Adaptive MCMC via Combining Local Samplers

Kiarash Shaloudegi[1]
[1]Imperial College London

András György[2,1]
[2]DeepMind

Abstract

Markov chain Monte Carlo (MCMC) methods are widely used in machine learning. One of the major problems with MCMC is the question of how to design chains that mix fast over the whole state space; in particular, how to select the parameters of an MCMC algorithm. Here we take a different approach and, similarly to parallel MCMC methods, instead of trying to find a single chain that samples from the whole distribution, we combine samples from several chains run in parallel, each exploring only parts of the state space (e.g., a few modes only). The chains are prioritized based on the kernel Stein discrepancy, which provides a good measure of performance locally. The samples from the independent chains are combined using a novel technique for estimating the probability of different regions of the sample space. Experimental results demonstrate that the proposed algorithm may provide significant speedups in different sampling problems. Most importantly, when combined with the state-of-the-art NUTS algorithm as the base MCMC sampler, our method remained competitive with NUTS on sampling from unimodal distributions, while significantly outperformed state-of-the-art competitors on synthetic multimodal problems as well as on a challenging sensor localization task.

1 Introduction

We consider the problem of computing expectations $\mathbb{E}_P[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx$ for some complicated target distribution P with density p over a set $\mathcal{X} \subset \mathbb{R}^d$ and a target function $f : \mathcal{X} \to \mathbb{R}$. Such expectations often arise in Bayesian inference and maximum likelihood estimation (Andrieu et al., 2003; Brooks et al., 2011). Oftentimes, p has a closed form, but it is only known up to a normalization constant, making the computation of the integral especially challenging (Andrieu et al., 2003). Markov chain Monte Carlo (MCMC) methods are a family of numerical estimation methods, which are successfully applied to estimate the aforementioned expectations, especially in high-dimensional problems. MCMC algorithms take random samples from an ergodic Markov chain with stationary distribution P, and approximate the expectation via averaging over the produced sample.

The challenging problem in designing MCMC methods is to ensure that the distribution of the samples converge to P fast, and, in practice, some domain specific knowledge has to be used in the design of their proposal distributions to achieve fast convergence (Andrieu et al., 2003). This need for specialized design led to the development of dozens of methods for each problem, each of which has their own tunable parameters (Neufeld et al., 2014). Consequently, choosing the right method with corresponding parameters to achieve fast convergence is quite difficult and requires considerable time and effort.

A large body of work has been devoted in the literature to address this difficulty and to find ways to set the algorithms' parameters optimally; for instance, optimal tuning of the Metropolis-Hasting algorithm Roberts & Rosenthal, 2001; Bédard, 2008; Roberts et al., 1997; Atchadé et al., 2011; Brooks et al., 2011, Chapter 4). The problem with this line of research is that the solutions rely on some Markov chain parameters that are typically unknown (Łatuszyński et al., 2013).

A more promising line of research to address the parameter setting issue is based on adaptive MCMC methods. In this framework, the MCMC samples are used to learn about the target distribution, and the algorithms adjust their parameters as they progresses (Łatuszyński et al., 2013). To do so, they rely on optimizing some objective functions such as expected squared jumping distance (Pasarica & Gelman, 2010; Wang et al., 2013), the area under the autocorrelation function up to some

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

specific lag (Mahendran et al., 2012), the difference between the proposal covariance matrix and its empirical estimation (Haario et al., 2001, 2006; Sejdinovic et al., 2014; Mbalawata et al., 2015), or the difference between the optimal acceptance rate (in practice, a recommendation thereof) and its empirical estimate (Yang & Rosenthal, 2017). Perhaps the most successful adaptive method for finding the optimal step size in Hamiltonian Monte Carlo (HMC), called no-U-turn sampler (NUTS), is based on monitoring when the sample trajectories "turn back", and the resulting algorithm provides state-of-the-art performance in a number of problems Hoffman & Gelman (2014). For more details about adaptive MCMC methods, the reader is referred to the tutorial paper of Andrieu & Thoms (2008).

In practice, making sure that a sampler can move between distant modes cannot be guaranteed by the aforementioned adaptive methods (in general, we are interested in "high-probability" regions not modes, but, for simplicity and following the standard language of MCMC literature, we will often refer to modes throughout the paper). There are two main approaches to deal with distant modes: (i) running parallel chains and combining their final samples; and (ii) sampling from powers of the target function (the inverse power is referred to as temperature), known as annealing. Indeed, in the literature, there are several successful methods based on combining these two ideas, such as parallel tempering (Earl & Deem, 2005) and (annealed) sequential Monte Carlo samplers, also known as particle filters (Doucet et al., 2001; Moral et al., 2006). The core idea of these methods and their variants is to run several chains with different temperatures and periodically exchange information between them. An alternative to parallelization is the idea of *regeneration* (Mykland et al., 1995; Ahn et al., 2013). Regeneration partitions a long Markov chain into smaller independent segments such that the samples are unbiased in each segment, hence can be combined together without any further consideration. The process allows to combine samples from different samplers and tune the parameters of a Markov chain after each regeneration. Although theoretically elegant, the application of regeneration methods is limited in practice since they require a properly tuned distribution for detecting regenerations.

In this paper we combine the strengths of adaptive and parallel MCMC methods. Instead of trying to find a single sampler that approximates the target distribution well on its whole domain, we run several samplers and select which sampler to use at any given time in a *sequential* manner, based on all the samples obtained before. Our main contributions are (i) adapting the bandit-based adaptive Monte Carlo (MC)[1] method of Neufeld et al. (2014) to MCMC; and (ii) a novel method for combining samples from multiple chains. The resulting algorithm is suitable for sampling from challenging multimodal distributions and is fairly insensitive to the choice of its parameters. The next subsection gives a more detailed overview of our approach and the corresponding challenges.

