Gradient-free Hamiltonian Monte Carlo with Efficient Kernel Exponential Families

Heiko Strathmann
Gatsby Unit
University College London
heiko.strathmann@gmail.com

Dino Sejdinovic
Department of Statistics
University of Oxford
dino.sejdinovic@gmail.com

Samuel Livingstone
Department of Statistics
University College London
samuel.livingstone@ucl.ac.uk

Zoltán Szabó
Gatsby Unit
University College London
zoltan.szabo@gatsby.ucl.ac.uk

Arthur Gretton
Gatsby Unit
University College London
arthur.gretton@gmail.com

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. Computational costs are reduced by two novel efficient approximations to this gradient. While being asymptotically exact, KMC mimics HMC in terms of sampling efficiency and offers substantial mixing improvements to state-of-the-art gradient-free samplers. 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 these trajectories cover wide parts of the target's support, and the probability of accepting a move along a trajectory is often close to one. Remarkably, this property is mostly invariant to dimensionality. Thus, HMC is often superior to random walk methods, which need 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, but can only be estimated at any given point, e.g. in 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. Recently, there have been efforts to mimic HMC's behaviour using stochastic gradients from mini-batches in Big Data [7], or stochastic finite differences in ABC [8]. Stochastic gradient based HMC methods, however, often suffer from low acceptance rates or additional bias that is hard to quantify [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. Yet, for densities with non-linear support across components, this approach does not work very well. Recently, [12] introduced a Kernel Adaptive Metropolis-Hastings (KAMH) algorithm, with proposals locally aligned to the target density. By adaptively learning target covariance in a Reproducing Kernel Hilbert Space (RKHS), KAMH achieves improved sampling efficiency.

In this paper, we extend the idea of using kernel methods to learn efficient proposal distributions [12]. Rather than *locally* smoothing the target density, however, we estimate its gradients *globally*. More precisely, we fit an (unnormalised) infinite dimensional exponential family model in a RKHS via score matching [13, 14]. This is a non-parametric method to model the log unnormalised target density as an 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 dimensionality – in sharp contrast to classical kernel density estimation [15, Sec. 6.5]. A Gaussian Process (GP) was also used in [16] as an emulator of the target density in order to speed up HMC, however this work requires access to the log target density in closed form, to provide training points for the GP regressor.

We require our adaptive algorithm to be computationally efficient, as it deals with high-dimensional MCMC chains of growing length. Thus, we develop two novel approximations to the infinite dimensional exponential family model. The first approximation, *score matching lite*, is based on computing the solution in terms of 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 [17]. This results in an extremely efficient online 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; alternatively the two approaches could be combined.

Experiments show that KMC inherits the efficiency of HMC, and therefore mixes significantly better than state-of-the-art gradient-free adaptive samplers on a number of target densities, including on synthetic examples, and when used in PM-MCMC and ABC-MCMC.

Paper outline: In Section 2, we place our contribution in the context of previous work and cover HMC basics. Section 3 introduces Hamiltonian dynamics induced by kernel exponential families. Section 4 contains our approximate estimators of of the log density and its gradient, and Section 5 applies these results to obtain our Kamiltonian Monte Carlo algorithm. We demonstrate the efficiency of KMC in a number of experiments in Section 6.

2 Background and Previous Work

Let the domain of interest $\mathcal{X}$ be a compact[1] subset of $\mathbb{R}^d$, 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 $x^* \sim Q(\cdot|x_t)$, and sets $x_{t+1} \leftarrow x^*$ with probability $\min(1, [\pi(x^*)Q(x_t|x^*)]/[\pi(x_t)Q(x^*|x_t)])$, and $x_{t+1} \leftarrow x_t$ otherwise. In this paper, we generally assume that π is intractable[2], i.e. that we can neither evaluate $\pi(x)$ nor[3]

[1] The compactness restriction is imposed to satisfy the assumptions in [13].

[2] π is unavailable due to analytic intractability, as opposed to computationally expensive in the Big Data context.

[3] Throughout the paper ∇ denotes the gradient operator w.r.t. to x.

