Gradient-free Hamiltonian Monte Carlo with efficient kernel exponential families

Abstract

We propose *Kamiltonian Monte Carlo (KMC)*, a gradient-free adaptive MCMC algorithm based on Hamiltonian Monte Carlo (HMC). On target densities where HMC is unavailable due to intractable gradients, KMC adaptively learns the target's gradient structure by fitting an exponential family model in a Reproducing Kernel Hilbert Space (RKHS). Computational costs are reduced by two novel efficient approximations. While being asymptotically exact, KMC mimics HMC in terms of sampling efficiency and offers substantial mixing improvements in up to moderately high dimensions. We support our claims with experimental studies on both toy and real-world applications, including Approximate Bayesian Computation and exact-approximate MCMC.

1 Introduction

Estimating expectations using Markov Chain Monte Carlo (MCMC) is a fundamental approximate inference technique in Bayesian statistics. MCMC itself can be computationally demanding, and the expected estimation error depends directly on the correlation between successive points in the Markov chain. Therefore, MCMC efficiency can be achieved by taking large steps with high probability.

Hamiltonian Monte Carlo (HMC), [1], is an MCMC algorithm that improves efficiency by exploiting gradient information. It simulates particle movement along the contour lines of a dynamical system which is constructed from the target density. Projections of those trajectories cover wide parts of the target density and the probability of accepting a move along a trajectory is often close to one. Remarkably, this property is mostly invariant with input space dimension. Thus, HMC is often superior to random walk methods which have to decrease their step size at a much faster rate to maintain a reasonable acceptance probability with increasing dimension, [1, Sec. 4.4].

Unfortunately, for a large class of problems gradient information is not available. For example, in Pseudo-Marginal MCMC (PM-MCMC), [2][3], the posterior density does not have an analytic expression even up to a normalising constant. Therefore, it cannot be evaluated at any given point, only estimated, e.g. Bayesian Gaussian Process classification, [4]. A related context is MCMC for Approximate Bayesian Computation (ABC-MCMC), where a Bayesian posterior has to be approximated through repeated simulation from a likelihood model, [5][6]. In both cases, plain HMC cannot be applied, leaving random walk methods as the only mature alternative. There have been recent efforts to mimic HMC's behaviour using stochastic gradients [7][8]. However, stochastic gradient based HMC methods typically suffer from low acceptance rates or additional, hard to quantify bias, [9].

Random walk methods can be tuned by proposing local steps whose scaling matches the target density. For example, Adaptive Metropolis-Hastings (AMH) [10][11], is based on learning the global linear scaling of a target density from the history of the Markov chain trajectory. Yet, for densities with nonlinear support across components, this approach does not work very well. Recently, [12] introduced a Kernel Adaptive Metropolis-Hastings (KAMH) algorithm, whose proposals locally align with the target density. By learning target covariance in a Reproducing Kernel Hilbert Space (RKHS), KAMH achieves improved sampling efficiency on such targets.

In this paper, we extend the idea of using kernel methods for efficient proposal distributions. However, rather than *locally* smoothing the target density, we estimate its gradients *globally*. More precisely, we fit an (unnormalised) infinite dimensional exponential family model in a RKHS via score matching, following [13][14]. This is a recently proposed non-parametric method to model the log unnormalised target density as a RKHS function, and has been shown to approximate a rich class of density functions arbitrarily well. More importantly, the method has been empirically observed to be relatively robust to increasing dimensions – in sharp contrast to classical kernel density estimation, [15, Sec. 6.5].

To make the algorithm more efficient in the context of MCMC chains of growing length, we develop two novel approximations to the infinite dimensional exponential family model. The first approximation, *score matching lite*, is based on projecting the solution to a lower dimensional, yet growing, subspace in the RKHS. KMC with score matching lite (*KMC lite*) is geometrically ergodic on the same class of targets as standard random walks. The second approximation uses a finite dimensional feature space (*KMC finite*), combined with the random Fourier features framework of [16]. It results in an extremely efficient on-line style estimator that allows to use all of the Markov chain history – at the cost of decreased efficiency when initialised in the tails. A choice between KMC lite and KMC finite will ultimately depend on the ability to initialise the sampler within high-density regions of the target density, where the approaches could also be combined.

Experiments show that KMC inherits the efficiency of HMC and therefore mixes significantly better than any random walk based method on a number of target densities, including synthetic examples, PM-MCMC, and ABC-MCMC.

Paper outline: In Section 2, we place our work in the context of previous work and cover HMC basics. Section 3 introduces Hamiltonian dynamics induced by kernel exponential families and Section 4 contains two novel approximate estimators. We introduce the KMC algorithm in Section 3 and describe a number of experiments in Section 6.

2 Background & previous work

Let $\mathcal{X} = \mathbb{R}^d$ be the domain of interest, and denote the unnormalised target density on $\mathcal{X}$ by π. We are interested in constructing a Markov chain $x_1 \to x_2 \to \dots$ such that $\lim_{t\to\infty} x_t \sim \pi$. By running the Markov chain for a long time T, we can consistently approximate any expectation w.r.t π. Markov chains are constructed using the Metropolis-Hastings algorithm, which at the current state x_t draws a point from a proposal mechanism

$$x^* \sim Q(\cdot|x_t), \tag{1}$$

and sets $x_{t+1} \leftarrow x^*$ with probability $\min(1, [\pi(x^*)Q(x_t|x^*)]/[\pi(x_t)Q(x_t|x^*)])$, and $x_{t+1} \leftarrow x_t$ otherwise. In this paper, we generally assume that π is intractable[1], i.e. that we do can neither evaluate $\pi(x)$ nor $\nabla \log \pi(x)$ for any x (throughout the paper ∇ denotes the gradient operator wrt. to x), but can compute unbiased estimates of $\pi(x)$. Replacing $\pi(x)$ with an unbiased estimator results in PM-MCMC, [2][3], which asymptotically remains exact (*exact-approximate inference)*.

(Kernel) Adaptive Metropolis-Hastings In the absence of $\nabla \log \pi$, the usual choice of q in (1) is a random walk, i.e. a Gaussian $Q(\cdot|x_t) = \mathcal{N}(\cdot|x_t, \Sigma_t)$. A popular choice of the step scaling is $\Sigma_t \propto I$. When the (unknown) scale of the target density is not uniform across dimensions, or their are strong correlations, the original AMH algorithm [10][11] improves mixing via adaptively learning global covariance structure of π from the history of the Markov chain. For cases where local scaling does not match the global covariance structure of π, KAMH [12] improves mixing by learning the target covariance structure in a RKHS. When local steps are taken using this RKHS covariance, this amounts to a proposal that matches the local covariance of π around the current state x_t – without requiring access to $\nabla \log \pi$.

Hamiltonian Monte Carlo Hamiltonian Monte Carlo (HMC) utilises deterministic, measure-preserving maps to generate efficient Markov transitions, [1], [17]. Starting from the negative

[1]Unavailable due to analytic intractability, as opposed to computationally expensive in the Big Data context.

log unnormalised target density, referred to as *potential energy*, $U(q) \propto -\log \pi(q)$, we introduce an auxiliary *momentum* variable $p \sim \exp(-K(p))$. The joint distribution for (p, q) is then proportional to $\exp(-H(p, q))$, where $H(p, q) := K(p) + U(q)$ is called the *Hamiltonian*. $H(p, q)$ defines a *Hamiltonian flow*, parametrised by a trajectory length $t \in \mathbb{R}$, which is a map $\phi_t^H : (p, q) \mapsto (p^*, q^*), \forall t \in \mathbb{R}$ for which $H(p^*, q^*) = H(p, q)$ for any t. This allows construction of π-invariant Markov chains: for a chain at state $q = x_t$, we repeatedly 1) re-sample $p' \sim \exp(-K(\cdot))$, and then 2) apply the Hamiltonian flow for time t, giving $(p^*, q^*) = \phi_t^H(p', q)$. The flow can be generated by the *Hamiltonian operator*

$$\hat{H} = \frac{\partial K}{\partial p}\frac{\partial}{\partial q} - \frac{\partial U}{\partial q}\frac{\partial}{\partial p} =: \hat{K} + \hat{U}. \tag{2}$$

In practice, $\hat{H}$ is usually unavailable and we need to resort to approximations. Here we limit ourselves to the leap-frog integrator, see [1] for details. To correct for discretisation error, a Metropolis acceptance procedure can be applied: starting from a (p', q), the end-point of the approximate trajectory is accepted with probability $\min[1, \exp(-H(p^* q^*) + H(q, p'))]$. In practice HMC is often able to propose distant, uncorrelated moves with a high acceptance probability.

