution path	$\lambda_p = 2/(d+2)$
Learning rate of covariance matrix	$\lambda_C = 2/(d^2+6)$
Acceptance threshold	$p_{thresh} = 0.44$

The entire process of MCMA sampling is presented in Algorithm 2. In the Algorithm 2, the covariance adaptation aims to strike a balance between exploration and acceptance rate. This is achieved by monitoring the relationship between the average success rate $\bar{p}_{succ}$ and the threshold p_{thresh}. When $\bar{p}_{succ}$ exceeds the threshold, it indicates a high acceptance rate, which implies less exploration. In such cases, the influence of the evolution path $\mathbf{p}_{c,n}$ needs to be minimized. Thus, the covariance is updated with a shorter evolution path.

Algorithm 2: MCMA Sampling

Input: Target $\pi(\mathbf{x})$, Initial state $\mathbf{x}_0$, σ_0, $\mathbf{L}_0$
Output: Sequence of generated samples $(\mathbf{x}_n), n = 0, 1, \cdots, N$

for $n = 0, 1, 2, \cdots N$ **do**
- Generate candidate $\mathbf{x}^* = \mathbf{x}_n + \sigma_n L_n \mathbf{z}_n$
- Determine the acceptance ratio $\alpha(\mathbf{x}^*; \mathbf{x}_n) = \min\left\{1, \frac{\pi(\mathbf{x}^*)q(\mathbf{x}_n; \mathbf{x}^*)}{\pi(\mathbf{x}_n)q(\mathbf{x}^*; \mathbf{x}_n)}\right\}$
- **if** $u \leq \alpha(\mathbf{x}^*; \mathbf{x}_n)$ *with* $u \sim U(0, 1)$ **then**
  - $\mathbf{x}_{n+1} = \mathbf{x}^*$
  - $\bar{p}_{succ} = (1 - \lambda_\sigma)\bar{p}_{succ} + \lambda_\sigma \alpha_p$
  - **if** $\bar{p}_{succ} < p_{thresh}$ **then**
    - $\mathbf{p}_{c,n+1} = (1 - \lambda_p)\mathbf{p}_{c,n} + \sqrt{\lambda_p(2 - \lambda_p)} L_n \mathbf{z}_n$
    - $\mathbf{L}_{n+1} =$ *rank-one-update*$(\mathbf{L}_n, 1 - \lambda_C, \lambda_p, \mathbf{p}_{c,n})$
  - **else**
    - $\mathbf{p}_{c,n+1} = (1 - \lambda_p)\mathbf{p}_{c,n}$
    - $\mathbf{L}_{n+1} =$ *rank-one-update*$(\mathbf{L}_n, 1 + \lambda_C(\lambda_p(2 - \lambda_p) - 1), \lambda_C, \mathbf{p}_{c,n})$
- **else**
  - $\mathbf{x}_{n+1} = \mathbf{x}_n$
  - $\bar{p}_{succ} = (1 - \lambda_\sigma)\bar{p}_{succ}$
- $\sigma_{n+1} = \sigma_n \exp\left(\frac{1}{k_\sigma}\left(\frac{\bar{p}_{succ} - \alpha^*}{1 - \alpha^*}\right)\right)$

return Sequence of samples $(\mathbf{x}_n), n = 0, 1, \cdots, N$

When the average acceptance rate $\bar{p}_{succ}$ is below the threshold, it suggests that the acceptance rate is lower than desired, indicating the need for further exploration. In this situation, we elongate the evolution path to encourage continued sampling in the same direction. By stretching the evolution path, we aim to maintain a consistent exploration pattern, allowing the algorithm to explore the search space more effectively.

4 Stopped Adaptation and Diminishing Adaptation

In adaptive MCMC, the inherent adaptation process can autonomously refine the proposal distribution, thereby elevating its capacity to navigate the state space and bolstering overall efficiency. While adaptation offers numerous advantages, it can also jeopardize the stationarity of the target distribution by compromising ergodicity. To uphold the stationary distribution in adaptive MCMC, many theories has been proposed [1, 16, 2]. In a word, these theories can classified into two categories: Stopped Adaptation (SA) and Diminishing Adaptation(DA).

