ENHANCED MIXTURE POPULATION MONTE CARLO VIA STOCHASTIC OPTIMIZATION AND MARKOV CHAIN MONTE CARLO SAMPLING

Yousef El-Laham   *Petar M. Djurić*   *Mónica F. Bugallo*

Department of Electrical and Computer Engineering
Stony Brook University, Stony Brook, NY 11794-2350
{yousef.ellaham, petar.djuric, monica.bugallo}@stonybrook.edu

ABSTRACT

The population Monte Carlo (PMC) algorithm is a popular adaptive importance sampling (AIS) method used for approximate computation of intractable integrals. Over the years, many advances have been made in the theory and implementation of PMC schemes. The mixture PMC (M-PMC) algorithm, for instance, optimizes the parameters of a mixture proposal distribution in a way that minimizes that Kullback-Leibler divergence to the target distribution. The parameters in M-PMC are updated using a single step of expectation maximization (EM), which limits its accuracy. In this work, we introduce a novel M-PMC algorithm that optimizes the parameters of a mixture proposal distribution, where parameter updates are resolved via stochastic optimization instead of EM. The stochastic gradients w.r.t. each of the mixture parameters are approximated using a population of Markov chain Monte Carlo samplers. We validate the proposed scheme via numerical simulations on an example where the considered target distribution is multimodal.

***Index Terms*—** population Monte Carlo, Bayesian inference, stochastic optimization, MCMC sampling, Rényi divergence

1. INTRODUCTION

For the past several decades, Monte Carlo (MC) methods have been the key tools for approximate Bayesian inference [1, 2, 3]. In Bayesian signal processing, the goal is to perform numerical integration, where the integrals of interest are expectations with respect to the posterior probability distribution of the set of unknowns in a probabilistic model (also called the *target distribution*) . Since the inception of the Markov chain Monte Carlo (MCMC) sampling algorithm [4], the main focus of the Bayesian community has been on the design of efficient and robust MC schemes for high-dimensional posterior inference [5, 6, 7]. Over the years, research efforts have resulted into a variety of adaptive and advanced MCMC sampling algorithms [8, 9], including the Hamiltonian MC (HMC) approach [10, 11]. The most appealing characteristic of MCMC sampling methods is that their estimates are guaranteed to converge to the true value as the number of iterations tends to infinity [12].

Another class of MC methods employed for Bayesian inference is *adaptive importance sampling* (AIS) [13]. Unlike MCMC sampling methods which approximate the posterior by constructing a Markov chain, AIS methods approximate the posterior using a set of weighted samples drawn from an alternative distribution called the *proposal distribution*. The weights are computed according to the ratio of the target and the proposal, and they allow for obtaining consistent estimates of desired integrals. At each iteration of an AIS scheme, the proposal is adapted in order to improve the sampling efficiency of the algorithm. In the ideal scenario, the adapted proposal should be as close as possible to the target, which would maximize the performance of the algorithm. While many AIS schemes have been proposed in the literature [14, 15, 16, 17, 18], there is still a need for the development of AIS algorithms that efficiently adapt complex proposals, such as mixture proposals.

This work proposes a novel AIS methodology that incorporates ideas from stochastic optimization and MCMC sampling. The proposed method provides a mechanism for fully adapting a mixture proposal distribution in a way that minimizes the Rényi divergence to the target. The proposal updates are resolved using a stochastic gradient step, where stochastic gradients are estimated from samples obtained from a set of MCMC chains. The MCMC chains are run in parallel and guarantee that each mxiand is well-represented in the computation of the stochastic gradients. The proposed scheme can be viewed as a more general implementation of the parallel interacting Markov AIS (PIMAIS) scheme proposed in [19] and the mixture population MC (MPMC) scheme proposed in [15].

2. PROBLEM FORMULATION

Consider an unknown vector $\mathbf{x} \in \mathbb{R}^{d_x}$. Suppose that we observe data $\mathcal{Y} \triangleq \{\mathbf{y}_n \in \mathbb{R}^{d_y}\}_{n=1}^N$ such that $\mathbf{y}_n \sim p(\mathbf{y}|\mathbf{x})$ for $n = 1, \ldots, N$ are conditionally independent and identically distributed. We are interested in the Bayesian estimation of the unknown $\mathbf{x}$. This amounts to obtaining the posterior distribution of $\mathbf{x}$ given the data $\mathcal{Y}$, which is expressed using Bayes' theorem as follows:

$$\pi(\mathbf{x}) \triangleq p(\mathbf{x}|\mathcal{Y}) = \frac{\ell(\mathcal{Y}|\mathbf{x})p(\mathbf{x})}{\int_{\mathbb{R}^{d_x}} \ell(\mathcal{Y}|\mathbf{x})p(\mathbf{x})d\mathbf{x}}, \quad (1)$$

where $\ell(\mathcal{Y}|\mathbf{x}) = \prod_{n=1}^N p(\mathbf{y}_n|\mathbf{x})$ is the *likelihood* of $\mathbf{x}$, $p(\mathbf{x})$ is the *prior* of $\mathbf{x}$, and $Z \triangleq \int_{\mathbb{R}^{d_x}} \ell(\mathcal{Y}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$ is a normalization constant commonly referred to as the *marginal likelihood*. Typically, Z is intractable and thus, one only has access to the *non-normalized posterior*, which is defined as the product of the likelihood and the prior, i.e., $\tilde{\pi}(\mathbf{x}) \triangleq \ell(\mathcal{Y}|\mathbf{x})p(\mathbf{x})$. Ultimately, the goal is to approximate posterior expectations, which in general take the following form:

$$\mathcal{H} = \int_{\mathbb{R}^{d_x}} h(\mathbf{x})\pi(\mathbf{x})d\mathbf{x}, \quad (2)$$

where $h(\mathbf{x})$ is a function assumed to be integrable w.r.t. $\pi(\mathbf{x})$.

The authors would like to thank the support of the NSF (CCF-1617986) and the NIH (RO1HD097188-01). The Stony Brook Research Computing and Cyberinfrastructure and the Institute for Advanced Computational Science at Stony Brook University allowed for access to the high-performance SeaWulf computing system, supported by NSF (award OAC-1531492).

3. PRELIMINARIES

3.1. MCMC Sampling

In MCMC sampling, the integral in (2) is estimated using a MC approximation, where the samples employed in the MC approximation are obtained by constructing a Markov chain whose stationary distribution is the target posterior. The most fundamental MCMC sampling method is the Metropolis-Hastings (MH) algorithm [5]. At each iteration of the MH algorithm, a sample $\mathbf{x}^*$ is proposed by propagating the previous state $\mathbf{x}_{i-1}$ using a Markov transition kernel $r(\mathbf{x}^*|\mathbf{x}_{i-1})$. The proposed sample is accepted as the next state in the Markov chain $\mathbf{x}_i$ with probability

$$\alpha_i = \min\left(1, \frac{\tilde{\pi}(\mathbf{x}^*)r(\mathbf{x}_{i-1}|\mathbf{x}^*)}{\tilde{\pi}(\mathbf{x}_{i-1})r(\mathbf{x}^*|\mathbf{x}_{i-1})}\right). \tag{3}$$

If the proposed sample is not accepted, the updated state takes the value of the previous state, i.e. $\mathbf{x}_i = \mathbf{x}_{i-1}$. States in the constructed chain $\mathcal{X} = \{\mathbf{x}_\tau\}_{\tau=I_0+1}^{I}$ are taken as samples from the target posterior, where I_0 is called the burn-in period and I is the total number of iterations.

The performance of an MH sampler depends on the choice of the Markov transition kernel. Adaptive MH algorithms have been proposed as a means to learn the transition kernel that optimizes the efficiency of the algorithm. For example, elements of stochastic optimization have been employed in the adaptive MCMC literature, where the optimal transition kernel is obtained using the Robbins-Monro stochastic approximation algorithm [20].

3.2. Adaptive Importance Sampling

IS methods estimate (2) using a batch of M samples $\{\mathbf{x}^{(m)}\}_{m=1}^{M}$ drawn from an alternative distribution $q(\mathbf{x};\boldsymbol{\theta})$ called the proposal, where $\boldsymbol{\theta}$ denotes the proposal parameters. The drawn samples are weighted according to the ratio of the target and proposal, i.e.,

$$\tilde{w}^{(m)} = \frac{\tilde{\pi}(\mathbf{x}^{(m)})}{q(\mathbf{x}^{(m)};\boldsymbol{\theta})}, \quad m = 1, \ldots, M. \tag{4}$$

The integral in (2) is approximated using the self-normalized IS estimator, which is given by:

$$\hat{\mathcal{H}}_{IS} = \sum_{m=1}^{M} \bar{w}^{(m)} h(\mathbf{x}^{(m)}), \tag{5}$$

where $w^{(m)} = \tilde{w}^{(m)} / \sum_{j=1}^{M} \tilde{w}^{(j)}$ is the normalized weight of the mth sample. It is well-known that (5) is a consistent estimator of (2). The performance of IS depends largely on the choice of the proposal $q(\mathbf{x};\boldsymbol{\theta})$. For instance, if the goal is to estimate the normalization constant Z, the proposal should be chosen in a way such that $q(\mathbf{x};\boldsymbol{\theta}) \propto \tilde{\pi}(\mathbf{x})$. Unfortunately, this is difficult to achieve in practice, especially when the dimension of the unknown vector $\mathbf{x}$ is large.