Intractable densities In many cases the gradient of $\log \pi(q) = \text{const} - U(q)$ is unavailable, often leaving random-walk based methods as the state-of-the-art [12][11]. It is well known that overcoming random-walk behaviour can result in significantly more efficient sampling, [1].

3 Kernel induced Hamiltonian dynamics

Kamiltonian Monte Carlo is based on the idea of replacing the potential energy operator $\hat{U}$ in (2) by a kernel induced surrogate computed from the history of the Markov chain. As we will see, the surrogate does not require gradients of the log-target density. It induces a kernel Hamiltonian flow, which can be numerically integrated using standard leap-frog integration. As the discretisation error in HMC, any deviation of the kernel induced flow the true flow is corrected for via the Metropolis acceptance procedure. Consequently, the stationary distribution of the Markov chain will *remain correct*[2].

Infinite exponential families in RKHS We construct a kernel induced potential energy surrogate whose gradients match the gradients of the true potential energy $\hat{U}$ in (2), or $\log \pi(x)$ – without accessing it directly. To that end, we fit an infinite dimensional exponential family model [13] of the form

$$\exp\left(\langle f, k(x, \cdot)\rangle_{\mathcal{H}} - A(f)\right). \tag{3}$$

Here, $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) of real valued functions on $\mathcal{X}$. It has a uniquely associated symmetric, positive definite function (*kernel*) $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which satisfies $f(x) = \langle f, k(x, \cdot)\rangle$ for any $f \in \mathcal{H}$, [18]. The canonical feature map $k(\cdot, x) \in \mathcal{H}$ here takes the role of the *sufficient statistics* and $f \in \mathcal{H}$ are the *natural parameters*. $A(f) := \log \int_{\mathcal{X}} \exp(\langle f, k(x, \cdot)\rangle_{\mathcal{H}})dx$ is the cumulant generating function. (3) defines a very rich class of probability distributions and a broad class of densities (for example continuous densities defined on compact domains) can be approximated arbitrarily well. It was shown in [13] that it is possible to consistently fit an *unnormalised* version of (3) by directly minimising the expected gradient mismatch between the infinite dimensional exponential family model and the true (observed through samples) target density – score matching [14]. This technique avoids the problem of dealing with the intractable cumulant generating function $A(f)$ in an elegant way and reduces the problem to solving a linear system. More importantly, it makes the model relatively robust to increasing dimensions, as opposed to classic kernel density estimation, which was confirmed in experiments. [13] also established convergences rates in Kullback-Leibler divergence, Hellinger and total-variation distances. We will come back to the topic of estimation and develop two efficient approximate empirical estimators later. For now, assume access to an $f \in \mathcal{H}$ such that $\nabla f(x) \approx \nabla \log \pi(x)$.

[2]As usual when constructing adaptive MCMC algorithms, we will need to take care when generating proposals based on the history of the Markov chain.

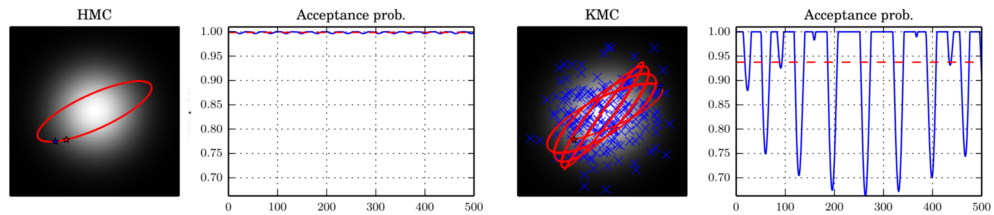

Figure 1: Hamiltonian trajectories on 2-dimensional standard Gaussian. End points of such trajectories (blue stars) form the proposal of HMC-like algorithms. **Left:** Plain Hamiltonian trajectories oscillate on a stable orbit and acceptance probability is close to one. **Right:** Kernel induced trajectories and acceptance probabilities on an estimated energy function.

Kernel induced Hamiltonian flow We define a kernel induced Hamiltonian operator $\hat{H}_k = \hat{K} + \hat{U}_k$, where $\hat{K}$ is defined as in (2), and we replaced the potential energy U with the kernel surrogate $U_k = f$. This induces a kernel induced potential energy operator $\hat{U}_k = \frac{\partial U_k}{\partial p}\frac{\partial}{\partial q}$ and corresponding kernel Hamiltonian flow. It is clear that the kernel induced potential energy operator results in different trajectories than those induced by the true operator (2). However, any bias on the resulting Markov chain, in addition to discretisation error from the leap-frog integrator, is naturally corrected for in the Metropolis step. We accept an end-point $\phi_t^{\hat{H}}(p', q)$ of a trajectory along the *kernel induced* flow with probability

$$\min\left[1, \exp\left(H\left(\phi_t^{H_k}(p', q)\right) - H(p', q)\right)\right], \tag{4}$$

where $H\left(\phi_t^{\hat{H}}(p', q)\right)$ denotes evaluation of the *true* Hamiltonian at $\phi_t^{H_k}(p', q)$. Any deviations of the kernel induced flow from the true flow results in a decreased acceptance probability (4). We therefore need to control the approximation quality of the kernel induced potential energy to maintain high acceptance probability in practice. See Figure 1 for an illustrative example.

4 Two efficient estimators for infinite exponential families in RKHS

We now come back to the topic of estimating the infinite dimensional exponential family model (3) from data. The original estimator of (3) has computational costs of $\mathcal{O}(n^3 d^3)$ when applied to n samples of dimension d, [13]. As this is limiting in the adaptive MCMC context where the estimator has to be updated on a regular basis, we propose two efficient approximations, with orthogonal strengths and weaknesses. Both the original estimator for (3) and our approximations are based on score matching, see Appendix A.1 for a brief review.

4.1 Infinite exponential families lite

The original estimator of f in (3) takes a dual form and lies in span $\{k(x_{i\ell}, \cdot)\}_{i,\ell=1}^{n,d}$, [13, Thm. 4]. The update of the proposal at the iteration t of MCMC would require an inversion of a $td \times td$ matrix. This is clearly prohibitive if one is to run even a moderately large number of iterations of a Markov chain. Following [12], we take simple approach to avoid prohibitive computational cost in t: forming a proposal using a random sub-sample of a fixed size n from the Markov chain history, $\mathbf{z} = \{z_i\}_{i=1}^n \subseteq \{x_i\}_{i=1}^t$. In addition, in order to reduce excessive computational costs arising from large d, we develop an approximation to the full dual solution in [13] by expressing the solution in terms of span $(\{k(z_i, \cdot)\}_{i=1}^n)$. This growing subspace of RKHS basis functions[3] does not grow in d but only in n. That is, we assume that the log unnormalised density of infinite dimensional exponential family model in (3) takes the dual form

$$f(x) = \sum_{i=1}^{n} \alpha_i k(z_i, x), \tag{5}$$

[3]Note that any sub-space could be used as long as the support of π is covered.

where $\alpha \in \mathbb{R}^n$ are real valued parameters that are obtained via minimising the empirical score matching objective (10) in Appendix A.1. The estimator is summarised in the following proposition (proof is given in Appendix A.2).