The SA approach entails discontinuing the modification of the proposal distribution once specific conditions are met. This prevents excessive adjustments that might introduce biases or impede convergence. By SA, the MCMC sampler can traverse the target distribution without further modifications to the

proposal, thus ensuring that generated samples remain true representatives of the target distribution and preserve the ergodic nature of the Markov chain. SA could be guided by factors like predefined iteration limits, indicators of convergence, or predefined quality criteria for the samples.

On the other hand, the DA strategy involves gradually reducing the extent of adaptation over iterations or time. As the sampler progresses and the chain converges, the rate of adaptation is gradually reduced. This prevents the proposal distribution from becoming too focused on the current state, thereby facilitating effective exploration of diverse regions within the target distribution. This strategy achieves a delicate balance between thorough exploration and accurate sampling, which is especially pivotal when dealing with intricate or high-dimensional distributions.

There are three adaptation schemes mentioned in this paper: AM, MGaA, and MCMA. In the AM algorithm, DA is already taken into account. In Eq(2), the coefficient $\frac{1}{n+1}$ in the second term on the right side of the equation ensures the DA of covariance. However, in the MGaA and MCMA algorithms, the covariance matrix's learning rate λ_C remains constant, leading to unaltered adaptation of the proposal distribution throughout all iterations. To maintain stationarity in MGaA and MCMA sampling, we propose introducing DA into these two sampling methods. This can be achieved by introducing a damping factor γ_n to the learning rate, denoted as $\gamma_n \lambda_C$. Furthermore, we put three different damping factors: fast ($\gamma_n = n^{-1}$), medium ($\gamma_n = n^{-1/2}$), and slow ($\gamma_n = n^{-1/4}$). In addition to DA, this paper also considers the SA in MGaA and MCMA sampling.

5 Experiments

5.1 Test Suite

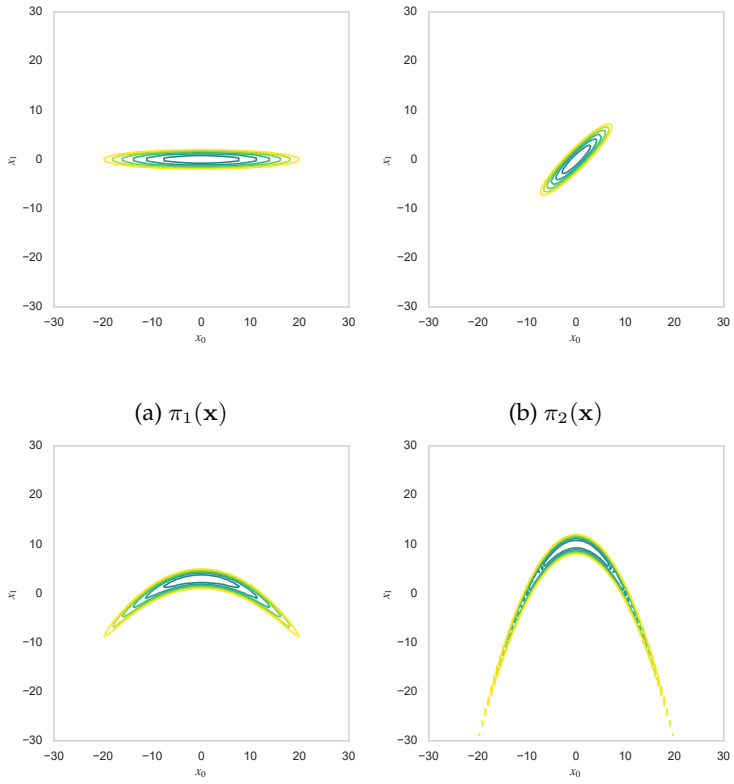
By utilizing these well-established test suites, we were able to assess the performance and effectiveness of our algorithms across a range of challenging distributions. These test suites provided a comprehensive evaluation framework for comparing and analyzing the behavior of our proposed methods. In this paper, we focused on the distributions proposed by Haario et al. [8]. These distributions are multivariate Gaussian distributions with varying covariance matrices, the summary of the Haario's distributions are shown in table 3. Besides, to more effectively illustrate Haario's distributions, the projections of first two dimensions are depicted in Figure 1.