AIS methods are a family of IS algorithms that iteratively adapt the parameters of the proposal in order to obtain improved estimates of the target expectations. Many variants of AIS have been proposed in the literature. For example, one variant of AIS embeds MCMC sampling to adapt the location parameters of the proposal. There are also other AIS methods which adapt the proposal parameters using stochastic optimization, where stochastic gradients approximated using samples drawn from the proposal are used to adapt the proposal parameters [21, 22, 23, 24].

4. PROPOSED METHOD

In this section, we introduce a novel M-PMC sampler, called *controlled mixture population Monte Carlo* (CMPMC), that uses stochastic optimization to adapt the parameters of a mixture proposal. The stochastic gradients w.r.t. the proposal parameters are approximated using samples generated via a MCMC sampling method and then used in an instance of an off-the-shelf stochastic optimization algorithm to adapt the parameters of the proposal.

4.1. Rényi Divergence Minimization

The goal in AIS methods is to adapt the proposal distribution so that it is as close as possible to the target. In other words, we would like to determine the proposal parameters that solve the following minimization problem:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}\in\Theta}{\arg\min}\, \mathcal{D}_\alpha\left(\pi \| q_{\boldsymbol{\theta}}\right), \tag{6}$$

where Θ denotes the feasibility set of $\boldsymbol{\theta}$ and $\mathcal{D}_\alpha\left(\pi\|q_{\boldsymbol{\theta}}\right)$ denotes the Rényi divergence of order α whose expression is given by:

$$\mathcal{D}_\alpha\left(\pi\|q_{\boldsymbol{\theta}}\right) = \frac{1}{\alpha-1}\log\left(\int_{-\infty}^{\infty} \pi(\mathbf{x})^\alpha q(\mathbf{x};\boldsymbol{\theta})^{1-\alpha} d\mathbf{x}\right), \tag{7}$$

where $0 < \alpha < \infty$ and $\alpha \neq 1$. We remark that $\lim_{\alpha\to1} \mathcal{D}_\alpha(\pi\|q_{\boldsymbol{\theta}}) = \mathcal{D}_{\mathrm{KL}}(\pi\|q_{\boldsymbol{\theta}})$, where $\mathcal{D}_{\mathrm{KL}}(\pi\|q_{\boldsymbol{\theta}})$ is the Kullback-Leibler divergence (KLD) between $\pi(\mathbf{x})$ and $q(\mathbf{x};\boldsymbol{\theta})$. When $\alpha > 1$, minimizing (7) is equivalent to minimizing $C_\alpha(\boldsymbol{\theta}) \triangleq Z^\alpha \exp\left((\alpha-1)\mathcal{D}_\alpha\left(\pi\|q_{\boldsymbol{\theta}}\right)\right)$, since $Z > 0$ and for $\alpha > 1$, $f(z) = \exp((\alpha-1)z)$ is a monotonically increasing function in $z \in \mathbb{R}$ [22]. Taking this into account, we would like to compute the gradient of $C_\alpha(\boldsymbol{\theta})$ so that we can use a stochastic optimization algorithm to obtain the optimal proposal parameters as defined by (6). It is not difficult to see that $C_\alpha(\boldsymbol{\theta})$ can be expressed as the following:

$$C_\alpha(\boldsymbol{\theta}) = \mathbb{E}_\pi\left[\left(\frac{\tilde{\pi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right)^{\alpha-1}\right]. \tag{8}$$

Given this expression for $C_\alpha(\boldsymbol{\theta})$, we can now compute the gradient as follows:

$$\nabla_{\boldsymbol{\theta}} C_\alpha(\boldsymbol{\theta}) = \mathbb{E}_\pi\left[\nabla_{\boldsymbol{\theta}}\left(\left(\frac{\tilde{\pi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right)^{\alpha-1}\right)\right]. \tag{9}$$

We obtain a final expression for the gradient of $C_\alpha(\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$ as:

$$\nabla_{\boldsymbol{\theta}} C_\alpha(\boldsymbol{\theta}) = (1-\alpha)\mathbb{E}_\pi\left[\left(\frac{\tilde{\pi}(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right)^{\alpha-1} \frac{\nabla_{\boldsymbol{\theta}} q(\mathbf{x};\boldsymbol{\theta})}{q(\mathbf{x};\boldsymbol{\theta})}\right]. \tag{10}$$