Proposition 1. *Given a set of samples* $\mathbf{z} = \{z_i\}_{i=1}^n$ *and assuming* $f(x) = \sum_{i=1}^n \alpha_i k(z_i, x)$ *for the Gaussian kernel of the form* $k(x, y) = \exp\left(-\sigma^{-1}\|x - y\|_2^2\right)$, *and* $\lambda > 0$, *the unique minimiser of the* $\lambda\|f\|_{\mathcal{H}}^2$*-regularised empirical score matching objective* (10) *is given by*

$$\hat{\alpha}_\lambda = -\frac{\sigma}{2}(C + \lambda I)^{-1} b, \tag{6}$$

where $b \in \mathbb{R}^n$ *and* $C \in \mathbb{R}^{n \times n}$ *with*

$$b = \sum_{\ell=1}^d \left(\frac{2}{\sigma}(Ks_\ell + D_{s_\ell}K\mathbf{1} - 2D_{x_\ell}Kx_\ell) - K\mathbf{1}\right) \text{ and } C = \sum_{\ell=1}^d \left[D_{x_\ell}K - KD_{x_\ell}\right]\left[KD_{x_\ell} - D_{x_\ell}K\right],$$

and with entry-wise products $s_\ell := x_\ell \odot x_\ell$ *and* $D_x :=$ *diag*(x).

The estimator has costs of $\mathcal{O}(n^3 + dn^2)$ computation (for both computing C, b and inverting C) and $\mathcal{O}(n^2)$ storage for a fixed random chain history sub-sample size n. This can be further reduced to *linear* computation and storage via low-rank approximations to the kernel matrix and conjugate gradient methods, see Appendix A.2.

Gradients of the estimated log-density are given as $\nabla f(x) = \sum_{i=1}^n \alpha_i \nabla k(x, x_i)$, i.e. they simply require to evaluate gradients of the kernel function.

4.2 Exponential families in finite feature spaces

Instead of fitting an infinite-dimensional model on a subset of the available data, the second estimator is based on fitting a finite dimensional approximation using *all* available data $\{x_i\}_{i=1}^t$ – in *primal* form. As we will see, updating the estimator when a new data point arrives can be done in an online fashion.

Define an m-dimensional approximate[4] feature space $\mathcal{H}_m = \mathbb{R}^m$, denote by $\phi_x \in \mathcal{H}^m$ the embedding of a point $x \in \mathcal{X} = \mathbb{R}$ into $\mathcal{H}_m = \mathbb{R}^m$. Assume that the embedding approximates the kernel function as a finite rank expansion $k(x, y) \approx \phi_x^\top \phi_y$. The log unnormalised density of the infinite model (3) can be approximated in this feature space as

$$f(x) = \langle \theta, \phi_x \rangle_{\mathcal{H}_m} = \theta^\top \phi_x \tag{7}$$

In order to fit $\theta \in \mathbb{R}^m$, we again minimise the score matching objective in (10) in Appendix A.1. The following proposition summarises the estimator.

Proposition 2. *Given a set of samples* $\mathbf{x} = \{x_i\}_{i=1}^t$ *and assuming* $f(x) = \theta^\top \phi_x$ *for a finite dimensional feature embedding* $x \mapsto \phi_x \in \mathbb{R}^m$, *and* $\lambda > 0$, *the unique minimiser of the* $\lambda\|\theta\|_2^2$*-regularised empirical score matching objective* (10) *is given by*

$$\hat{\theta}_\lambda := (C + \lambda I)^{-1} b, \tag{8}$$

where

$$b := -\frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \ddot{\phi}_{x_i}^\ell \in \mathbb{R}^m, \qquad C := \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left(\dot{\phi}_{x_i}^\ell \left(\dot{\phi}_{x_i}^\ell\right)^T\right) \in \mathbb{R}^{m \times m},$$

with $\dot{\phi}_x^\ell := \frac{\partial}{\partial x_\ell}\phi_x$ *and* $\ddot{\phi}_x^\ell := \frac{\partial^2}{\partial x_\ell^2}\phi_x$.

An example feature embedding based on random Fourier features, [16], and a standard Gaussian kernel, $\phi_x = \sqrt{\frac{2}{m}}\left[\cos(\omega_1^T x + u_1), \ldots, \cos(\omega_m^T x + u_m)\right]$, with $\omega_i \sim \mathcal{N}(\omega)$ and $u_i \sim \texttt{Uniform}[0, 2\pi]$. Proof and details can be found in Appendix A.3. The estimator has a one-off cost of $\mathcal{O}(tm^2 + m^3)$ computation and $\mathcal{O}(m^2)$ storage. However, given that we have computed a solution based on the Markov chain history $\{x_i\}_{i=1}^t$, it is straight-forward to on-line update C, b and the solution $\hat{\theta}_\lambda$ after a new point arrived x_{t+1}. This is achieved via storing running averages and low-rank updates of

[4]We deliberately don't state the form of the approximation yet, but will give details later.

Algorithm 1 Kamiltonian Monte Carlo – pseudo-code

Input: Target (estimator) π, adaptation schedule a_t, HMC parameters,
Size of basis m or sub-sample size n.
At iteration $t+1$, current state x_t, history $\{x_i\}_{i=1}^t$, *with probability* a_t,

KMC lite:

1. Update sub-sample $\mathbf{z} \subseteq \{x_i\}_{i=1}^t$
2. Re-compute C, b from Prop. 1
3. Solve $\hat{\alpha}_\lambda = -\frac{\sigma}{2}(C+\lambda I)^{-1}b$
4. $\nabla f(x) \leftarrow \sum_{i=1}^n \alpha_i \nabla k(x, z_i)$

KMC finite:

1. Update to C, b from Prop. 2
2. Perform rank-d update to C^{-1}
3. Update $\hat{\theta}_\lambda = (C+\lambda I)^{-1}b$
4. $\nabla f(x) \leftarrow [\nabla \phi_x]^\top \hat{\theta}$

5. Propose (p', x^*) with kernel induced Hamiltonian flow, using $\nabla_x U = \nabla_x f$
6. Perform Metropolis step using π, $\quad x_{t+1} \leftarrow x^*$ w.p. (4) and $x_{t+1} \leftarrow x_t$ otherwise

matrix inversions and allows to update the estimator in $\mathcal{O}(m^2)$ computation and storage, which is independent of t. Further details are given in Appendix A.3.

Gradients of the estimated log-density are given as $\nabla f(x) = [\nabla \phi_x]^\top \hat{\theta}$, i.e. they require evaluation of the gradient of the feature space embedding.

5 Kamiltonian Monte Carlo

Constructing a kernel induced Hamiltonian flow as in Section 3 from the gradients of the infinite dimensional exponential family model (3) and approximate estimators 5,(7) we arrive at a gradient free, adaptive kernel Hamiltonian Monte Carlo algorithm: *Kamiltonian Monte Carlo,* see Algorithm 1.

Computational efficiency vs. geometric ergodicity KMC with the finite estimator (7) allows for on-line updates using the *full* Markov chain history and therefore is a more elegant solution than sub-sampling. However, due to the parametric nature of this approximate model, the tails of the estimator are not guaranteed to decay. For example, the random Fourier features embedding in below Proposition 2 contains periodic cosine functions and therefore oscillates in the tails – resulting in a catastrophic acceptance probability. As we will demonstrate in the experiments, this problem does not appear when initialised in high-density regions of the posterior. In situations where this information about the target density is unknown, we suggest to use the lite estimator (5), at least for burn-in. KMC lite falls back to a random walk in the tails as gradients of the estimator (5) decay. Consequently, it inherits convergence properties of random walk samplers, as formalised in the following result, which comes at the expense of increased computational costs and having to sub-sample the chain history. We give intuition below, see Appendix A.4 a proof.