5.2 Performance Metrics

To assess the reliability of the estimates, we employ several performance metrics: the Markov chain standard error (se_{MC}), the relative effective sample size (R_{ess}), the distance between the true mean and the sample mean (d_{coords}), and the run time (T_{run}).

Table 3: Summary of target distributions

Target	Description	Equation
π_1	Uncorrelated Gaussian	$\pi_1(\mathbf{x}) = \mathcal{N}(\mathbf{0}, C_u)$ where C_u is a diagonal matrix
π_2	Correlated Gaussian	$\pi_2(\mathbf{x}) = \mathcal{N}(\mathbf{0}, C_c)$ where C_c is the Householder transformation of C_u
π_3	Moderately twisted Gaussian	$\pi_3(\mathbf{x}) = \pi_1(\phi_b(\mathbf{x}))$ where $\phi_b(\mathbf{x}) = (x_1, x_2 + b(x_1 - 100), x_3, \ldots, x_d)$ and $b = 0.03$
π_4	Highly twisted Gaussian	$\pi_4(\mathbf{x}) = \pi_1(\phi_b(\mathbf{x}))$ where $\phi_b(\mathbf{x}) = (x_1, x_2 + b(x_1 - 100), x_3, \ldots, x_d)$ and $b = 0.1$



(a) $\pi_1(\mathbf{x})$ (b) $\pi_2(\mathbf{x})$

(c) $\pi_3(\mathbf{x})$ (d) $\pi_4(\mathbf{x})$

Fig. 1: The Haario's target distributions

1. Gelman Rubin's $\hat{R}$
   The Gelman-Rubin test ($\hat{R}$) is a test of samples generated from more than

one chain. It compares the difference within and between the chains. It is expected that once the chains have converged, the value should be close 1.

2. The Relative Effective Sample Size (R_{ess})
   R_{ess} is a measure of how equivalent a non-*i.i.d* samplers can be in relation to *i.i.d* effective samples. This translates to how much more effort do we need to put to a non *i.i.d* sampler so as to be as effective as the *i.i.d*. If the R_{ess} of a parameter is small, then the estimate of the posterior distribution of that parameter will be poor. R_{ess} indicates the ratio of the good samples to the overall number of generated samples. It is a relative measure to how an *i.i.d* would perform. A high R_{ess} value indicates a good sampler.
3. Markov chain Standard Error (se_{MC} and d_{Tot})
   The se_{MC} is used to measure of the variance among the chains for the different runs. The lower the se_{MC} the better the sampler.
4. Distance from the true mean (d_{coords})
   The distance from true mean measures from different runs and different coordinates. d_{coords} indicates the distances for the individual coordinates whereas d_{Tot} gives the average distance for all the coordinates.
5. Run time (T_{run})
   The run time is determined in terms of the amount of time required to complete an experiment. This measure is specifically important in comparing the efficiency of different samplers.

Table 4: Performance Metrics

Name	Value
Total distance	$d_{tot} = \|\boldsymbol{\mu} - \bar{\mathbf{x}}\|$
Distance per coordinate	$d_{coord} = \|\mu_i - \bar{x}_i\|$ for $i = 1, \cdots, d$
Relative effective sample size	$R_{ess} = N_{eff}/N$
Markov chain standard error	$\hat{se}_{mc} = \sqrt{1/N_{eff} \sum_{i=n}^{N} (\mathbf{x}_i - \bar{\mathbf{x}})^2}$
Gelman-Rubin	$\hat{R}$

The performance metrics are given in Table 4. The metrics d_{tot} and $d_{coord}, i = 1, \cdots, d$, measure the distance between the true and sample mean of the target, either the total distance or coordinate wise. The metric $\hat{se}_{mc}$ gives the standard error in the coordinate wise distances. This error is (much) larger than for i.i.d. samples. The metric $R_{ess} = \frac{N_{eff}}{N}$, where N is the number of samples generated and N_{eff} is the number of effectively independent samples. The metric T_{run} is the average time needed to complete an experiment. Low values of d_{tot}, d_{coord}, $\hat{se}_{mc}$, and T_{run}, and high values of R_{ess} are preferred.