$\nabla \log \pi(x)$ for any 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 is a random walk, i.e. $Q(\cdot|x_t) = \mathcal{N}(\cdot|x_t, \Sigma_t)$. A popular choice of the scaling is $\Sigma_t \propto I$. When the (unknown) scale of the target density is not uniform across dimensions, or if there are strong correlations, the original AMH algorithm [10, 11] improves mixing by adaptively learning global covariance structure of π from the history of the Markov chain. For cases where the local scaling does not match the global covariance structure of π, for instance when the support of the target is highly nonlinear, KAMH [12] improves mixing by learning the target covariance structure in a RKHS. KAMH proposals are Gaussian with a covariance 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) often overcomes random walk behaviour by utilising deterministic, measure-preserving maps to generate efficient Markov transitions that [1, 18]. Starting from the negative log unnormalised target density, referred to as the *potential energy* $U(q) = -\log \pi(q)$, we introduce an auxiliary *momentum* variable $p \sim \exp(-K(p))$ with $p \in \mathcal{X}$. The joint distribution of (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 $\phi_t^H : (p, q) \mapsto (p^*, q^*)$ for which $H(p^*, q^*) = H(p, q)$ for any $t \in \mathbb{R}$. This allows construction of π-invariant Markov chains: for a chain at state $q = x_t$, we repeatedly (i) re-sample $p' \sim \exp(-K(\cdot))$, and then (ii) apply the Hamiltonian flow for time t, giving $(p^*, q^*) = \phi_t^H(p', q)$. The flow can be generated by the *Hamiltonian operator*

$$\hat{H} := \frac{\partial K}{\partial p}\frac{\partial}{\partial q} - \frac{\partial U}{\partial q}\frac{\partial}{\partial p} =: \hat{K} + \hat{U}. \tag{1}$$

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 (p', q), the end-point of the approximate trajectory is accepted with probability $\min[1, \exp(-H(p^*q^*) + H(q, p'))]$. HMC is often able to propose distant, uncorrelated moves with a high acceptance probability.

Intractable densities In many cases the gradient of $\log \pi(q) = -U(q)$ is unavailable, leaving random-walk based methods as the state-of-the-art [12, 11]. KMC aims to overcome random-walk behaviour, so as to obtain significantly more efficient sampling [1].

3 Kernel Induced Hamiltonian Dynamics

KMC replaces the potential energy operator $\hat{U}$ in (1) by a kernel induced surrogate $\hat{U}_k$ computed from the history of the Markov chain. As we will see, this surrogate does not require gradients of the log-target density. The surrogate induces a kernel Hamiltonian flow, which can be numerically simulated using standard leap-frog integration. As with the discretisation error in HMC, any deviation of the kernel induced flow to the true flow is corrected via a Metropolis acceptance procedure. Consequently, the stationary distribution of the Markov chain will remain correct.[4]

Infinite Dimensional Exponential Families in a RKHS We construct a kernel induced potential energy surrogate whose gradients approximate the gradients of the true potential energy U in (1), without accessing π or $\nabla\pi$ directly, but only using the history of the Markov chain. To that end, we model the (unnormalised) target density $\pi(x)$ with an infinite dimensional exponential family model [13] of the form

$$\text{const} \times \pi(x) \approx \exp\left(\langle f, k(x, \cdot)\rangle_{\mathcal{H}} - A(f)\right), \tag{2}$$

which in particular implies

$$\nabla f \approx -\nabla U = \nabla \log \pi.$$

Here $\mathcal{H}$ is a RKHS of real valued functions on $\mathcal{X}$. The RKHS has a uniquely associated symmetric, positive definite *kernel* $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which satisfies $f(x) = \langle f, k(x, \cdot)\rangle$ for any $f \in \mathcal{H}$ [19].

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

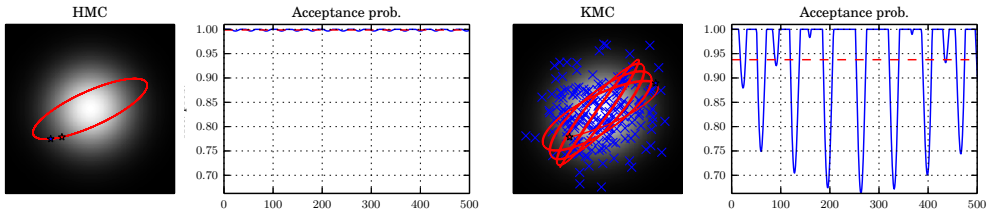

Figure 1: Hamiltonian trajectories on a 2-dimensional standard Gaussian. End points of such trajectories (red stars to blue stars) form the proposal of HMC-like algorithms. **Left:** Plain Hamiltonian trajectories oscillate on a stable orbit, and acceptance probability is close to one. **Right:** Kernel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

The canonical feature map $k(\cdot, x) \in \mathcal{H}$ here takes the role of the *sufficient statistics* while $f \in \mathcal{H}$ are the *natural parameters*, and $A(f) := \log \int_{\mathcal{X}} \exp(\langle f, k(x, \cdot)\rangle_{\mathcal{H}}) dx$ is the cumulant generating function. (2) defines broad class of densities: when universal kernels are used, the family is dense in the space of continuous densities on compact domains, with respect to the Total Variation, KL, and Hellinger divergences [13, Section 3]. It is possible to consistently fit an *unnormalised* version of (2) by directly minimising the expected gradient mismatch between the model (2) and the true target density π (observed through the Markov chain history). This is achieved by generalising the score matching approach [14] to infinite dimensional parameter spaces. The technique avoids the problem of dealing with the intractable $A(f)$, and reduces the problem to solving a linear system. More importantly, the approach is observed to be relatively robust to increasing dimensions, as opposed to classic kernel density estimation. We will return to the topic of estimation in Section 4, where we develop two efficient approximate empirical estimators. For now, assume access to an $\hat{f} \in \mathcal{H}$ such that $\nabla f(x) \approx \nabla \log \pi(x)$.