4.2. Solving the Minimization Problem

In this work, we are interested in optimizing a proposal distribution that is a mixture distribution, which has the following form:

$$q(\mathbf{x};\boldsymbol{\theta}) = \sum_{d=1}^{D} \rho_d q_d(\mathbf{x};\boldsymbol{\theta}_d), \tag{11}$$

where ρ_d and $\boldsymbol{\theta}_d \in \Theta_d$ denote the weight and parameters of the dth mixand $q_d(\mathbf{x};\boldsymbol{\theta}_d)$ and D is the total number of mixands. The mixand weights satisfy $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_D]^\top \in \mathcal{S}$, where $\mathcal{S}$ denotes the probability simplex. We are interested in computing the gradients of $q(\mathbf{x};\boldsymbol{\theta})$ w.r.t. each of the proposal parameters $\boldsymbol{\theta} = [\rho_1, \dots, \rho_D, \boldsymbol{\theta}_1^\top, \dots, \boldsymbol{\theta}_D^\top]^\top$. We can compute the derivative of (11) w.r.t. ρ_j easily since the distribution is linear in the mixand weights, i.e.,

$$\nabla_{\rho_j} q(\mathbf{x};\boldsymbol{\theta}) = q_j(\mathbf{x};\boldsymbol{\theta}_j). \tag{12}$$

If $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(K)}$ represent a mini-batch of K samples from the target $\pi(\mathbf{x})$, we can obtain a stochastic approximation to $\nabla_{\rho_j} C_\alpha(\boldsymbol{\theta})$ using the following MC estimator:

$$g(\rho_j) = \frac{1-\alpha}{K} \sum_{k=1}^{K} \left(\frac{\tilde{\pi}(\mathbf{z}^{(k)})}{q(\mathbf{z}^{(k)};\boldsymbol{\theta})} \right)^{\alpha-1} \frac{q_j(\mathbf{z}^{(k)};\boldsymbol{\theta}_j)}{q(\mathbf{z}^{(k)};\boldsymbol{\theta})}. \tag{13}$$

To obtain the gradient w.r.t. $\boldsymbol{\theta}_j$, we can utilize the log-derivative to simplify the computation:

$$\begin{aligned} \nabla_{\boldsymbol{\theta}_j} q(\mathbf{x};\boldsymbol{\theta}) &= \rho_j \nabla_{\boldsymbol{\theta}_j} q_j(\mathbf{x};\boldsymbol{\theta}_j) \\ &= \rho_j q_j(\mathbf{x};\boldsymbol{\theta}_j) \nabla_{\boldsymbol{\theta}_j} \log q_j(\mathbf{x};\boldsymbol{\theta}_j). \end{aligned}$$

This yields the following MC estimator of the gradient:

$$g(\boldsymbol{\theta}_j) = \frac{\rho_j(1-\alpha)}{K} \sum_{k=1}^{K} \left(\frac{\tilde{\pi}(\mathbf{z}^{(k)})}{q(\mathbf{z}^{(k)};\boldsymbol{\theta})} \right)^{\alpha-1} \frac{q_j(\mathbf{z}^{(k)};\boldsymbol{\theta}_j)}{q(\mathbf{z}^{(k)};\boldsymbol{\theta})} u_j(\mathbf{z}^{(k)}, \boldsymbol{\theta}_j), \tag{14}$$

where $u_j(\mathbf{z}^{(k)}, \boldsymbol{\theta}_j) = \nabla_{\boldsymbol{\theta}_j} \log q_j(\mathbf{x};\boldsymbol{\theta}_j)\big|_{\mathbf{x}=\mathbf{z}^{(k)}}$. We adopt ideas from the adaptive MCMC literature [20] and use an MCMC method targeting $\pi(\mathbf{x})$ to obtain the mini-batch of samples necessary for the stochastic gradient computation in (13) and (14).

4.3. Controlled Mixture Population Monte Carlo (CMPMC)

The proposed CMPMC algorithm applies the ideas developed in Section 4.2 to adapt the parameters of a mixture proposal in AIS. The sampling and weighting steps of the proposed method are exactly the same as M-PMC sampling, however, the adaptation step is executed via the use of D MCMC chains that are run in parallel. The MCMC chains are used to compute stochastic gradients w.r.t. each of the mixtures parameters and an off-the-shelf stochastic optimization algorithm is applied to resolve the adaptation. One possible implementation of CMPMC is shown in Algorithm 1.