5.3 Experiment results

In our experiments, we initially compare the performance of MGaA(GaA) and its various configurations. These configurations include: MGaA with SA (GaAs), MGaA with three distinct rates of DA (GaAd1, GaAd2, GaAd3). Additionally, we maintain the same configurations for MCMA as follows: MCMA(CMAES) and its variants (CMAESs, CMAESd1, CMAESd2, CMAESd3).

The three different rates of diminishing adaptation are defined as follows: fast ($\gamma_n = n^{-1}$), medium ($\gamma_n = n^{-1/2}$), and slow ($\gamma_n = n^{-1/4}$). These rates are used to evaluate the performance of the MGaA variants and MCMA variants and determine the most effective configurations for MGaA and MCMA. In order to avoid the random error, we run each experiment independently for 10 times. Then we compare the mean and variance of the performance metrics mentioned in Table 4.

The Gelman-Rubin statistic ($\hat{R}$) for all experiments was close to 1, indicating the convergence of the Markov chains. Besides, the experiment results on π_1 and π_2 are similar, all samplers performance well due to the simplicity of target distributions. As for the twisted Gaussian π_3 and π_4, the results are similar for both targets. The results for the highly twisted Gaussian π_4 of dimension $d = 25$ are shown in Fig 2 and Fig 3.

In Fig 2 and Fig 3, we plot the error bars and 95% confidence intervals of each metrics. In terms of d_{coord}, R_{ess}, $\hat{se}_{mc}$, and d_{tot}, medium and slow diminishing work best for MGaA and MCMA, respectively. Both do slightly better than stopped adaptation. As expected, in terms of T_{run} all performances are similar.

Besides, we compared these variants with AM and MH using the optimal proposal, the experiment results are shown in Fig 4. AM and MH are slightly better than CMAESd3(best variant of MCMA) and both are significantly better than GaAd2 (best variant of MGaA) with respect to all metrics. The reason is the Gaussian target, AM and MH use the optimal proposal.

6 Conclusion and Future Work

Adaptive MCMC, overcomes the challenges faced by traditional MCMC by enabling the proposal distribution to learn online. We have done a comparison of different adaptive schemes of MCMC samplers. We first look at different variants of both MCMA and MGaA. These variant are similar in design to their respective standard algorithms, except that the learning(adaptation) is either stopped or reduced as the sampling progressed. This technique is important to maintain Markovian property. We implemented three different rates for diminishing adaptation factors i.e fastest adaptation $\frac{1}{n}$, medium adaptation $\frac{1}{\sqrt{n}}$ and slow adaptation $\frac{1}{\sqrt[4]{n}}$.

We have only shown the performances on target π_4 for $d = 25$. But in most cases the performances on the other targets and for all dimensions d considered are similar. We conclude that all samplers converge according to the $\hat{R}$-test. The