Proposition 3. *Assume $d = 1$, $\pi(x)$ is log-concave in the tails, and regularity conditions of [19, Thm 2.2] (implying π-irreducibility and smallness of compact sets), MCMC adaptation stops after a fixed time, and a fixed number L of ϵ-leapfrog steps. If $\limsup_{\|x\|_2 \to \infty} \nabla f(x) = 0$ and $\exists M : \forall x : \|\nabla f(x)\|_2 \leq M$, then KMC lite is geometrically ergodic from π-almost any starting point.*

Intuition Define $c(x) := \epsilon^2 \sum_{i=0}^{L-1} \nabla f(x_{i\epsilon})/2$ and $d(x_0) := \epsilon(\nabla f(x_0) + \nabla f(x_{L\epsilon}))/2 + \epsilon \sum_{i=1}^{L-1} \nabla f(x_{i\epsilon})$. At x_t, the marginal KMC proposal on position space looks like $x^*(p') = x_t + c(x_t) + N\epsilon p'$ where wlog. $p' \sim \mathcal{N}(0, I)$. This is accepted with probability $\text{acc}(x_t, x^*(p')) = \min\left(1, \frac{\pi(x^*(p'))}{\pi(x_t)} \exp\left(-\frac{1}{2}\left[p' d(x_t) + d(x_t)^2\right]\right)\right)$. From the distribution of p', we have $c(x_t) \xrightarrow{p} 0$ as $\|x_t\|_2 \to \infty$, and similarly for $d(x_t)$. So for large x_t, we have $x^* \approx x_t + L\epsilon p'$ and $\text{acc}(x_t, x^*) \approx \pi(x^*)/\pi(x_t)$, meaning in the tails the chain will behave as a Random Walk Metropolis. So KMC lite is geometrically ergodic whenever the Random Walk Metropolis is. Generalisations to $d \geq 2$ require an additional curvature condition of [19] but are out of the scope of this paper.

Vanishing adaptation MCMC algorithms that use the history of the Markov chain for constructing proposals might not be asymptotically correctness. We follow KAMH [12] and the idea of

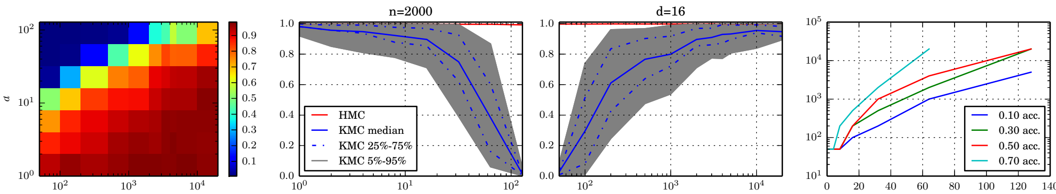

Figure 2: Acceptance probability of kernel induced Hamiltonian flow in high dimensions. **Left:** As a function of $n = m$ and d. **Middle:** Slices through left plot with error bars as a function in d, and dimension $n = m$. **Right:** Data needed to reach a certain acceptance probability as a function in d.

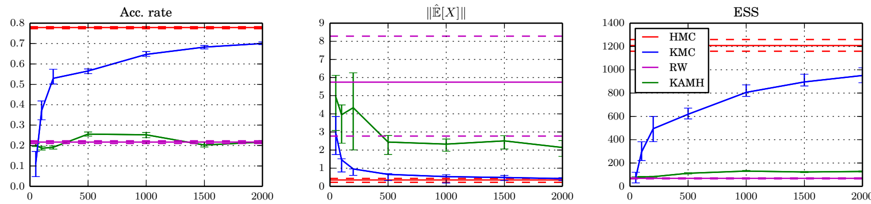

Figure 3: Results for 8-dimensional synthetic Banana. As the number of seen data increases, KMC performs close to HMC – outperforming KAMH and RW. 80% error bars over 30 runs.

"vanishing adaptation", [11], to avoid such biases. Let $\{a_t\}_{i=0}^{\infty}$ be a schedule of decaying probabilities such that $\lim_{t\to\infty} a_t = 0$ and $\sum_{t=0}^{\infty} a_t = \infty$ and update the density gradient estimate according to that schedule in Algorithm 1. Intuitively, adaptation becomes less likely as the MCMC chain is progressing, but never fully stops – while sharing asymptotic convergence from adaptation that stops at a fixed point. See, [20, Theorem 1]. Note that Proposition 3 is a stronger statement about the convergence rate.

6 Experiments

We start by quantifying KMC finite performance using the finite estimator on synthetic targets. We emphasise that these results can be reproduced with the lite version.

KMC finite: Stability of trajectories in high dimensions In order to quantify (finite) KMC's efficiency in growing dimensions, we study average acceptance probabilities purely over trajectories from the origin along the kernel induced Hamiltonian flow (no MCMC yet) on a standard Gaussian target. Figure 2 shows the average acceptance over 100 independent trials as a function of (i) dimension and (ii) number of basis functions and data $m = n$. In dimensions up to $d \approx 100$, we are able to obtain acceptance probabilities comparable to plain HMC with the finite estimator fitted in a few seconds on a laptop computer.

KMC finite: HMC-like mixing on a synthetic example We now show that KMC is able to match performance of HMC as it sees more data. We compare KMC, HMC, an isotropic random walk (RW), and KAMH to sample from the 8-dimensional non-linear banana-shaped example from [12], [10]. To only quantify mixing, both KMC and KAMH use the same set of fixed burn-in samples. We quantify performance on estimating the target's mean, which is exactly $\mathbf{0}$. We tune the scaling of KAMH and RW to achieve roughly 23% acceptance probability. We set HMC parameters to achieve 80% acceptance probability and then match those of KMC. We run all samplers for 2000+200 iterations from a randomly chosen start point (within a higher density region), discard the burn-in and compute average acceptance rate, the norm of the empirical mean $\|\hat{\mathbb{E}}[x]\|$, and average effective sample size (ESS) across dimensions. For KAMH and KMC, we repeat the experiment for an increasing number of samples/basis function $m = n$. Figure 3 shows results as a function of data used. KMC clearly outperforms RW and KAMH, and almost matches performance of HMC.

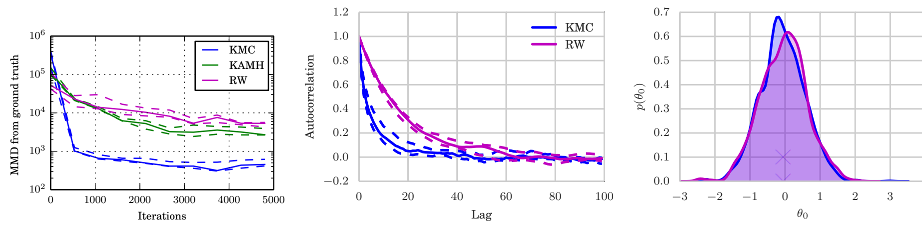

Figure 4: **Left:** Results for 9-dimensional marginal posterior over length scales of a GPC model applied to the UCI Glass dataset. The plots shows convergence of all mixed moments up to order 3 to a previously generated ground-truth (lower MMD is better). **Middle/right:** ABC-MCMC autocorrelation and marginal (θ_0) posterior for a 10-dimensional skew normal likelihood.

KMC lite: Pseudo-Marginal MCMC for GP Classification on real world data Following [12, Section 5.1], we next apply KMC to sample the marginal posterior over hyper-parameters of a Gaussian Process Classification (GPC) model, induced by the UCI Glass dataset, [21]. Note that HMC is *unavailable* for this problem. Our experimental protocol mostly follows [12, Section 5.1], but uses only 200+5000 MCMC iterations. We compare convergence in terms of all mixed moments of order up to 3 to a set of benchmark samples (MMD, [22], lower is better). KMC uses between 1 and 10 leapfrog steps of a size chosen uniformly in $[0.01, 0.1]$, a standard Gaussian momentum, and a cross-validation tuned kernel width (tuned after burn-in), achieving 45% acceptance. We did not extensively tune the HMC parameters of KMC as the current settings were sufficient here. Both KMC and KAMH use 1000 samples from the chain history. Figure 4 (left) shows that KMC clearly outperforms both RW and the earlier state-of-the-art KAMH. These results are backed by the average ESS (not plotted), which is around 800 for KMC and is around 90 and 60 for KAMH and RW respectively. All samplers took roughly 1h of computing time – most time is spent on estimating the marginal likelihood, which is in line with [12].