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 (1), and we have 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 a 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 (1). That said, 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 starting at (p', q) along the *kernel induced* flow with probability

$$\min\left[1, \exp\left(H\left(\phi_t^{H_k}(p', q)\right) - H(p', q)\right)\right], \tag{3}$$

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 (3). 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 Exponential Families in RKHS

We now address the topic of estimating the infinite dimensional exponential family model (2) from data. The original estimator in [13] has large computational costs. This is problematic in the adaptive MCMC context, where the model has to be updated on a regular basis. We propose two efficient approximations, each with its particular strengths and weaknesses. Both the original estimator for (2) and our approximations are based on score matching, see Appendix A.1 for a brief review.

4.1 Infinite Dimensional Exponential Families Lite

The original estimator of f in (2) takes a dual form in a RKHS sub-space spanned by $nd + 1$ kernel derivatives, [13, Thm. 4]. The update of the proposal at the iteration t of MCMC requires inversion of a $(td + 1) \times (td + 1)$ matrix. This is clearly prohibitive if we are to run even a moderately large number of iterations of a Markov chain. Following [12], we take a simple approach to avoid

prohibitive computational costs in t: we form 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 order to reduce excessive computational costs arising when d is large, we develop an approximation to the full dual solution in [13] by expressing the solution in terms of span $(\{k(z_i, \cdot)\}_{i=1}^n)$, which covers the support of the true density by construction, and grows with increasing n. That is, we assume that the log unnormalised density of the model in (2) takes the dual form

$$f(x) = \sum_{i=1}^{n} \alpha_i k(z_i, x), \tag{4}$$

where $\alpha \in \mathbb{R}^n$ are real valued parameters that are obtained by minimising the empirical score matching objective (see (9) in Appendix A.1). This representation is of a form similar to [20, Section 4.1], the main differences being that the basis functions are chosen randomly, the basis set grows with n, and we will require an additional regularising term. The estimator is summarised in the following proposition, which is proved 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* (9) *is given by*

$$\hat{\alpha}_\lambda = -\frac{\sigma}{2}(C + \lambda I)^{-1} b, \tag{5}$$

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

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

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

The estimator has a cost of $\mathcal{O}(n^3 + dn^2)$ in computation (both for computing C, b, and for 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, which are derived in 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. Evaluation and storage of $\nabla f(\cdot)$ both cost $\mathcal{O}(dn)$, which interestingly is independent of the target π, and only depends on the sub-sample $\mathbf{z}$.

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 online.

Define an m-dimensional approximate[5] feature space $\mathcal{H}_m = \mathbb{R}^m$, and denote by $\phi_x \in \mathcal{H}^m$ the embedding of a point $x \in \mathcal{X} = \mathbb{R}^d$ 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 (2) can be approximated in this feature space as

$$f(x) = \langle \theta, \phi_x \rangle_{\mathcal{H}_m} = \theta^\top \phi_x \tag{6}$$

In order to fit $\theta \in \mathbb{R}^m$, we again minimise the score matching objective in (9) in Appendix A.1.

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* (9) *is given by*

$$\hat{\theta}_\lambda := (C + \lambda I)^{-1} b, \tag{7}$$

where

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

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

[5] 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 $\{x_i\}_{i=1}^t$, perform (1-4) 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, $x_{t+1} \leftarrow x^*$ w.p. (3) and $x_{t+1} \leftarrow x_t$ otherwise

An example feature embedding based on random Fourier features [17] and a standard Gaussian kernel is $\phi_x = \sqrt{\frac{2}{m}} \left[\cos(\omega_1^T x + u_1), \ldots, \cos(\omega_m^T x + u_m)\right]$, with $\omega_i \sim \mathcal{N}(\omega)$ and $u_i \sim \texttt{Uniform}[0, 2\pi]$. The estimator has a one-off cost of $\mathcal{O}(tdm^2 + m^3)$ computation and $\mathcal{O}(m^2)$ storage. Given that we have computed a solution based on the Markov chain history $\{x_i\}_{i=1}^t$, however, it is straightforward to update C, b, and the solution $\hat{\theta}_\lambda$ online, after a new point x_{t+1} arrives. This is achieved by storing running averages and performing low-rank updates of matrix inversions, and costs $\mathcal{O}(dm^2)$ computation and $\mathcal{O}(m^2)$ storage, *independent* of t. Further details are given in Appendix A.3.