At the ith iteration of the CMPMC algorithm, M samples are drawn from the proposal $q(\mathbf{x};\boldsymbol{\theta}_i) = \sum_{d=1}^{D} \rho_{d,i} q_d(\mathbf{x};\boldsymbol{\theta}_{d,i})$. After weighting, we proceed to computing the stochastic gradient w.r.t. to the mixture weights and the mixand parameters: $g(\boldsymbol{\rho}_i) = [g(\rho_{1,i}), \dots, g(\rho_{D,i})]^\top, g(\boldsymbol{\theta}_{1,i}), \dots, g(\boldsymbol{\theta}_{D,i})$. For the dth mixand, a Markov chain $\mathcal{Z}_{d,i} = \{\mathbf{z}_{d,i}^{(k)}\}_{k=1}^{K}$ is constructed using an MCMC sampler targeting $\pi(\mathbf{x})$ with initial state $\mathbf{z}_{d,i}^{(0)}$. The samples that comprise this Markov chain are taken as samples from $\pi(\mathbf{x})$ and are used to compute $g(\rho_{d,i})$ and $g(\boldsymbol{\theta}_{d,i})$ via (13) and (14), respectively. The mixand parameters $\boldsymbol{\theta}_{d,i}$ can be updated via an instance of an algorithm like projected stochastic gradient descent (PSGD), i.e.,

$$\boldsymbol{\theta}_{d,i+1} = \Pi_{\Theta_d}\left(\boldsymbol{\theta}_{d,i} - \gamma_{d,i} g(\boldsymbol{\theta}_{d,i})\right), \quad d = 1, \dots, D, \tag{15}$$

where $\Pi_{\Theta_d}(\cdot)$ denotes the projection onto the feasible set Θ_d and $\gamma_{d,i}$ denotes the learning rate dth mixand's parameters at the ith iteration. The corresponding PSGD update for mixture weights $\boldsymbol{\rho}_i = [\boldsymbol{\rho}_{1,i}, \dots, \boldsymbol{\rho}_{D,i}]^\top$ is as follows:

$$\boldsymbol{\rho}_{i+1} = \Pi_{\mathcal{S}}\left(\boldsymbol{\rho}_i - \eta_i g(\boldsymbol{\rho}_i)\right), \tag{16}$$

where $\Pi_{\mathcal{S}}(\cdot)$ denotes the projection onto the probability simplex and η_i denotes the mixture weights' learning rate at the ith iteration.

Algorithm 1 Controlled Mixture PMC (CMPMC)

```
Initialization: Set $\rho_{d,1}, \boldsymbol{\theta}_{d,1}, \mathbf{z}_{d,1}^{(0)}$ for $d = 1, \dots, D$.
for $i = 1, \dots, I$ do
    Draw $M$ samples from the current proposal
        $$\mathbf{x}_i^{(m)} \sim \sum_{d=1}^{D} \rho_{d,i} q_d(\mathbf{x};\boldsymbol{\theta}_{d,i}), \quad m = 1, \dots, M.$$
    Compute the importance weights
        $$\tilde{w}_i^{(m)} = \frac{\tilde{\pi}(\mathbf{x}_i^{(m)})}{\sum_{d=1}^{D} \rho_{d,i} q_d(\mathbf{x}_i^{(m)};\boldsymbol{\theta}_{d,i})}, \quad m = 1, \dots, M.$$
    for $d = 1, \dots, D$ do in parallel
        1. Construct $\mathcal{Z}_{d,i} = \{\mathbf{z}_{d,i}^{(k)}\}_{k=1}^{K}$ using an MCMC sampler
           targeting $\pi(\mathbf{x})$, with initial state $\mathbf{z}_{d,i}^{(0)}$. Set $\mathbf{z}_{d,i+1}^{(0)} = \mathbf{z}_{d,i}^{(K)}$.
        2. Compute the stochastic gradients $g(\rho_{d,i})$ and $g(\boldsymbol{\theta}_{d,i})$ using
           $\mathcal{Z}_{d,i}$ as approximate samples from the target.
        3. Update the mixand parameter vector
               $$\boldsymbol{\theta}_{d,i+1} = \Pi_{\Theta_d}\left(\boldsymbol{\theta}_{d,i} - \gamma_{d,i} g(\boldsymbol{\theta}_{d,i})\right).$$
    end for
    Update the mixture weights
        $$\boldsymbol{\rho}_{i+1} = \Pi_{\mathcal{S}}\left(\boldsymbol{\rho}_i - \eta_i g(\boldsymbol{\rho}_i)\right).$$
end for
Output: Return $\mathcal{X}_i = \{\mathbf{x}_i^{(m)}, \tilde{w}_i^{(m)}\}_{m=1}^{M}$ for $i = 1, \dots, I$.
```