KMC lite: Reduced simulations and no additional bias in ABC We now apply KMC in the context of Approximate Bayesian Computation (ABC), which often is employed when the data likelihood is intractable but can be simulated from, see e.g. [6]. ABC-MCMC [5], targets an approximate posterior, by constructing an unbiased Monte Carlo estimator of the approximate likelihood. As each such evaluation requires expensive simulations from the likelihood, the goal of all ABC methods is to reduce the number of such simulations. [8] recently proposed Hamiltonian ABC: combining the synthetic likelihood approach, [23], with stochastic finite differences gradients. We remark that this requires to simulate from the likelihood in *every* leapfrog step, and that the additional bias from Gaussian likelihood approximation can be problematic. In contrast, KMC does not require simulations to propose but rather invests into an accept/reject step (4) that ensures convergence to the *original* ABC target. On a 10-dimensional skw-normal distribution $p(y|\theta) = 2\mathcal{N}(\theta, I)\,\Phi(\langle\alpha, y\rangle)$ with $\alpha = 10$, Figure 4 (right) compares performance of RW, HABC, and KMC. KMC mixes as well as HABC compared to RW – without suffering from additional bias. We leave a systematic study for future work.

7 Discussion

We have introduced KMC, a kernel-based gradient free adaptive MCMC algorithm that mimics HMC's behaviour via estimating target gradients in a RKHS from the MCMC chain history. In experiments, KMC outperforms random walk based sampling methods in up to moderately high dimensions ($d \leq 100$), including recent kernel-based approaches [12]. KMC is in particular useful when gradients of the target density are unavailable, such as PM-MCMC or ABC-MCMC, as HMC is not an option there. We have proposed two efficient empirical estimators with orthogonal strengths and weaknesses and given experimental evidence of the robustness of both.

Future work includes establishing theoretical consistency and uniform convergence rates for the empirical estimators, and a thorough experimen tal study in the ABC-MCMC context where we see KMC's biggest potential. We will publish our code to reproduce all experimental results.

References

[1] R.M. Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2, 2011.

[2] M.A. Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160, 2003.

[3] C. Andrieu and G.O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, April 2009.

[4] M. Filippone and M. Girolami. Pseudo-marginal Bayesian inference for Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[5] Paul Marjoram, John Molitor, Vincent Plagnol, and Simon Tavaré. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.

[6] S.A. Sisson and Y. Fan. *Handbook of Markov chain Monte Carlo*.

[7] T. Chen, E. Fox, and C. Guestrin. Stochastic Gradient Hamiltonian Monte Carlo. In *ICML*, pages 1683–1691, 2014.

[8] E. Meeds, R. Leenders, and M. Welling. Hamiltonian ABC. In *UAI*, 2015.

[9] M. Betancourt. The Fundamental Incompatibility of Hamiltonian Monte Carlo and Data Subsampling. *arXiv preprint*, 2015.

[10] H. Haario, E. Saksman, and J. Tamminen. Adaptive proposal distribution for random walk Metropolis algorithm. *Computational Statistics*, 14(3):375–395, 1999.

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

[12] D. Sejdinovic, H. Strathmann, M. Lomeli, C. Andrieu, and A. Gretton. Kernel Adaptive Metropolis-Hastings. In *ICML*, 2014.

[13] B. Sriperumbudur, K. Fukumizu, R. Kumar, A. Gretton, and A. Hyvärinen. Density Estimation in Infinite Dimensional Exponential Families. *arXiv preprint*, 2014.

[14] A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *JMLR*, 6:695–709, 2005.

[15] Larry Wasserman. *All of nonparametric statistics*. Springer, 2006.

[16] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.

[17] M. Betancourt, S. Byrne, and M. Girolami. Optimizing The Integrator Step Size for Hamiltonian Monte Carlo. *arXiv preprint*, 2015.

[18] A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer, 2004.

[19] G.O. Roberts and R.L. Tweedie. Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms. *Biometrika*, 83(1):95–110, 1996.

[20] G.O. Roberts and J.S. Rosenthal. Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probabability*, 44(2):458–475, 03 2007.

[21] K. Bache and M. Lichman. UCI Machine Learning Repository, 2013.

[22] A. Gretton, K. Borgwardt, B. Schölkopf, A. J. Smola, and M. Rasch. A kernel two-sample test. *JMLR*, 2012.

[23] S. N. Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102–1104, 08 2010.

[24] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[25] J.R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie-Mellon University. Department of Computer Science, 1994. `http://goo.gl/IE7lt1`.

[26] Q. Le, T. Sarlós, and A. Smola. Fastfood–approximating kernel expansions in loglinear time. In *ICML*, 2013.

[27] P.E Gill, G.H. Golub, W. Murray, and M.A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.

[28] Matthias Seeger. Low rank updates for the cholesky decomposition. Technical report, University of California at Berkeley. Department of EECS, 2004. `http://goo.gl/f99Onw`.

[29] K.L. Mengersen and R.L. Tweedie. Rates of convergence of the Hastings and Metropolis algorithms. *The Annals of Statistics*, 24(1):101–121, 1996.

A Proofs & details

A.1 Score matching

The presentation follows [14]. We model the log unnormalised probability $\log \pi(x)$ with a parametric model of the form

$$\log \tilde{\pi}_Z(x; f) := \log \tilde{\pi}(x; f) - \log Z(f), \tag{9}$$

where f is a collection of parameters of yet unspecified dimension (c.f. natural parameters f of (3)), and $Z(f)$ is an unknown normalising constant. We aim to approximate π by $\tilde{\pi}$, i.e., to find $\hat{f}$ from a set of fixed n i.i.d. samples[5]. $\{x_i\}_{i=1}^n \sim \pi$, such that $\pi(x) \approx \tilde{\pi}(x; \hat{f}) \times$ const. From [14, Eq. 2], the criterion being optimised is the expected squared distance between score functions,

$$J(f) = \frac{1}{2} \int_{\mathcal{X}} \pi(x) \left\| \psi(x; f) - \psi_\pi(x) \right\|_2^2 dx,$$

where

$$\tilde{\psi}(x; \theta) = \nabla \log \tilde{\pi}_Z(x; f) = \nabla \log \tilde{\pi}(x; f),$$

and $\psi(x)$ is the derivative wrt x of the unknown true density $\pi(x)$. As shown in [14, Theorem 1], the *Fisher score* takes the form

$$\hat{J}(f) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\ell=1}^{d} \left[\partial_\ell \psi_\ell(x_i; f) + \frac{1}{2} \psi_\ell^2(x_i; f) \right], \tag{10}$$

where

$$\psi_\ell(x; f) = \frac{\partial \log \tilde{\pi}(x; f)}{\partial x_\ell} \qquad \text{and} \qquad \partial_\ell \psi_\ell(x; f) = \frac{\partial^2 \log \tilde{\pi}(x; f)}{\partial x_\ell^2}. \tag{11}$$

Both proposed approximate estimators of the infinite dimensional exponential family model (3) from [13] are based on minimising (10) using approximate version of the scores (11).

A.2 Lite estimator

Proof of Proposition 1

We assume the model log-density (9) takes the dual form in Proposition 1, then directly implement score functions (11) and derive a matrix expression of the empirical score matching objective (10), which can be minimised with a linear solve.

Proof. As assumed the log unnormalised density takes the form

$$f(x) = \sum_{i=1}^{m} \alpha_i k(x_i, \xi)$$

where $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the Gaussian kernel in the form

$$k(x_i, \xi) = \exp\left(-\sigma^{-1} \|x_i - x\|^2\right) = \exp\left(-\frac{1}{\sigma} \sum_{\ell=1}^{d} (x_{i\ell} - x_\ell)^2\right).$$

The score functions from 11 are then given by

$$\psi_\ell(x; \alpha) = \frac{2}{\sigma} \sum_{i=1}^{n} \alpha_i (x_{i\ell} - x_\ell) \exp\left(-\frac{\|x_i - x\|^2}{\sigma}\right)$$

and

$$\begin{aligned} \partial_\ell \psi_\ell(x; \alpha) &= \frac{-2}{\sigma} \sum_{i=1}^{n} \alpha_i \exp\left(-\frac{\|x_i - x\|^2}{\sigma}\right) + \left(\frac{2}{\sigma}\right)^2 \sum_{i=1}^{m} \alpha_i (x_{i\ell} - x_\ell)^2 \exp\left(-\frac{\|x_i - x\|^2}{\sigma}\right) \\ &= \frac{2}{\sigma} \sum_{i=1}^{n} \alpha_i \exp\left(-\frac{\|x_i - x\|^2}{\sigma}\right) \left[-1 + \frac{2}{\sigma} (x_{i\ell} - x_\ell)^2\right]. \end{aligned}$$

[5] We assume a fixed sample set here but will use both the full Markov chain history $\{x_i\}_{i=1}^t$ and a subsample of size n later.