1.1 Approach and challenges

In the simple case when all MCMC samplers mix well over the whole domain, our goal is to use the best sampler (which mixes the fastest) most of the time. This is very similar to the case of choosing from several unbiased MC samplers. For the latter, Neufeld et al. (2014) showed that scheduling which samplers to use at any point in time is equivalent to a stochastic multi-armed bandit problem (Bubeck & Cesa-Bianchi, 2012). This allows the straightforward application of bandit algorithms to select which sampler to use to get the next sample, and the decision depends on the overall performance of the samplers so far, measured by the variance for each sampler. Extending the same idea to the MCMC case is not trivial, since measuring the quality of MCMC samplers is a much harder task. In fact, until recently, there has not been any empirical measure that can monitor the sample convergence. Common MCMC diagnostics such as effective sample size, trace and mean plots, or asymptotic variance assume the chain asymptotically converges to the target distribution, so they cannot detect asymptotic bias (Gorham & Mackey, 2015). To address this issue, Gorham & Mackey (2015) developed an empirical sample-quality measure that can detect non-convergence (or bias) based on Stein's method. A kernelized version of this measure, called *kernel Stein discrepancy* (KSD) was subsequently developed by Liu et al. (2016); Chwialkowski et al. (2016); Gorham & Mackey (2017), which can be used to compare the quality of different samplers. *As our first contribution, we extend the bandit-based racing method of Neufeld et al. (2014) to MCMC samplers by using the KSD as the loss function in the bandit algorithms.* This is described in detail in Section 3.1 while the background on KSD is given in Section 2.

On the negative side, KSD is not able to detect underfitting if the target distribution has well-separated modes, so it cannot distinguish between two samplers such that one samples only one mode while the other samples both modes, and the samples are equally good locally (Liu et al., 2016; Chwialkowski et al., 2016; Gorham & Mackey, 2017). This brings us to the next problem: namely, MCMC methods using a single chain usually fail to explore the whole domain if the support has reasonably high-probability regions separated by

[1] Throughout the paper, we refer to samplers taking unbiased, independent samples as MC methods.

low-probability regions (of course, such notion of separation depends on the actual sampler used). Setting the parameters of the samplers to deal with this issue, or simply detecting its presence, is hard, and to our knowledge no practical solutions thereof are available. To alleviate this problem, following the parallel MCMC framework, we run several chains in parallel only expecting that they provide good samples locally. The hope is that the multiple instances will explore the space sufficiently, finding all the important regions of the support. Then, in the end, we combine all the samples (from all the samplers) to approximate the target distribution. This is again challenging due to two reasons: (i) it is not straightforward how the samples from different samplers should be weighted, and (ii) we do not want to waste resources to run several samplers exploring the same region of the domain. For (ii), we apply our bandit-based racing method locally (Section 3.2), while to address (i), *we develop a method to estimate the probability of the region a set of samples cover based on Rényi-entropy estimates (Pál et al., 2010), and use these to weight the samples, which is our second main contribution.* This is described in Section 3.3.

Our final sampling algorithm is put together in Section 4. Lastly, in Section 5 we demonstrate through a number of experiments that our method is competitive with state-of-the-art adaptive MCMC methods, such as the no-U-turn sampler NUTS (Hoffman & Gelman, 2014) or the recent sample reweighting method of Liu & Lee (2016) on simpler cases when the distribution is concentrated on a single "connected" region, while significantly outperforming the competitors, including parallel tampering and sequential MC, on more challenging cases where high-probability regions are separated with areas of low-probability, as well as on a challenging sensor-localization problem.

Due to space constraints, some details are omitted, and are only available in the long version of the paper (Shaloudegi & György, 2018).

2 Measuring sample quality

As mentioned in Section 1.1, measuring the quality of samples produced by an MCMC algorithm is crucial in our approach. To this end we are going to use the recently introduced *kernel Stein discrepancy* (KSD) (Liu et al., 2016; Chwialkowski et al., 2016; Gorham & Mackey, 2017).

Assume that the probability distribution Q_n over $\mathbb{R}^d$ is given by a weighted sample $\{(x_i, q_i)\}_{i\in[n]}$, where $[n]$ denotes $\{1,\dots,n\}$, $x_i \in \mathbb{R}^d$ and $Q_n(x_i) = q_i \geq 0$ with $\sum_{i\in[n]} q_i = 1$. Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a positive definite kernel and $\mathcal{K}$ its associated reproducing kernel Hilbert space (RKHS) with inner product $\langle\cdot,\cdot\rangle$. Then, for $x \in \mathbb{R}^d$, $f \in \mathcal{K}$, and $k(x,\cdot) \in \mathcal{K}$, we have $f(x) = \langle f, k(x,\cdot)\rangle$. Defining $s_p(x) \triangleq \nabla \log p(x)$ and

$$k_p(x,x') \triangleq s_p(x)^\top k(x,x') s_p(x') + s_p(x)^\top \nabla_{x'} k(x,x') + s_p(x')^\top \nabla_x k(x,x') + \mathbf{trace}(\nabla_x \nabla_{x'} k(x,x')),$$

the *kernel Stein discrepancy* (Gorham & Mackey, 2015, 2017), which measures how well Q_n approximates P, is defined as

$$\mathcal{S}(Q_n) = \sqrt{\mathbb{E}_{Q_n\times Q_n}[k_p(Z,Z')]} = \sqrt{\mathbf{q}^\top K_{p,n}\mathbf{q}}, \quad (1)$$

where $\mathbf{q} = (q_1,\dots,q_n)$ and $K_{p,n} = [k_p(x_i,x_j)]_{i,j\in[n]}$, provided $k(x,y)$ is twice differentiable and $\nabla_x\nabla_y k(x,y)$ are continuous, both are uniformly bounded, and $\mathbb{E}_P[\|\nabla_x \log p(X)\|_2] < \infty$. Note that $\mathcal{S}(Q_n)$ can be computed with our information about p, since it only depends on p through $s_p(x) = \nabla \log p(x)$, which cancels the effect of the unknown normalization constant. Also note that $\mathcal{S}(Q_n)$ is a convex function of Q_n.