Any MCMC technique can be employed for the adaptation procedure in the proposed scheme. Here, we apply a simple Metropolis-Hastings sampler to construct the chains, where the transition kernel is chosen to be an isotropic Gaussian, i.e., $r(\mathbf{z}^*|\mathbf{z}_{d,i-1}) = \mathcal{N}(\mathbf{z}_d^*; \mathbf{z}_{d,i-1}, \zeta_r^2 \mathbb{I}_{d_x})$. For robustness to strongly autocorrelated Markov chains, we apply a thinning procedure, where only every ξth sample in the chain is used in the computation of the stochastic gradients. We also remark that the Markov chains can be warm-started before running CMPMC in order to obtain more accurate stochastic gradients in the earlier iterations of the algorithm.

Remark on computational complexity: Since MCMC methods are naturally sequential schemes, the main burden of the adaptation step in CMPMC is the generation of the Markov chains $\mathcal{Z}_{1,i}, \dots, \mathcal{Z}_{D,i}$ at each iteration. To lessen this burden, we remark that one can straightforwardly modify the CMPMC algorithm to allow for different mixands to share the same Markov chain.

4.3.1. *Mixture of Multivariate Gaussians*

Suppose that the mixture we want to optimize has the form $q(\mathbf{x};\boldsymbol{\theta}_i) = \sum_{d=1}^{D} \rho_d \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{d,i}, \boldsymbol{\Lambda}_{d,i})$, where $\boldsymbol{\mu}_{d,i}$ and $\boldsymbol{\Lambda}_{d,i}$ are the mean and precision matrix of the dth Gaussian mixand respectively. Given this choice of proposal, we can easily determine the gradients $\nabla_{\boldsymbol{\mu}_{j,i}} \log \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{j,i}, \boldsymbol{\Lambda}_{j,i})$ and $\nabla_{\Lambda_{j,i}} \log \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{j,i}, \boldsymbol{\Lambda}_{j,i})$ that are necessary for the computation of the parameter updates:

$$\begin{aligned} &\nabla_{\boldsymbol{\mu}_{j,i}} \log \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{j,i}, \boldsymbol{\Lambda}_{j,i}) = \boldsymbol{\Lambda}_{j,i}(\mathbf{x} - \boldsymbol{\mu}_{j,i}), \\ &\nabla_{\boldsymbol{\Lambda}_{j,i}} \log \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{j,i}, \boldsymbol{\Lambda}_{j,i}) = \tfrac{1}{2}\left(\boldsymbol{\Lambda}_{j,i}^{-1} - (\mathbf{x} - \boldsymbol{\mu}_{j,i})(\mathbf{x} - \boldsymbol{\mu}_{j,i})^\top\right). \end{aligned} \tag{17}$$

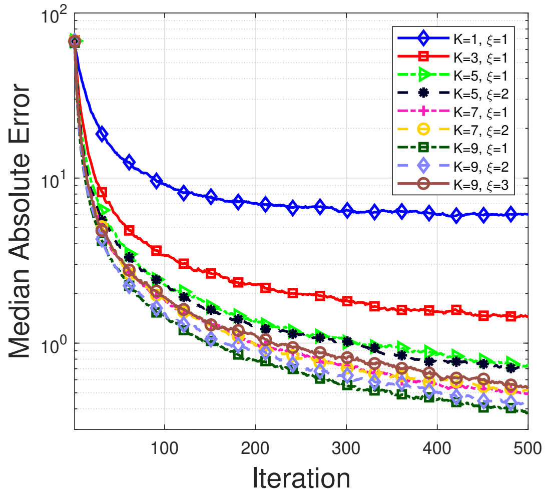

Fig. 1: Performance of CMPMC for various parameter settings.

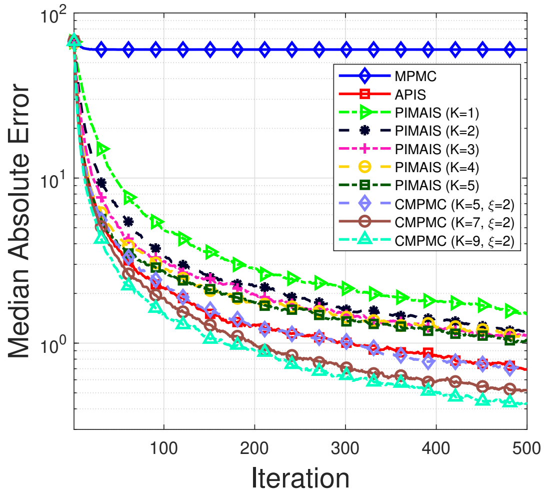

Fig. 2: Comparison of CMPMC to MPMC, APIS, and PIMAIS.