Substituting this into (10) yields

$$
\begin{aligned}
J(\alpha) &= \frac{1}{m}\sum_{i=1}^{n}\sum_{\ell=1}^{d}\left[\partial_\ell \psi_\ell(x_i;\alpha) + \frac{1}{2}\psi_\ell(x_i;\alpha)^2\right] \\
&= \frac{2}{m\sigma}\sum_{\ell=1}^{d}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)\left[-1 + \frac{2}{\sigma}(x_{i\ell} - x_{j\ell})^2\right] \\
&\quad + \frac{2}{m\sigma^2}\sum_{\ell=1}^{d}\sum_{i=1}^{n}\left[\sum_{j=1}^{n}\alpha_j (x_{j\ell} - x_{i\ell})\exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)\right]^2.
\end{aligned}
$$

We now rewrite $J(\alpha)$ in matrix form. The expression for the term $J(\alpha)$ being optimised is the sum of two terms.

First term:

$$
\sum_{\ell=1}^{d}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)\left[-1 + \frac{2}{\sigma}(x_{i\ell} - x_{j\ell})^2\right]
$$

We only need to compute

$$
\begin{aligned}
&\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)(x_{i\ell} - x_{j\ell})^2 \\
=&\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)\left(x_{i\ell}^2 + x_{j\ell}^2 - 2x_{i\ell}x_{j\ell}\right).
\end{aligned}
$$

Define

$$
x_\ell := \begin{bmatrix} x_{1\ell} & \ldots & x_{m\ell} \end{bmatrix}^\top.
$$

The final term may be computed with the right ordering of operations,

$$
-2(\alpha \odot x_\ell)^\top K x_\ell,
$$

where $\alpha \odot x_\ell$ is the entry-wise product. The remaining terms are sums with constant row or column terms, define $s_\ell := x_\ell \odot x_\ell$ with components $s_{i\ell} = x_{i\ell}^2$. Then

$$
\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i k_{ij} s_{j\ell} = \alpha^\top K s_\ell.
$$

Likewise

$$
\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i x_{i\ell}^2 k_{ij} = (\alpha \odot s_\ell)^\top K \mathbf{1}.
$$

Second term: Considering only the ℓ-th dimension, this is

$$
\sum_{i=1}^{n}\left[\sum_{j=1}^{n}\alpha_j (x_{j\ell} - x_{i\ell})\exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)\right]^2
$$

In matrix notation, the inner sum is a column vector,

$$
K(\alpha \odot x_\ell) - (K\alpha) \odot x_\ell.
$$

We take the entry-wise square and sum the resulting vector. Denote by $D_x := \mathrm{diag}(x)$, then the following two relations hold

$$
\begin{aligned}
K(\alpha \odot x) &= K D_x \alpha \\
(K\alpha) \odot x &= D_x K \alpha.
\end{aligned}
$$

This means that $J(\alpha)$ as defined previously,

$$
\begin{aligned}
J(\alpha) = \frac{2}{n\sigma} \sum_{\ell=1}^{d} &\left[\frac{2}{\sigma} \left[\alpha^T K s_\ell + (\alpha \odot s_\ell)^T K \mathbf{1} - 2(\alpha \odot x_\ell)^T K x_\ell \right] - \alpha^T K \mathbf{1} \right] \\
&+ \frac{2}{n\sigma^2} \sum_{\ell=1}^{d} \left[(\alpha \odot x_\ell)^T K - x_\ell^T \odot (\alpha^T K) \right] \left[K(\alpha \odot x_\ell) - (K\alpha) \odot x_\ell \right],
\end{aligned}
$$

can be rewritten as

$$
\begin{aligned}
J(\alpha) &= \frac{2}{n\sigma} \alpha^T \sum_{\ell=1}^{d} \left[\frac{2}{\sigma} (K s_\ell + D_{s_\ell} K \mathbf{1} - 2 D_{x_\ell} K x_\ell) - K \mathbf{1} \right] \\
&\quad + \frac{2}{n\sigma^2} \alpha^T \left(\sum_{\ell=1}^{d} \left[D_{x_\ell} K - K D_{x_\ell} \right] \left[K D_{x_\ell} - D_{x_\ell} K \right] \right) \alpha \\
&= \frac{2}{n\sigma} \alpha^T b + \frac{2}{n\sigma^2} \alpha^T C \alpha,
\end{aligned}
$$

where

$$
\begin{aligned}
b &= \sum_{\ell=1}^{d} \left(\frac{2}{\sigma} (K s_\ell + D_{s_\ell} K \mathbf{1} - 2 D_{x_\ell} K x_\ell) - K \mathbf{1} \right) \in \mathbb{R}^n \\
C &= \sum_{\ell=1}^{d} \left[D_{x_\ell} K - K D_{x_\ell} \right] \left[K D_{x_\ell} - D_{x_\ell} K \right] \in \mathbb{R}^{n \times n}.
\end{aligned}
$$

Assuming C is invertible, for $\lambda > 0$, this is minimised by

$$
\hat{\alpha} = \frac{-\sigma}{2} C^{-1} b.
$$

□

Similar to [13], we in practice add a term $\lambda \|\alpha\|^2$ for $\lambda \in \mathbb{R}^+$, in order to control the norm of the natural parameters in the RKHS $\|f\|^2_{\mathcal{H}}$. This results in the regularised and numerically more stable solution $\hat{\alpha}_\lambda := (C + \lambda I)^{-1} b$. We use the un-regularised solution for notational ease throughout the rest of the paper, but always regularise in practice.

Linear computational costs via low-rank approximations

Solving the linear system in (6) requires $\mathcal{O}(n^3)$ computation and $\mathcal{O}(n^2)$ storage for a fixed random sub-sample of the chain history $\mathbf{z}$. In order to allow for large n, and to exploit potential manifold structure in the RKHS, we apply a low-rank approximation to the kernel matrix via incomplete Cholesky [24, Alg. 5.12], that is a standard way to achieve linear computational costs for kernel methods. We rewrite the kernel matrix

$$
K \approx L L^T,
$$

where $L \in \mathbb{R}^{n \times \ell}$ is obtained via dual partial Gram–Schmidt orthonormalisation and costs both $\mathcal{O}(n\ell)$ computation and storage. Usually $\ell \ll n$, and ℓ can be chosen via an accuracy cut-off parameter on the kernel spectrum in the same fashion as for other low-rank approximations, such as PCA[6]. Given such a representation of K, we can rewrite any matrix-vector product as

$$
Kb \approx (LL^T)b = L(L^T b),
$$

[6] In this paper, we solely use the Gaussian kernel, whose spectrum decays exponentially fast.

where each left multiplication of L costs $\mathcal{O}(n\ell)$ and we never need to store LL^T. This idea can be used to achieve costs of $\mathcal{O}(n\ell)$ when computing b, and left-multiplying C. Combining the technique with conjugate gradient (CG) [25] allows to solve (6) with a maximum of n such matrix-vector products, yielding a total computational cost of $\mathcal{O}(n^2\ell)$. In practice, we can monitor residuals and stop CG after a fixed number of iterations $\tau \ll n$, where τ depends on the decay of the spectrum of K. We arrive at a *linear* total cost of $\mathcal{O}(n\ell\tau)$ computation and $\mathcal{O}(n\ell)$ storage. CG also has the advantage of allowing for 'hot starts', i.e. initialising the linear solver at a previous solution. Further details will be published with the implementation.

A.3 Finite feature space estimator

Proof of Proposition 2

We assume the model log-density (9) takes the primal form in a finite dimensional feature space as in Proposition 2, then again directly implement score functions (11) and minimise the empirical score matching objective (10) via a linear solve.