It is known that under some conditions, such as for unimodal distributions P with log-concave densities, the KSD measure goes to 0 if and only if Q_n converges weakly to P. However, the discriminative power of KSD weakens for multimodal distributions, particularly when the modes are well-separated: Gorham et al. (2016) demonstrated that for a one-dimensional Gaussian mixture target distribution (with two components), for practical sample sizes, the KSD measure fails to distinguish between two sets of samples, one drawn independently from one mode and the other drawn independently from the whole target distribution. Furthermore, KSD requires even more samples to distinguish between the two cases as the modes' distance increases (see Section 6.1 of Gorham et al., 2016 for more details). Another issue is that the complexity of computing the KSD for an empirical distribution is quadratic in the sample size, which quickly becomes infeasible as the sample size grows.

3 Sequential selection of samplers

In this section we present several strategies to select from a pool of MCMC samplers in a sequential manner. In all of our algorithms, the selection of the sampler to be used next depends on the quality of the samples generated by the different samplers, where the quality will be measured by the KSD measure (or its approximations). Formally, assume we have access to M MCMC samplers (e.g., multiple sampling methods and/or multiple instances of the same sampling algorithm with different parameters, such as starting point or step size), and denote the set of samplers by $[M]$. At every step of the algorithm, we select one of the samplers and use it to produce the next batch of samples.

3.1 Mixing samplers

First we consider the case when each sampler is asymptotically unbiased, that is, generates samples with an empirical distribution converging weakly to the target distribution P (this is usually satisfied for any standard MCMC sampler when P is unimodal). Our task is to sequentially allocate calls among the M samplers to minimize the KSD measure of the set of samples we collect. The goal is to design an algorithm which gives preference to samplers where the convergence is faster. This setup is similar to the one considered by Neufeld et al. (2014), who designed sequential sampling strategies for MC samplers generating independent and identically distributed (i.i.d.) samples, based on multi-armed bandit algorithms (Bubeck & Cesa-Bianchi, 2012). In this section we generalize their method to MCMC samplers. Our overall goal is to produce samples such that the total KSD measure (cf. Eq. 1) of the samples is small. However, computing the KSD measure is quadratic in the sample size, and so it becomes computationally infeasible even for relatively small sample sizes—note that any computation we spend on selecting samplers could also be used for sampling. Therefore, we are going to approximate the KSD measure as the average KSD over smaller blocks of samples.

For a sampler with total sampling budget n, we break the sampling process into T rounds: At each round the sampler takes a batch of samples of size n_b. Let S_t be the KSD measure of samples from the tth round; we approximate $\tilde{S}_n$, the KSD of the full sample of size n with the average $(1/T)\sum_{t=1}^{T} S_t$. We call this the *block-diagonal approximation*, as it corresponds to a block-diagonal approximation of the kernel matrix $K_{p,n}$ in computing (1). To quantify the accuracy of the approximation, we assume that there exists a function $g(t, n_b)$ such that $\lim_{\frac{n_b}{t}\to\infty} g(t, n_b) = 0$ and $\frac{1}{t}\sum_{b=1}^{t} \mathbb{E}[S_b] - \mathbb{E}[\tilde{S}_{tn_b}] \le g(t, n_b)$. Using the block-diagonal approximation, our goal is to compete with a sampler with the smallest average approximate block-KSD measure $(1/T)\sum_{t=1}^{T} S_{i,t}$, where $S_{i,t}$ is the KSD measure of the n_b samples generated by sampler i when it is called the tth time. Experimental results (presented in Section 5.1 of Shaloudegi & György, 2018) indicate that the block-diagonal approximation mostly preserves the ranking of the samplers (as defined by the true KSD measure), hence we pay very little price for the computational advantage we get.

Furthermore, solving this problem is well-suited for any bandit algorithm; here we adapt the UCB1 method of Auer et al. (2002). The resulting algorithm, which we call *KSD-UCB1*, keeps track of an optimistic estimate of the average approximate KSD value for each sampler, and every time selects a sampler whose performance is estimated to be the best possible (with high probability). More precisely, every time a sampler is selected, it is used to generate n_b samples. For each block of such samples the algorithm computes the KSD measure, and for each sampler $i \in [M]$ keeps track of the average KSD measure $\bar{\mu}_{i,t} = (1/T_i(t))\sum_{s=1}^{T_i(t)} S_{i,s}$ where $T_i(t)$ denotes the number of times sampler i is selected during the first $t-1$ choices. In the first M steps, each sampler is selected once to produce n_b samples. Then, in the $(t+1)$st step (with $t+1 > M$), the sampler minimizing $\bar{\mu}_{i,t} - \sqrt{2\log t/T_i(t)}$ (a high-probability lower bound on the KSD approximation) is selected to produce the next block of samples.[2]

If the KSD values $S_{i,1}, \dots, S_{i,T}$ were i.i.d., the standard bandit regret bound (Bubeck & Cesa-Bianchi, 2012) would yield $\sum_{t=1}^{T} \mathbb{E}[S_t] - \min_{i\in[M]} \sum_{t=1}^{T} \mathbb{E}[S_{i,t}] = O(\log T)$. In this case the convexity of the KSD measure would imply that after T rounds of sampling,

$$\begin{aligned}
&\mathbb{E}[\tilde{S}_{Tn_b}] - \min_i \mathbb{E}[\tilde{S}_{i,Tn_b}] \\
&\le \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[S_t] - \min_{i\in[M]} \sum_{t=1}^{T} \mathbb{E}[S_{i,t}] + g(T, n_b) \\
&\approx \frac{O(\log T)}{T} + g(T, n_b)\ .
\end{aligned}$$

This shows that increasing T and n_b, the performance of KSD-UCB1 would be close to that of the best sampler. However, in our case, the $S_{i,t}$ are not i.i.d. Assuming that the samplers mix (which is reasonable for a single mode distribution), the $S_{i,t}$ are getting closer and closer to be sampled i.i.d. as n_b increases. Also, as mentioned above, the effect of the block-diagonal approximation (and hence that of $g(T, n_b)$) is small in practice.