Given the expressions in (17), we can compute stochastic gradients w.r.t. the mixand parameters as according to (14) in order to adapt the mean and precision of each mixand.

5. SIMULATIONS

We validate the proposed method on a toy example, where the goal is to estimate the normalizing constant of a Gaussian mixture.

5.1. Approximating a Gaussian Mixture in $\mathbb{R}^2$

Suppose that the unnormalized target distribution has the form

$$\tilde{\pi}(\mathbf{x}) = 100 \times \left(\frac{1}{5} \sum_{j=1}^{5} \mathcal{N}(\mathbf{x}; \mathbf{m}_j, \mathbf{C}_j) \right), \tag{18}$$

where $\mathbf{m}_1 = [-10, 10]^\top$, $\mathbf{C}_1 = [2, 0.6; 0.6, 2]$, $\mathbf{m}_2 = [0, 16]$, $\mathbf{C}_2 = [2, -0.4; -0.4, 2]$, $\mathbf{m}_3 = [13, 8]$, $\mathbf{C}_3 = [2, 0.8; 0.8, 2]$, $\mathbf{m}_4 = [-9, 7]$, $\mathbf{C}_4 = [3, 0; 0, 0.5]$, $\mathbf{m}_5 = [14, -14]$, and $\mathbf{C}_5 = [2, -0.1; -0.1, 2]$. Our goal in this experiment is to estimate the normalizing constant given by (18), whose true value in this case is $Z = 100$. We compare the following methods: MPMC [15], adaptive population importance sampling (APIS) [17], PIMAIS [18], and CMPMC. For each algorithm, we use a mixture of $D = 25$ Gaussians as proposal distributions and draw $M = 200$ samples per iteration over $I = 500$ iterations. We remark that the APIS and PIMAIS algorithms assume that the mixture is equally weighted and a deterministic number of samples $N = M/D$ is drawn from each mixand. Furthermore, these algorithms only adapt the mean of each mixand.

We initialize with $\rho_{d,1} = \frac{1}{D}$, $\boldsymbol{\mu}_{d,1} \sim \mathcal{U}([-20, 20] \times [-20, 20])$, and $\boldsymbol{\Lambda}_{d,1} = \mathbb{I}_2$ for $d = 1, \ldots, D$. For the PIMAIS and CMPMC algorithms, we employ the MH algorithm with an isotropic Gaussian proposal with variance $\zeta_r^2 = 1$ to construct the Markov chains. For the CMPMC algorithm, we adapt the mean and precision of each component, where we use the RMSprop optimizer [25] to resolve the stochastic gradient updates w.r.t. $C_2(\boldsymbol{\theta})$. We evaluate the performance of each method by computing the absolute error of the normalizing constant estimate. The results are averaged over 500 MC simulations.

Figure 1 shows a comparison of the CMPMC algorithm in terms of median absolute error for different chain lengths K and thinning rates ξ. We can see that the worst performance is when $K = 1$, i.e. when only a single MH step is taken at each iteration. This can be attributed to the slower Markov chain convergence, which would lead to less accurate stochastic gradient estimates. By increasing the number of MH steps, the algorithm's performance improves substantially. We can also see that the thinning parameter ξ does have an effect on the performance of the algorithm, however, it is evident that the chain length K plays a more substantial role in this example.

A comparison of the CMPMC algorithm with MPMC, APIS and PIMAIS is shown in Fig. 2. We can see the MPMC algorithm performs the worst among all methods, obtaining the largest median absolute error. This is attributed to the MPMC algorithm consistently missing one of the five modes of the target distribution. Although APIS and PIMAIS perform well in this example, we can see that the proposed CMPMC algorithm attains better performance for all of the considered parameter settings. Since the CMPMC algorithm is optimizing an objective that is a monotonic transformation of the variance of an unbiased estimator of the normalizing constant, it is natural that it outperforms APIS and PIMAIS in this example.