Proof. As assumed the log unnormalised density takes the form

$$f(x) = \langle \theta, \phi_x \rangle_{\mathcal{H}_m} = \theta^\top \phi_x,$$

where $x \in \mathbb{R}^d$ is embedded into a finite dimensional feature space $\mathcal{H}_m = \mathbb{R}^m$ as $x \mapsto \phi_x$. The score function (11) then can be written as the simple linear form

$$\psi_\ell(\xi;\theta) = \theta^T \dot{\phi}_x^\ell \quad \text{and} \quad \partial_\ell \psi_\ell(\xi;\theta) = \theta^T \ddot{\phi}_x^\ell, \tag{12}$$

where we defined the m-dimensional feature vector derivatives $\dot{\phi}_x^\ell := \frac{\partial}{\partial x_\ell}\phi_x$ and $\ddot{\phi}_x^\ell := \frac{\partial^2}{\partial x_\ell^2}\phi_x$. Plugging those into the empirical score matching objective in (10), we arrive at

$$\begin{aligned}
J(\theta) &= \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left[\partial_\ell \psi_\ell(x_i;\theta) + \frac{1}{2}\psi_\ell^2(x_i;\theta)\right] \\
&= \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left[\theta^T \ddot{\phi}_{x_i}^\ell + \frac{1}{2}\theta^T \left(\dot{\phi}_{x_i}^\ell \left(\dot{\phi}_{x_i}^\ell\right)^T\right)\theta\right] \\
&= \frac{1}{2}\theta^T C \theta - \theta^T b
\end{aligned} \tag{13}$$

where

$$b := -\frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \ddot{\phi}_{x_i}^\ell \in \mathbb{R}^m \quad \text{and} \quad C := \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left(\dot{\phi}_{x_i}^\ell \left(\dot{\phi}_{x_i}^\ell\right)^T\right) \in \mathbb{R}^{m\times m}. \tag{14}$$

Assuming C is invertible (trivial for $n \geq m$), the objective is uniquely minimised by differentiating (13) wrt. θ, setting to zero, and solving for θ. This gives

$$\hat{\theta} := C^{-1} b. \tag{15}$$

□

Again, similar to [13], we in practice add a term $\lambda\|\theta\|^2$ for $\lambda \in \mathbb{R}^+$ to (13), in order to control the norm of the natural parameters $\theta \in \mathcal{H}^m$. This results in the regularised and numerically more stable solution $\hat{\theta}_\lambda := (C + \lambda I)^{-1} b$. We use the un-regularised solution (15) for notational ease throughout the rest of the paper, but always regularise in practice.

Next, we be more concrete about the approximate feature space $\mathcal{H}^m$. Note that the above approach can be combined with *any* set of finite dimensional approximate feature mappings ϕ_x.

Example: Random Fourier features for the Gaussian kernel

We now combine the finite dimensional approximate infinite dimensional exponential family model with the "random kitchen sink" framework made popular by [16]. Assume a translation invariant kernel $k(x, y) = \tilde{k}(x - y)$. Bochner's theorem gives the representation

$$k(x, y) = \tilde{k}(x - y) = \int_{\mathbb{R}^d} \exp\left(i\omega^T (x - y)\right) d\Gamma(\omega),$$

where $\Gamma(\omega)$ is the Fourier transform of the kernel. An approximate feature mapping for such kernels can be obtained via dropping imaginary terms and approximating the integral with Monte Carlo integration. This gives

$$\phi_x = \sqrt{\frac{2}{m}} \left[\cos(\omega_1^T x + u_1), \ldots, \cos(\omega_m^T x + u_m)\right],$$

with fixed random basis vector realisations that depend on the kernel via $\Gamma(\omega)$,

$$\omega_i \sim \Gamma(\omega),$$

and fixed random offset realisations

$$u_i \sim \texttt{Uniform}[0, 2\pi],$$

for $i = 1 \ldots m$. It is easy to see that this approximation is consistent for $m \to \infty$, i.e.

$$\mathbb{E}_{\omega,b}\left[\phi_x^T \phi_y\right] = k(x, y).$$

See [16] for details and a uniform convergence bound. Note that one can achieve logarithmic computational costs in d exploiting properties of Hadamard matrices, see [26].

The score functions 12 are given by

$$\begin{aligned}
\dot{\phi}_\xi^\ell &= \sqrt{\frac{2}{m}} \frac{\partial}{\partial \xi_\ell} \left[\cos(\omega_1^T \xi + u_1), \ldots, \cos(\omega_m^T \xi + u_m)\right] \\
&= -\sqrt{\frac{2}{m}} \left[\sin(\omega_1^T \xi + u_1)\omega_{1\ell}, \ldots, \sin(\omega_m^T \xi + u_m)\omega_{m\ell}\right] \\
&= -\sqrt{\frac{2}{m}} \left[\sin(\omega_1^T \xi + u_1), \ldots, \sin(\omega_m^T \xi + u_m)\right] \odot \left[\omega_{1\ell}, \ldots, \omega_{m\ell}\right],
\end{aligned}$$

where $\omega_{1\ell}$ is the ℓ-th component of ω_1, and

$$\begin{aligned}
\ddot{\phi}_\xi^\ell :&= -\sqrt{\frac{2}{m}} \frac{\partial}{\partial \xi_\ell} \left[\sin(\omega_1^T \xi + u_1), \ldots, \sin(\omega_m^T \xi + u_m)\right] \odot \left[\omega_{1\ell}, \ldots, \omega_{m\ell}\right] \\
&= -\sqrt{\frac{2}{m}} \left[\cos(\omega_1^T \xi + u_1), \ldots, \cos(\omega_m^T \xi + u_m)\right] \odot \left[\omega_{1\ell}^2, \ldots, \omega_{m\ell}^2\right] \\
&= -\phi_\xi \odot \left[\omega_{1\ell}^2, \ldots, \omega_{m\ell}^2\right],
\end{aligned}$$

where $\odot$ is the element-wise product. Consequently the gradient itself is given by

$$\nabla_\xi \phi_\xi = \begin{bmatrix} \dot{\phi}_\xi^1 \\ \vdots \\ \dot{\phi}_\xi^d \end{bmatrix} \in \mathbb{R}^{d \times m}$$

An example pair of translation invariant kernel and its Fourier transform for the well-known *Gaussian kernel* are

$$k(x, y) = \exp\left(-\gamma \|x - y\|_2^2\right) \quad \text{and} \quad \Gamma(\omega) = \mathcal{N}\left(\omega_i \middle| \mathbf{0}, \gamma^2 I_m\right).$$

Constant cost updates

A convenient property of the finite feature space approximation is that its primal representation of the solution allows to update 14 in an online fashion. When combined with MCMC, each new point x_{t+1} of the Markov chain history only adds a term of the form $-\sum_{\ell=1}^{d}\ddot{\phi}_{x_{t+1}}^{\ell}\in\mathbb{R}^{m}$ and $\sum_{\ell=1}^{d}\dot{\phi}_{x_{t+1}}^{\ell}(\dot{\phi}_{x_{t+1}}^{\ell})^{T}\in\mathbb{R}^{m\times m}$ to the moving averages of b and C respectively. Consequently, when at iteration t, rather than fully re-computing 15 at the cost of $\mathcal{O}(tm^{3})$ for every new point, we can use rank-d updates to construct the minimiser of 13 from the solution of the previous iteration. Assume we have computed the sum of all moving average terms,

$$\tilde{C}_{t}^{-1}:=\left(\sum_{i=1}^{t}\sum_{\ell=1}^{d}\left(\dot{\phi}_{x_{i}}^{\ell}\left(\dot{\phi}_{x_{i}}^{\ell}\right)^{T}\right)\right)^{-1}$$

from feature vectors derivatives $\ddot{\phi}_{x_{i}}^{\ell}\in\mathbb{R}^{m}$ of some set of points $\{x_{i}\}_{i=1}^{t}$, and subsequently receive receive a new point x_{t+1}. We can then write the inverse of the new sum as

$$\tilde{C}_{t+1}^{-1}:=\left(\tilde{C}_{t}+\sum_{\ell=1}^{d}\left(\dot{\phi}_{x_{t+1}}^{\ell}\left(\dot{\phi}_{x_{t+1}}^{\ell}\right)^{T}\right)\right)^{-1}.$$

This is the inverse of the rank-d perturbed previous matrix $\tilde{C}_{t}$. We can therefore construct this inverse using d successive applications of the Sherman-Morrison-Woodbury formula for rank-one updates [27], each using $\mathcal{O}(m^{2})$ computation. Since $\tilde{C}_{t}$ is positive definite[7], we can represent its inverse as a numerically much more stable Cholesky factorisation $\tilde{C}_{t}=\tilde{L}_{t}\tilde{L}_{t}^{T}$. It is also possible to perform cheap rank-d updates of such Cholesky factors, see [27][28][8]. Denote by $\tilde{b}_{t}$ the sum of the moving average b. We solve (15) as

$$\hat{\theta}=C^{-1}b=\left(\frac{1}{t}\tilde{C}_{t}\right)^{-1}\left(\frac{1}{t}\tilde{b}_{t}\right)=\tilde{C}_{t}^{-1}\tilde{b}_{t}=\tilde{L}_{t}^{-T}\tilde{L}_{t}^{-1}\tilde{b}_{t},$$

using cheap triangular back-substitution from $\tilde{L}_{t}$, and never storing $\tilde{C}_{t}^{-1}$ or $\tilde{L}_{t}^{-1}$ explicitly.