3.2 Locally mixing samplers

In practice, if the target distribution is multimodal and the modes are far from each other, MCMC methods often get stuck in some of the modes and fail to explore all the regions where P is supported; while eventually all asymptotically consistent methods reach each mode, this may not happen after any practically reasonable time. To model this situation, we assume that the support of P is partitioned into sets $A_1, \dots, A_K$ with $P(A_j) > 0$ for all $j \in [K]$ (the A_j are pairwise disjoint and their union is the support of P) such that the empirical distribution of the samples generated by sampler $i \in [M]$ converges weakly to $P(\cdot|A_j)$ for some $j \in [K]$, where $P(\cdot|A)$ is the conditional distribution of P over A. We refer to the sets A_j as *regions*, and a

[2] The algorithm assumes that $S_{i,t} \in [0, 1]$, which can be achieved by rescaling the KSD measures. In practice, a reasonable estimate for the range can be obtained from $S_{i,1}$ for each sampler i.

sampler satisfying the above condition a *locally mixing* sampler.

For simplicity we first consider the case where there is one sampler in each region (consequently $M = K$). This setup is similar to stratified sampling: The idea is to partition the domain into non-overlapping regions (a.k.a. strata), draw samples from each region, and combine the final samples to estimate $\mathbb{E}_P[f(X)]$ (Owen, 2013). The problem in stratified sampling is to find the optimal number of samples that need to be taken from each stratum in order to minimize the Monte Carlo integration error. Given the total number of samples n, the optimal strategy for minimizing the mean squared error (MSE) is to sample each stratum $n_i = \frac{\sigma_i P(A_i)}{\sum_{i=1}^K \sigma_i P(A_i)} n$ times (relaxing the integrality constraints), where σ_i is the conditional standard deviation of $f(X)$ given that X falls into the ith region (Carpentier et al., 2015).

One can immediately see that the problem we consider in this subsection is very similar to stratified sampling, with the important differences that our samplers are not i.i.d., and we do not minimize the squared error but the KSD measure. Denoting the distribution of samples from region A_i by $Q_{n,i}$ after taking n samples in total, let w_i denote the weight of sampler i generating these samples (recall that here, by assumption, we have one sampler in each region). Then our total weighted sample distribution becomes

$$Q_n = \textstyle\sum_{i=1}^M w_i Q_{n,i}. \tag{2}$$

Since according to our assumptions, $Q_{n,i}$ converges weakly to P_{A_i} for every i, we need to have $w_i \to P(A_i)$ for all $i \in [M]$ to ensure that Q_n converges to P weakly (we will refer to this as the w_i being asymptotically consistent). A procedure for estimating w_i this way will be given in the next section. Assuming for a moment that $w_i = P(A_i)$, using the convexity of the KSD measure, we have

$$\tilde{S}_n = \mathcal{S}(Q_n) \le \sum_{i=1}^M w_i \mathcal{S}(Q_{n,i}) = \sum_{i=1}^M w_i \tilde{S}_{i,n_i}, \tag{3}$$

where, as before, $\tilde{S}_n$ and $\tilde{S}_{i,n_i}$ denote the KSD measures of the whole sample and, resp., that of sampler i (with number of samples n_i obtained by sampler i). Based on this inequality, we could aim for minimizing $\sum_{i=1}^M w_i \tilde{S}_{i,n_i}$ and use the ideas from adaptive stratified sampling (Carpentier et al., 2015); however, multiple challenges preclude us to do it: (i) w_i is not known in advance; (ii) the stratified sampling algorithm is based on the known convergence rate of the sample average, but we do not know how fast $\tilde{S}_{i,n_i}$ approaches zero (as a function of n_i); and (iii) the computational complexity of calculating $\tilde{S}_{i,n_i}$ is $O(n_i^2)$. To handle (i), we address the problem of estimating w_i in Section 3.3. For (ii), a conservative approach is to uniformly minimize $w_i \tilde{S}_{i,n_i}$, hence selecting sampler i for which this quantity is the largest. For (iii), we again use a block-diagonal approximation to $\tilde{S}_{i,n_i}$, which causes problems with (ii), since the estimate does not converge to 0 for a fixed block size. We tested experimentally several methods to alleviate these problems (Shaloudegi & György, 2018, Section 5.3): The simplest strategy considered was to select regions uniformly at random. Another approach was to minimize the maximum $\tilde{S}_{i,n_i}$, that is, choosing $i = \operatorname{argmax} \tilde{S}_{i,n_i}$, which does not require the knowledge of the weights w_i, while we also tested strategies selecting regions based on their estimated weights or, similarly to stratified sampling, based on their estimated variance. Although quite different, the strategies considered seemed to perform rather similarly under different circumstances.

Unfortunately, we cannot guarantee that we start samplers in such a way that we have a single sampler for each region. If we know which sampler belongs to which region (recall that we assume that the samplers are locally mixing and hence belong to a region A_i), we can combine any of the region selection strategies described above with the bandit method described in Section 3.1 in a straightforward way: in each region A_i, run an instance of KSD-UCB1 over the samplers exploring this region, and use this KSD-UCB1 instance as a single locally mixing sampler in any of the above region selection methods. Finally, when the sampling budget is exhausted (n samples are generated), estimate the weights w_i, and reweight the samples corresponding to region A_i according to (2). We refer to this procedure as *KSD-UCB1-M* (M stands for Multiple regions). Clearly, since the bandit algorithms sample each sampler infinitely often, we have the following consistency result:

Under the local mixing assumption made at the beginning of this section, the final weighted sample (2) *obtained by KSD-UCB1-M is asymptotically unbiased as long as the weight estimates w_i are asymptotically unbiased.*

Thus, we need to find some asymptotically unbiased estimates of the probabilities of the regions A_i.

3.3 Weight estimation

In this section we consider the problem of finding the weights w_i in (2). As discussed after the equation, this amounts to finding the probability of the region the samples cover, which is again challenging since we have access only to an unnormalized density. This problem is faced by every algorithm which tries to speed up MCMC methods by running parallel chains. As an example, in big-data scenarios it is common to

split the data into subsets, run an MCMC sampler on each subset, and combine the resulting samples in the end. To our knowledge, most work in the literature solves this problem by estimating the density of each batch of samples separately (Angelino et al., 2016; Nemeth & Sherlock, 2018), using typically either a Gaussian approximation (Scott et al., 2016) or some kernel-density estimation method (Neiswanger et al., 2014). According to Nemeth & Sherlock (2018), the first approach works well in practice, in spite of not being supported by any theory, while the second, kernel-based estimation scales poorly with the dimension d.