6. CONCLUSIONS

This work proposed a novel adaptive importance sampling (AIS) methodology, called controlled mixture population Monte Carlo (CMPMC) that fully adapts the parameters of a mixture proposal distribution. In the CMPMC algorithm, the proposal parameters are adapted using an instance of a stochastic optimization algorithm. The novelty in the method is the way stochastic gradients are computed, which is by using a population of MCMC chains that are iteratively constructed throughout the algorithm. A separate MCMC chain is constructed for each mixand in order to guarantee that each mixand is well represented in the gradient computation. Numerical results show that the proposed scheme outperforms other AIS methods. Interesting prospects for future work include a reduction in the computational complexity of the algorithm, so that it may be on the same order of complexity as AIS methods like the original mixture population Monte Carlo (MPMC) scheme.

7. REFERENCES

[1] C. Robert and G. Casella, *Monte Carlo statistical methods*, Springer Science & Business Media, 2013.

[2] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian data analysis*, Chapman and Hall/CRC, 2013.

[3] J. V. Candy, *Bayesian signal processing: classical, modern, and particle filtering methods*, vol. 54, John Wiley & Sons, 2016.

[4] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[5] W. K. Hastings, "Monte Carlo sampling methods using markov chains and their applications," 1970.

[6] W. R. Gilks and P. Wild, "Adaptive rejection sampling for Gibbs sampling," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 41, no. 2, pp. 337–348, 1992.

[7] T. Bengtsson, P. Bickel, B. Li, et al., "Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems," in *Probability and statistics: Essays in honor of David A. Freedman*, pp. 316–334. Institute of Mathematical Statistics, 2008.

[8] C. Andrieu and J. Thoms, "A tutorial on adaptive mcmc," *Statistics and computing*, vol. 18, no. 4, pp. 343–373, 2008.

[9] F. Liang, C. Liu, and R. Carroll, *Advanced Markov chain Monte Carlo methods: learning from past samples*, vol. 714, John Wiley & Sons, 2011.

[10] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid Monte Carlo," *Physics letters B*, vol. 195, no. 2, pp. 216–222, 1987.

[11] M. Girolami and B. Calderhead, "Riemann manifold Langevin and Hamiltonian Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 2, pp. 123–214, 2011.

[12] K. L. Mengersen, R. L. Tweedie, et al., "Rates of convergence of the Hastings and Metropolis algorithms," *The annals of Statistics*, vol. 24, no. 1, pp. 101–121, 1996.

[13] M. F. Bugallo, V. Elvira, L. Martino, D. Luengo, J. Miguez, and P. M. Djuric, "Adaptive importance sampling: the past, the present, and the future," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 60–79, 2017.

[14] O. Cappé, A. Guillin, J. Marin, and C. P. Robert, "Population Monte Carlo," *Journal of Computational and Graphical Statistics*, vol. 13, no. 4, pp. 907–929, 2004.

[15] O. Cappé, R. Douc, A. Guillin, J. Marin, and C. P. Robert, "Adaptive importance sampling in general mixture classes," *Statistics and Computing*, vol. 18, no. 4, pp. 447–459, 2008.

[16] J. Cornuet, J. Marin, A. Mira, C. P. Robert, et al., "Adaptive multiple importance sampling," *Scandinavian Journal of Statistics*, vol. 39, no. 4, pp. 798–812, 2012.

[17] L. Martino, V. Elvira, D. Luengo, and J. Corander, "An adaptive population importance sampler: Learning from uncertainty," *IEEE Transactions on Signal Processing*, vol. 63, no. 16, pp. 4422–4437, 2015.

[18] L. Martino, V. Elvira, D. Luengo, and J. Corander, "Layered adaptive importance sampling," *Statistics and Computing*, vol. 27, no. 3, pp. 599–623, 2017.

[19] L. Martino, V. Elvira, D. Luengo, and J. Corander, "Parallel interacting Markov adaptive importance sampling," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, 2015, pp. 499–503.

[20] C. Andrieu and C. P. Robert, *Controlled MCMC for optimal sampling*, INSEE, 2001.

[21] E. K. Ryu and S. P. Boyd, "Adaptive importance sampling via stochastic convex programming," *arXiv preprint arXiv:1412.4845*, 2014.

[22] E. K. Ryu, *Convex optimization for Monte Carlo: Stochastic optimization for importance sampling*, Ph.D. thesis, Stanford University, 2016.

[23] Y. El-Laham, P. M. Djurić, and M. F. Bugallo, "A variational adaptive population importance sampler," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5052–5056.

[24] Y. El-Laham and M. F. Bugallo, "Stochastic gradient population Monte Carlo," *IEEE Signal Processing Letters*, vol. 27, pp. 46–50, 2020.

[25] M. C. Mukkamala and M. Hein, "Variants of rmsprop and adagrad with logarithmic regret bounds," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2545–2553.