Using such updates, the computational costs for updating the approximate infinite dimensional exponential family model in *every* iteration of the Markov chain are $\mathcal{O}(dm^{2})$, which *constant in t*. We can therefore use *all* points in the history for constructing a proposal – without the previously exploding computational costs of $\mathcal{O}(tdm^{3})$.

Algorithmic description:

1. Update sums

$$\tilde{b}_{t+1}\leftarrow\tilde{b}_{t}-\sum_{\ell=1}^{d}\ddot{\phi}_{x_{t+1}}^{\ell}\quad\text{and}\quad\tilde{C}_{t+1}\leftarrow\tilde{C}_{t}+\frac{1}{2}\sum_{\ell=1}^{d}\dot{\phi}_{x_{t+1}}^{\ell}(\dot{\phi}_{x_{t+1}}^{\ell})^{T}$$

2. Perform rank-d update to obtain updated Cholesky factorisation $\tilde{L}_{t+1}\tilde{L}_{t+1}^{T}=\tilde{C}_{t+1}$.
3. Update approximate infinite dimensional exponential family parameters

$$\hat{\theta}\leftarrow\tilde{L}_{t+1}^{-T}\tilde{L}_{t+1}^{-1}\tilde{b}_{t+1}$$

A.4 Ergodicity of KMC lite

Notation Denote by $\alpha(x_{t},x^{*}(p'))$ is the probability of accepting a (p',x^{*}) proposal at state x_{t}. Let $a\wedge b=\min(a,b)$. Define $c(x):=\epsilon^{2}\sum_{i=0}^{L-1}\nabla f(x_{i\epsilon})/2$ and $d(x):=\epsilon(\nabla f(x)+\nabla f(x_{L\epsilon}))/2+\epsilon\sum_{i=1}^{L-1}\nabla f(x_{i\epsilon})$.

[7] C is the empirical covariance of the feature derivatives $\dot{\phi}_{x_{i}}^{\ell}$.

[8] We use the open-source implementation provided at `https://github.com/jcrudy/choldate`

Proof of Proposition 3

Proof. We assumed $\pi(x)$ is log-concave in the tails, meaning $\exists x_U > 0$ s.t. for $x^* > x_t > x_U$, we have $\pi(x^*)/\pi(x_t) \leq e^{-\alpha_1(\|x^*\|_2 - \|x_t\|_2)}$ and for $x_t > x^* > x_U$, we have $\pi(x^*)/\pi(x_t) \geq e^{-\alpha_1(\|x^*\|_2 - \|x_t\|_2)}$, and a similar condition holds in the negative tail. Furthermore, we assumed fixed HMC parameters: L leapfrog steps L of size ϵ, and wlog the identity mass matrix I. Following [19, 29], it is sufficient to show

$$\limsup_{\|x_t\|_2 \to \infty} \int \left[e^{s(\|x^*(p')\|_2 - \|x_t\|_2)} - 1 \right] \alpha(x_t, x^*(p')) \mu(dp') < 0,$$

for some $s > 0$, where $\mu(\cdot)$ is a standard Gaussian measure. Denoting the integral $I_{-\infty}^{\infty}$, we split it into

$$I_{-\infty}^{-x_t^\delta} + I_{-x_t^\delta}^{x_t^\delta} + I_{x_t^\delta}^{\infty},$$

for some $\delta \in (0, 1)$. We show that the first and third terms decay to zero whilst the second remains strictly negative as $x_t \to \infty$ (a similar argument holds as $x_t \to -\infty$). Taking $I_{-x_t^\delta}^{x_t^\delta}$, we can choose an x_t large enough that $x_t - C - L\epsilon x_t^\delta > x_U$, $-\gamma_1 < c(x_t - x_t^\delta) < 0$ and $-\gamma_2 < d(x_t - x_t^\delta) < 0$. So for $p' \in (0, x_t^\delta)$ we have

$$L\epsilon p' > x^* - x_t > L\epsilon p' - \gamma_1 \implies e^{-\alpha_1(-\gamma_1 + L\epsilon p')} \geq e^{-\alpha_1(x^* - x_t)} \geq \pi(x^*)/\pi(x_t),$$

where the last inequality is from (i). For $p' \in (\gamma_2^2/2, x_t^\delta)$

$$\alpha(x_t, x^*) \leq 1 \wedge \frac{\pi(x^*)}{\pi(x_t)} \exp\left(p'\gamma_2/2 - \gamma_2^2/2\right) \leq 1 \wedge \exp\left(-\alpha_2 p' + \alpha_1\gamma_1 - \gamma_2^2/2\right),$$

where x_t is large enough that $\alpha_2 = \alpha_1 L\epsilon - \gamma_2/2 > 0$. Similarly for $p' \in (\gamma_1/L\epsilon, x_t^\delta)$

$$e^{sL\epsilon p'} - 1 \geq e^{s(x^* - x_t)} - 1 \geq e^{s(L\epsilon p' - \gamma_1)} - 1 > 0.$$

Because γ_1 and γ_2 can be chosen to be arbitrarily small, then for large enough x_t we will have

$$\begin{aligned} 0 < I_0^{x_t^\delta} &\leq \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{sL\epsilon p'} - 1] \exp\left(-\alpha_2 p' + \alpha_1\gamma_1 - \gamma_2^2/2\right) \mu(dp') + I_0^{\gamma_1/L\epsilon} \\ &= e^{c_1} \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{s_2 p'} - 1] e^{-\alpha_2 p'} \mu(dp') + I_0^{\gamma_1/L\epsilon}, \end{aligned} \tag{16}$$

where $c_1 = \alpha_1\gamma_1 - \gamma_2^2/2 > 0$ for large enough x_t, as γ_1 and γ_2 are of the same order. Now turning to $p' \in (-x_t^\delta, 0)$, we can use an exact rearrangement of the same argument (noting that c_1 can be made arbitrarily small) to get

$$I_{-x_t^\delta}^{0} \leq e^{c_1} \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{-s_2 p'} - 1] \mu(dp') < 0. \tag{17}$$

Combining (16) and (17) and rearranging as in [29, Theorem 3.2] shows that $I_{-x_t^\delta}^{x_t^\delta}$ is strictly negative in the limit if $s_2 = sL\epsilon$ is chosen small enough, as $I_0^{\gamma_2/L\epsilon}$ can also be made arbitrarily small.

For $I_{-\infty}^{-x_t^\delta}$ it suffices to note that the Gaussian tails of $\mu(\cdot)$ will dominate the exponential growth of $e^{s(\|x^*(p')\|_2 - \|x_t\|_2)}$ meaning the integral can be made arbitrarily small by choosing large enough x_t, and the same argument holds for $I_{x_t^\delta}^{\infty}$. □

B Various

Free parameters KMC, for both the lite and the finite estimator has two free parameters: the Gaussian kernel bandwidth σ, and the regularisation parameter λ. Earlier adaptive kernel-based MCMC methods, [12], did not cover choosing parameters. As KMC's performance is eventually tied with the quality of the approximate infinite dimensional exponential family model in (5) or (7), we can use the score matching objective function in (13) to compare σ, λ pairs via corss-validation in a principled way.