Here we take a different approach and rather than estimating the density of the sample batches, we directly estimate the probabilities $P(A_i)$ via Rényi entropy.

Formally, suppose the domain of P is partitioned into non-overlapping regions $A_1, \dots, A_K$, and from each region A_i we have a set of samples $X^{(A_i)}$. The Rényi entropy of order $\alpha \neq 1$ for a density p is defined as $R_\alpha(p) = \frac{1}{1-\alpha} \log \int_{\mathbb{R}^d} p^\alpha(x) \mathrm{d}x$. The conditional density of P restricted to a set A is denoted as $p(x|A) = \frac{p(x)}{P(A)} \mathbb{I}_{\{x \in A\}}$, and its Rényi entropy is $R_\alpha(p|A) = \frac{1}{1-\alpha} \log \int_A \left(\frac{p(x)}{P(A)}\right)^\alpha \mathrm{d}x$. From this definition it trivially follows that $\log P(A) = R_\alpha(p|A) - \frac{1}{1-\alpha} \log \mathbb{E}\left[p(X)^{\alpha-1} \middle| X \in A\right]$. In our case, instead of p we only have access to $\hat{p} = cp$ for some $c > 0$. Replacing p with $\hat{p}$ in the integral, we obtain $\log(P(A)c) = R_\alpha(p|A) - \frac{1}{1-\alpha} \log \mathbb{E}\left[\hat{p}(X)^{\alpha-1} \middle| X \in A\right]$. Thus, we can estimate $P(A)$ by estimating the two terms above.

Given a sample $X^{(A)}$ taken i.i.d. from $p(\cdot|A)$, the second term can be estimated by the empirical average $\hat{B}_\alpha(X^{(A)}) = \frac{1}{|X^{(A)}|} \sum_{x \in X^{(A)}} \hat{p}(x)^{\alpha-1}$, while for the first term we can use a graph-based estimate (Hero & Michel, 1999). In particular, we are going to use the estimator $\hat{R}_\alpha(X^{(A)})$ of Pál et al. (2010), which is based on generalized k-nearest neighbor graphs of the sample $X^{(A)}$, and it converges to $R_\alpha(p|A)$ for any $\alpha \in (0, 1)$ as the sample size grows to infinity. Thus, we obtain that $\beta(X^{(A)}) = \hat{R}_\alpha(X^{(A)}) - \hat{B}_\alpha(X^{(A)})$ is an asymptotically unbiased estimate of $\log(P(A)c)$. Therefore, given a fixed partition $A_1, \dots, A_K$,

$$P(A_i) \approx w_i = e^{\beta(A_i)} / \textstyle\sum_{j \in [K]} e^{\beta(A_j)} \ . \tag{4}$$

More precisely, if the minimum number of samples in the partitions is m, then $P(A_i) = \lim_{m \to \infty} w_i$.

It is also possible to determine the rate of convergence in the above: Theorem 2 of Pál et al. (2010) shows that with a proper parametrization of the estimator $\hat{R}_\alpha(X^{(A)})$, its (additive) error after $m = |X^{(A)}|$ samples is of order $m^{-1/(d+\gamma)}$ with high probability for an arbitrary choice of $\gamma > 0$. On the other hand, the error of $\hat{B}_\alpha(X^{(A)})$ is of order $O(m^{-1/2})$ by standard concentration inequalities if p is bounded (see, e.g., Boucheron et al., 2013). This implies that $|\beta(X^{(A)}) - \log(P(A)c)| = O(m^{-1/(d+\gamma)})$. Therefore, $P(A_i)$ can be estimated with a multiplicative error of $O(\exp(const \cdot m^{-1/(d+\gamma)}))$ where m now denotes the minimum number of samples over the partition cells $A_1, \dots, A_K$; in other words, $|\log P(A_i) - \log w_i| = O(m^{-1/(d+\gamma)})$. Note that although the error of our estimator scales quite unfavorably in the dimension d, in practice it seems to work well even for moderately large dimensions (around 20); see Section 5 for details.

4 The final algorithm

So far we have discussed how to solve our problem if we have either local mixing or all the samplers mix globally. While the latter is the case asymptotically for all the MCMC samplers used in practice, the mixing may be too slow to be observed for any practical number of samples. On the other hand, the problem does not simplify to the local mixing scenario, since–even if well-separated regions are actually present–, the chains often jump from one region to another even if they do not cover the whole domain.

To be able to adapt our KSD-UCB1-M algorithm, we need to group together the samplers covering the same region (even though what a region is is not clearly defined in this scenario). The problem is especially hard since the grouping of the samplers is non-stationary, and we should also be able to track when a sampler leaves or joins a region (equivalently, group). Furthermore, if the groups are too large, we do not explore the whole domain, while if they are too small, we waste resources by running multiple samplers for the same region.

To solve this issue, we propose a simple heuristic to identify samplers that are close together: In each round of the sampling, we take all the samples from the last batch of each sampler, and for each sample point we look at its N nearest neighbors. Then we find the grouping where any two samplers are grouped together that have points which are nearest neighbors of each other (this can easily be done recursively). By this simple heuristic, in each round we can group together the samplers that are close to each other. Note that here we do not make any assumption regarding the number of the regions (e.g., well-separated modes) of the distribution. By having all the samples, the algorithm can easily identify multiple regions by running a clustering algorithm. The final step of the algorithm is to determine the correct weight for each sample point. Recall that in the locally mixing case we weighted the empirical distribution of each region by their estimated probability (cf. Eq. 2). Here, since we do not have