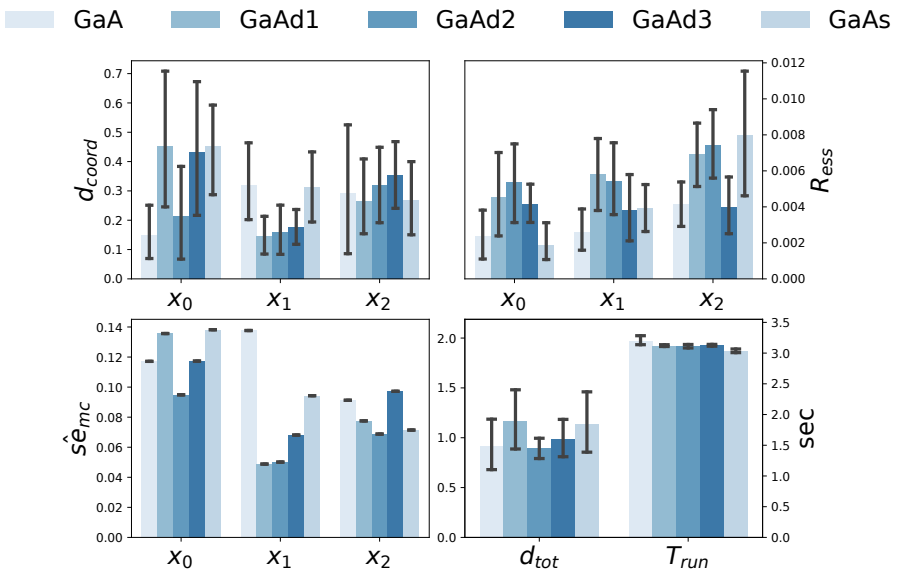


Fig. 2: Metrics d_{coord} , R_{ess}, $\hat{se}_{mc}$, and T_{run} for MGaA and its invariants

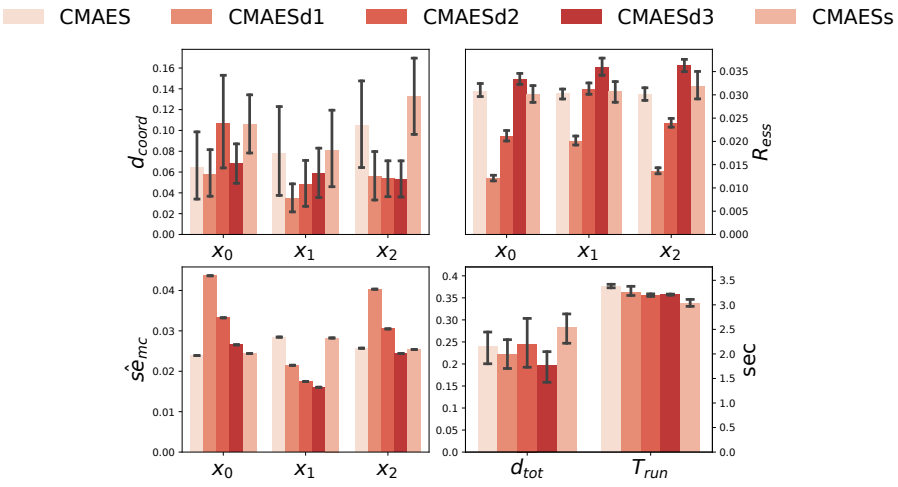


Fig. 3: Metrics d_{coord} , R_{ess}, $\hat{se}_{mc}$, and T_{run} for for MCMA and its invariants

adaptation schemes used in AM and CMAESd3 give similar performance although they are very different, both better than MGaA and its variants. However, when the target becomes more complex, MCMA will perform better than AM and MH.

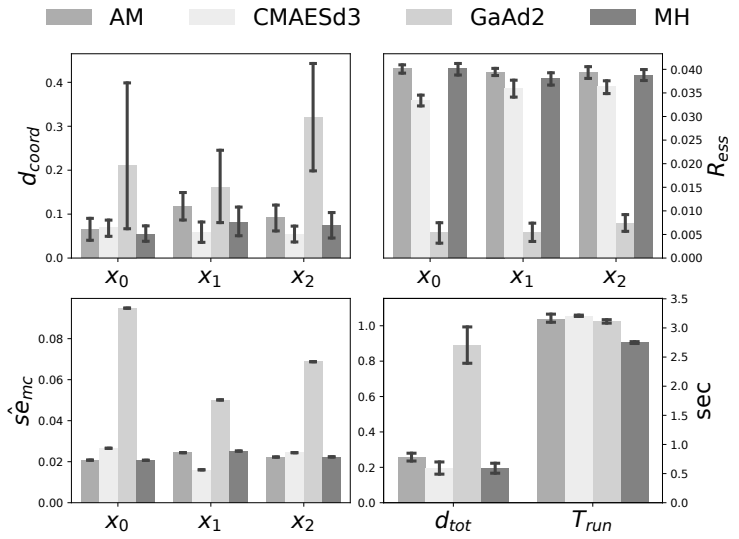


Fig. 4: Metrics d_{coord} , R_{ess}, $\hat{se}_{mc}$, and T_{run} for *AM*, *M-CMA3*, *M-GaA2* and *MH* on π_4. Note that *MH* uses the optimal proposal covariance