Algorithm 1 KSD-MCMC-WR

Given: Distribution $p(x)$; M samplers; total number of rounds T; batch size n_b; number N of nearest neighbors for clustering; the order α of the Rényi entropy.
Initialize: For each $i \in [M]$, draw n_b samples from the ith sampler with random initialization; compute $S_{i,1}$ and set $\bar{\mu}_{i,1} = S_{i,1}$ and $T_i(1) = 1$.
for $t \in \{M+1, \dots, T\}$ **do**
- Cluster the samplers by clustering the samples from their last batches:
  - Initially, the last batch of samples from each sampler forms a cluster.
  - Merge two clusters if any point of one cluster has a point from the other cluster among its N nearest neighbors.
  - Find the number of clusters n_c.
  - Define $\mathcal{A}_i \subset [M]$ for $i \in [n_c]$ as the set of samplers belonging to cluster i ($\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ for $i \neq j$).
- Choose a cluster I_t (e.g., uniformly at random).
- Draw a batch of samples of size n_b from sampler $i_t = \arg\min_{i \in \mathcal{A}_{I_t}} \left(\bar{\mu}_{i,T_i(t-1)} - \sqrt{\frac{8 \log t}{T_i(t-1)}} \right)$.
- Set $T_i(t) = T_i(t-1) + 1$ and $T_j(t) = T_j(t-1)$ for $j \neq i$.
- Observe $S_{i,T_i(t)}$, and compute $\bar{\mu}_{i,T_i(t)} = (1 - \frac{1}{T_i(t)}) \bar{\mu}_{i,T_i(t-1)} + \frac{S_{i,T_i(t)}}{T_i(t)}$.

end for
- Cluster all the samples into M clusters by k-means clustering; for each cluster calculate its estimated probability w_i using (4) and output the reweighted samples with weight w_i/n_i where n_i is the total number of samples in cluster $i \in [M]$.

these regions, in the end we assign the samples into M clusters using k-means clustering, and weight the empirical measure within each cluster by the estimated probability of the cluster (using Eq. 4). Algorithm 1 shows the whole procedure, called KSD-MCMC with reweighting (KSD-MCMC-WR).

5 Experiments

We conducted a large number of experiments to empirically evaluate the choices in our design process. Due to space limitations, these are omitted from the paper, together with some details of the experiments presented in this section (all of these details are available in Section 5 of Shaloudegi & György 2018).

One of the conclusions of the aforementioned experiments is that our final algorithm KSD-MCMC-WR should be used with NUTS as its base sampler (recall that NUTS is a state-of-the-art MCMC method for unimodal problems). The left figure in Figure 1 shows that in a typical multimodal scenario (where the modes of the target distribution are relatively far), different versions of our method can significantly outperform both individual instances and a vanilla uniformly averaged parallel version of NUTS in estimating the mean of the target distribution (*Uniform:* each NUTS sampler is used equally; *UCB1* and *ϵ-greedy* are the bandit algorithms used by KSD-MCMC-WR; *equal probability* means the regions are selected uniformly, otherwise proportionally to $w_i \tilde{S}_{i,n_i}$). The right figure demonstrates that KSD-MCMC-WR remains competitive with NUTS for unimodal problems even for larger dimensions.

The next experiment compares KSD-MCMC-WR with two popular parallel MCMC samplers, annealed sequential Monte Carlo with resampling (SMC) (Moral et al., 2006) and parallel tempering (PT) (Earl & Deem, 2005). Since SMC and PT are problematic to use with NUTS, we apply MALA as the base sampler for these methods, while we keep NUTS for KSD-MCMC-WR (other omitted experiments confirm that our method with a MALA base sampler is usually superior to SMC and PT). Figure 3 shows how the relative performance of the algorithms change for estimating the mean of random Gaussian mixtures with 5 isotropic modes as the dimension increases (note that for each d the parameters of the distributions were selected independently). The results show that KSD-MCMC-WR provides consistently much better results than SMC and PT.

The two novel components of our final algorithm, KSD-MCMC-WR, are (i) using bandit algorithms with clustering and KSD-approximations as rewards; and (ii) clustering and reweigthing in the end via estimating Rényi entropy. While (i) requires (ii) to combine the parallel chains, perhaps surprisingly, when the modes of the target distribution are close enough for the base samplers to easily move from one to another, but they are far enough so that the KSD measure cannot distinguish if a chain explores a single mode or multiple modes clustered together, (ii) often leads to equal or superior performance (shown as "uniform+clustering" in the figures), since typically several chains are clustered together which do not fully explore all the modes corresponding to their cluster. This phenomenon is demonstrated in Figure 2, where the left figure shows a case when the modes are relatively far and KSD-MCMC-WR is better, while the right figure shows a situation where the modes are close, and "uniform+clustering" is better (the effect is amplified as the number of base samplers increases).

Our last experiment considers a realistic (yet still synthetic) problem of Ihler et al. (2005), where the task is to determine the location of several sensors randomly placed in a planar region from noisy pairwise distance measurements, where the probability of receiving a measurement decreases with the distance of the corresponding two sensors. This setup results in a multimodal posterior distribution. Figure 4 shows that KSD-MCMC-WR also excels in this problem, significantly outperforming both SMC and PT.

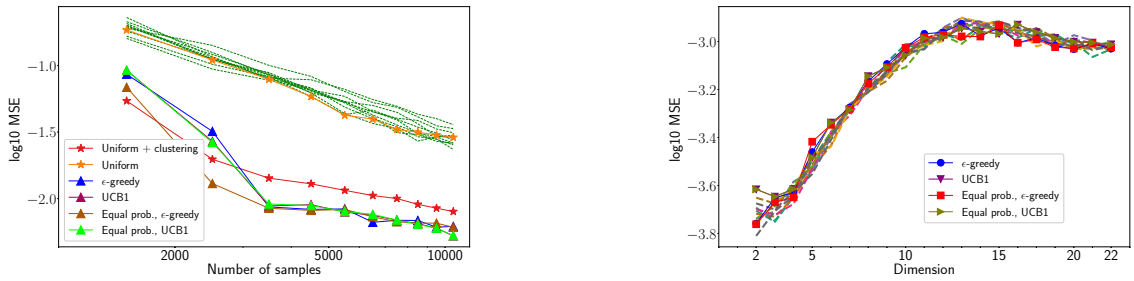

Figure 1: MSE for different sample sizes and dimensions with separated modes for variants of KSD-MCMC-WR. Left: The target distribution is a 2-dimensional Gaussian mixture with 5 isotropic modes (with parameters selected randomly). The dashed green lines show the results for the individual NUTS samplers. Right: Unimodal multivariate Gaussian target distribution with random parameters. The total number of samples is 10000.

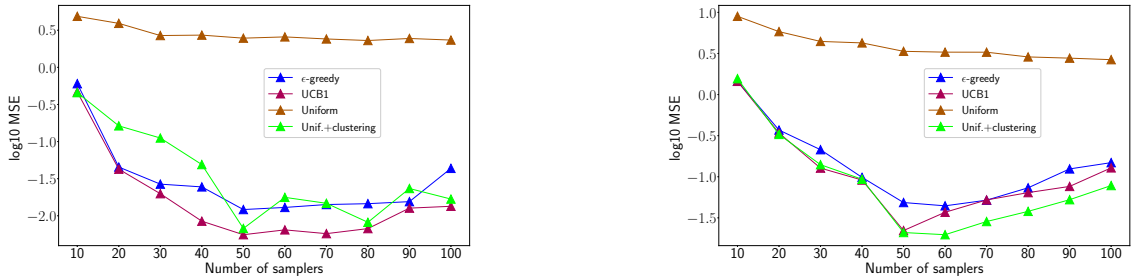

Figure 2: MSE for different number of NUTS samplers as a function of number of samplers for a 2-dimensional Gaussian mixture model with 20 isotropic modes. Two representative cases are shown: (left) the modes are far from each other, (right) some modes are close to each other. The total number of samples is 20000.

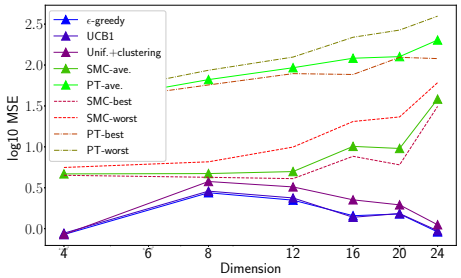

Figure 3: MSE for different dimensions when using KSD-MCMC-WR with 10 randomly initialized NUTS samplers and batch size $n_b = 10$, for Gaussian mixture targets with 5 isotropic modes. SMC and PT were run separately for every parameter setting of MALA (10 different parameters) with the best, worst and average values reported.

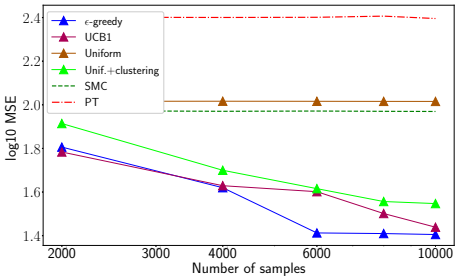

Figure 4: Running KSD-MCMC-WR for a sensor network localization problem with 10 NUTS instances started with different (random) initialization. The dashed green lines show the results for the individual NUTS samplers.

In summary, in all experiments KSD-MCMC-WR using NUTS as base sampler provided excellent performance, always being competitive with, but usually significantly better than the state-of-the-art.

6 Conclusion

Selecting the best MCMC method and its best parameter setting for a given problem is a hard task. Even if the parameters are selected correctly, in case of multimodal distribution, MCMC chains can easily get trapped in different modes for a long time, which may lead to biased results. To solve these issues, in this paper we proposed an adaptive MCMC method, KSD-MCMC-WR, that runs several chains in parallel, measures the quality of samples from each mode locally via kernel Stein discrepancy, and decides sequentially how to allocate the sampling budget among the different samplers using multi-armed bandit algorithms. The final step of our method is to combine the samples obtained by the different samplers: for this we developed a novel weighting scheme based on Rényi-entropy estimation, which might be of independent interest. Extensive experiments on several setups demonstrated that in typical scenarios, KSD-MCMC-WR with NUTS base samplers usually provides significant improvements for multimodal distributions while remains a safe choice for the easier, unimodal cases.

Acknowledgement

The authors would like to thank Firas Hamze for insightful discussions about different sampling methods and Tor Lattimore for his comments on an earlier version of the manuscript.

References

Ahn, Sungjin, Chen, Yutian, and Welling, Max. Distributed and adaptive darting monte carlo through regenerations. volume 31 of *Proceedings of Machine Learning Research*, pp. 108–116, 2013.

Andrieu, Christophe and Thoms, Johannes. A tutorial on adaptive MCMC. *Statistics and Computing*, 18(4): 343–373, December 2008. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-008-9110-y.

Andrieu, Christophe, De Freitas, Nando, Doucet, Arnaud, and Jordan, Michael I. An introduction to MCMC for machine learning. *Machine learning*, 50 (1-2):5–43, 2003.

Angelino, E., Johnson, M. J., and Adams, R. P. *Patterns of Scalable Bayesian Inference*. Now Publishers Inc., Hanover, MA, USA, 2016.

Atchadé, Yves F., Roberts, Gareth O., and Rosenthal, Jeffrey S. Towards optimal scaling of metropolis-coupled Markov chain Monte Carlo. *Statistics and Computing*, 21(4):555–568, October 2011. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-010-9192-1.

Auer, Peter, Cesa-Bianchi, Nicolo, and Fischer, Paul. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

Boucheron, Stéphane, Lugosi, Gábor, and Massart, Pascal. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

Brooks, Steve, Gelman, Andrew, Jones, Galin L., and Meng, Xiao-Li. *Handbook of markov chain monte carlo*. Chapman and Hall/CRC, 2011.

Bubeck, Sébastien and Cesa-Bianchi, Nicolò. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012. ISSN 1935-8237, 1935-8245. doi: 10.1561/2200000024.

Bédard, Mylène. Optimal acceptance rates for Metropolis algorithms: Moving beyond 0.234. *Stochastic Processes and their Applications*, 118(12):2198–2222, December 2008. ISSN 03044149. doi: 10.1016/j.spa.2007.12.005.