MCMA Sampling exhibits strong adaptability to complex distributions, utilizing recombination and mutation to introduce variations in sample placement. The results obtained from MCMA Sampling are highly promising, indicating that adaptation significantly enhances sampler effectiveness.

It was observed that MCMA with the highest diminishing factor outperforms the original MCMA as well as other samplers. The rate at which adaptation occurs is influenced by the diminishing factor. Adaptive samplers that halt adaptation or increase the diminishing factor outperform those with continuous adaptation or slow diminish.

References

1. Andrieu, C., Moulines, É.: On the ergodicity properties of some adaptive MCMC algorithms. The Annals of Applied Probability **16**(3), 1462–1505 (2006)
2. Andrieu, C., Thoms, J.: A tutorial on adaptive MCMC. Statistics and Computing **4**(18), 343–373 (2008)
3. Dunson, D.B., Johndrow, J.E.: The Hastings algorithm at fifty. Biometrika **107**(1), 1–23 (2020)
4. Gelman, A., Roberts, G.O., Gilks, W.R., et al.: Efficient Metropolis jumping rules. Bayesian statistics **5**, 599–607 (1996)
5. Gelman, A., Rubin, D.B.: Inference from iterative simulation using multiple sequences. Statistical science **7**(4), 457–472 (1992)
6. Haario, H., Saksman, E., Tamminen, J.: An adaptive Metropolis algorithm. Bernoulli **7**(2), 223–242 (2001)

7. Haario, H., Laine, M., Mira, A., Saksman, E.: DRAM: efficient adaptive MCMC. Statistics and Computing **16**(4), 339–354 (2006)
8. Haario, H., Saksman, E., Tamminen, J.: Adaptive proposal distribution for random walk Metropolis algorithm. Computational Statistics **14**(3), 375–395 (1999)
9. Huelsenbeck, J.P., Larget, B., Miller, R.E., Ronquist, F.: Potential applications and pitfalls of Bayesian inference of phylogeny. Systematic biology **51**(5), 673–688 (2002)
10. Igel, C., Suttorp, T., Hansen, N.: A computational efficient covariance matrix update and a (1+ 1)-CMA for evolution strategies. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. pp. 453–460 (2006)
11. Krause, O., Igel, C.: A more efficient rank-one covariance matrix update for evolution strategies. In: Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII. pp. 129–136 (2015)
12. Liu, B., Milgo, E., Ronoh, N., Bernard, M.: Comparison of MCMC adaptation schemes: A preliminary empirical study. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. pp. 303–306 (2023)
13. Müller, C.L., Sbalzarini, I.F.: Gaussian adaptation as a unifying framework for continuous black-box optimization and adaptive Monte Carlo sampling. In: IEEE Congress on Evolutionary Computation. pp. 1–8 (2010)
14. Müller, C.L., Sbalzarini, I.F.: Gaussian adaptation revisited: An entropic view on covariance matrix adaptation. Applications of evolutionary computation **6024**, 432–441 (2010)
15. Qian, S.S., Stow, C.A., Borsuk, M.E.: On Monte Carlo methods for Bayesian inference. Ecological modelling **159**(2-3), 269–277 (2003)
16. Roberts, G.O., Rosenthal, J.S.: Coupling and ergodicity of adaptive Markov Chain Monte Carlo algorithms. Journal of applied probability **44**(2), 458–475 (2007)
17. Roberts, G.O., Rosenthal, J.S.: Examples of adaptive MCMC. Journal of computational and graphical statistics **18**(2), 349–367 (2009)