Carpentier, Alexandra, Munos, Remi, and Antos, András. Adaptive strategy for stratified Monte Carlo sampling. *Journal of Machine Learning Research*, 16:2231–2271, 2015.

Chwialkowski, Kacper, Strathmann, Heiko, and Gretton, Arthur. A Kernel Test of Goodness of Fit. *arXiv:1602.02964 [stat]*, February 2016. arXiv: 1602.02964.

Doucet, Arnaud, Defreitas, Nando, and Gordon, Neil. *An Introduction to Sequential Monte Carlo Methods*. Springer-Verlag,, New York, 2001.

Earl, David J. and Deem, Michael W. Parallel tempering: theory, applications, and new perspectives. *Physical Chemistry Chemical Physics (PCCP)*, 7 23: 3910–6, 2005.

Gorham, Jack, Duncan, Andrew B., Vollmer, Sebastian J., and Mackey, Lester. Measuring sample quality with diffusions. *arXiv preprint arXiv:1611.06972*, 2016.

Gorham, Jackson and Mackey, Lester. Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems*, pp. 226–234, 2015.

Gorham, Jackson and Mackey, Lester. Measuring sample quality with kernels. In *ICML*, pp. 1292–1301, 2017.

Haario, Heikki, Saksman, Eero, and Tamminen, Johanna. An Adaptive Metropolis Algorithm. *Bernoulli*, 7(2):223, April 2001. ISSN 13507265. doi: 10.2307/3318737.

Haario, Heikki, Laine, Marko, Mira, Antonietta, and Saksman, Eero. DRAM: Efficient adaptive MCMC. *Statistics and Computing*, 16(4):339–354, December 2006. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-006-9438-0.

Hero, A. O. and Michel, O. J. J. Asymptotic theory of greedy approximations to minimal k-point random graphs. *IEEE Transactions on Information Theory*, 45(6):1921–1938, 1999.

Hoffman, Matthew D. and Gelman, Andrew. The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

Ihler, A. T., Fisher, J. W., Moses, R. L., and Willsky, A. S. Nonparametric belief propagation for self-localization of sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):809–819, 2005.

Liu, Qiang and Lee, Jason D. Black-box importance sampling. *arXiv preprint arXiv:1610.05247*, 2016. URL `https://arxiv.org/abs/1610.05247`.

Liu, Qiang, Lee, Jason D., and Jordan, Michael I. A kernelized Stein discrepancy for goodness-of-fit tests. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

Mahendran, Nimalan, Wang, Ziyu, Hamze, Firas, and De Freitas, Nando. Adaptive MCMC with Bayesian Optimization. In *AISTATS*, volume 22, pp. 751–760, 2012.

Mbalawata, Isambi S., Särkkä, Simo, Vihola, Matti, and Haario, Heikki. Adaptive Metropolis algorithm using variational Bayesian adaptive Kalman filter. *Computational Statistics & Data Analysis*, 83:101–115, March 2015. ISSN 01679473. doi: 10.1016/j.csda.2014.10.006.

Moral, Pierre Del, Doucet, Arnaud, and Jasra, Ajay. Sequential monte carlo samplers. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 68(3):411–436, 2006.

Mykland, Per, Tierney, Luke, and Yu, Bin. Regeneration in markov chain samplers. *Journal of the American Statistical Association*, 90(429):233–241, 1995. ISSN 01621459.

Neiswanger, W., Wang, C., and Xing, E. P. Asymptotically exact, embarrassingly parallel mcmc. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI, pp. 623–632, Arlington, Virginia, United States, 2014. AUAI Press. ISBN 978-0-9749039-1-0.

Nemeth, Christopher and Sherlock, Christopher Gerrard. Merging mcmc subposteriors through gaussian-process approximations. *Bayesian Analysis*, 13(2):507–530, 3 2018. ISSN 1936-0975.

Neufeld, James, György, András, Schuurmans, Dale, and Szepesvári, Csaba. Adaptive Monte Carlo via bandit allocation. In *ICML*, pp. 1944–1952, 2014.

Owen, Art B. *Monte Carlo theory, methods and examples.* 2013. URL `http://statweb.stanford.edu/~owen/mc/`.

Pál, Dávid, Póczos, Barnabás, and Szepesvári, Csaba. Estimation of rényi entropy and mutual information based on generalized nearest-neighbor graphs. NIPS, pp. 1849–1857, 2010.

Pasarica, Cristian and Gelman, Andrew. Adaptively scaling the Metropolis algorithm using expected squared jumped distance. *Statistica Sinica*, pp. 343–364, 2010.

Roberts, Gareth O. and Rosenthal, Jeffrey S. Optimal Scaling for Various Metropolis-Hastings Algorithms. *Statistical Science*, 16(4):351–367, 2001. ISSN 0883-4237.

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

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. Bayes and big data: The consensus monte carlo algorithm. *International Journal of Management Science and Engineering Management*, 11:78–88, 2016.

Sejdinovic, Dino, Strathmann, Heiko, Garcia, Maria Lomeli, Andrieu, Christophe, and Gretton, Arthur. Kernel Adaptive Metropolis-Hastings. In *ICML*, pp. 1665–1673, 2014.

Shaloudegi, Kiarash and György, András. Adaptive MCMC via combining local samplers. *CoRR*, abs/1806.03816, 2018. URL `http://arxiv.org/abs/1806.03816`.

Wang, Ziyu, Mohamed, Shakir, and De Freitas, Nando. Adaptive Hamiltonian and Riemann manifold Monte Carlo samplers. In *International Conference on Machine Learning (ICML)*, pp. 1462–1470, 2013.

Yang, Jinyoung and Rosenthal, Jeffrey S. Automatically tuned general-purpose mcmc via new adaptive diagnostics. *Computational Statistics*, 32(1):315–348, March 2017.

Łatuszyński, Krzysztof, Roberts, Gareth O., and Rosenthal, Jeffrey S. Adaptive Gibbs samplers and related MCMC methods. *The Annals of Applied Probability*, 23(1):66–98, February 2013. ISSN 1050-5164. doi: 10.1214/11-AAP806.