Gradients of the estimated log-density are written $\nabla f(x) = [\nabla \phi_x]^\top \hat{\theta}$, i.e., they require the evaluation of the gradient of the feature space embedding. Both the evaluation and storage of $\nabla f(\cdot)$ cost $\mathcal{O}(m)$, which is again independent of π *and* the Markov chain history.

5 Kamiltonian Monte Carlo

Constructing a kernel induced Hamiltonian flow as in Section 3 from the gradients of the infinite dimensional exponential family model (2), and approximate estimators (4),(6), we arrive at a gradient free, adaptive MCMC algorithm: *Kamiltonian Monte Carlo* (Algorithm 1). KMC overcomes random-walk behaviour of competing state-of-the-art samplers KAMH and AMH.

Computational Efficiency, Geometric Ergodicity, and Burn-in KMC finite using (6) allows for online updates using the *full* Markov chain history, and is this respect a more elegant solution than KMC lite, which has greater computational cost and requires sub-sampling the chain history. Due to the parametric nature of this approximate model, however, the tails of the estimator are not guaranteed to decay. For example, the random Fourier feature embedding described below Proposition 2 contains periodic cosine functions, and therefore oscillates in the tails of (6), resulting in a reduced acceptance probability. As we will demonstrate in the experiments, this problem does not appear when KMC finite is initialised in high-density regions, nor after burn-in. In situations where information about the target density support is unknown, and during burn-in, we suggest to use the lite estimator (5), whose gradients decay outside of the training data. More formally, in the tails or before burn-in completion, KMC lite is guaranteed to fall back to a random walk, and smoothly transitions to HMC-like proposals as the MCMC chain grows.

Proposition 3. *Assume $d = 1$, $\pi(x)$ is log-concave in the tails, and the regularity conditions of [21, 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 $\limsup_{\|x\|_2 \to \infty} \|\nabla f(x)\|_2 = 0$, and $\exists M : \forall x : \|\nabla f(x)\|_2 \leq M$, then KMC lite is geometrically ergodic from π-almost any starting point.*

Proof Sketch (see Appendix A.4 for the detailed proof) Define $c(x^{(0)}) := \epsilon^2 \sum_{i=0}^{L-1} \nabla f(x^{(i\epsilon)})/2$ and $d(x^{(0)}) := \epsilon(\nabla f(x^{(0)}) + \nabla f(x^{(L\epsilon)}))/2 + \epsilon \sum_{i=1}^{L-1} \nabla f(x^{(i\epsilon)})$, where $x^{(i\epsilon)}$ is the i-th point of the leapfrog integration from $x = x^{(0)}$. 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

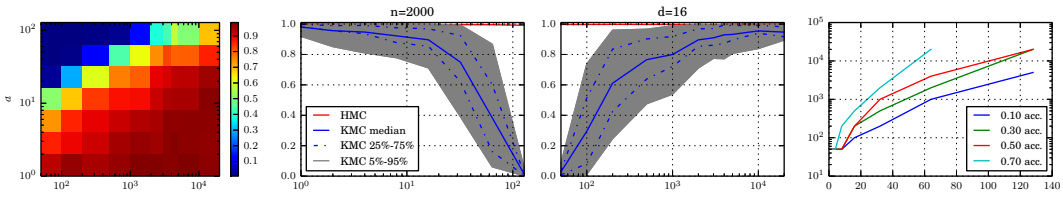

Figure 2: Acceptance probability of kernel induced Hamiltonian flow in high dimensions. **Left:** As a function of $n = m$ (x-axis) and d (y-axis). **Middle:** Slices through left plot with error bars for a fixed $n = m$ and as a function in d (left), and for a fixed d as a function of $n = m$ (right). **Right:** Number of data $n = m$ needed to reach given acceptance probabilities as a function of d.

$c(x_t) \xrightarrow{p} 0$ 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 $\text{acc}(x_t, x^*) \approx \min(1, \pi(x^*)/\pi(x_t))$, meaning in the tails the chain will behave as a Random Walk Metropolis. Consequently, KMC lite is geometrically ergodic whenever the Random Walk Metropolis is. Generalisations to $d \geq 2$ require an additional curvature condition of [21] 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 correct. We follow [12, Sec. 4.2] and the idea of "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 $\sum_{t=0}^{\infty} a_t = \infty$. We update the density gradient estimate according to this schedule in Algorithm 1. Intuitively, adaptation becomes less likely as the MCMC chain progresses, but never fully stops, while sharing asymptotic convergence with adaptation that stops at a fixed point [22, Theorem 1]. Note that Proposition 3 is a stronger statement about the convergence rate.

Free Parameters KMC has two free parameters: the Gaussian kernel bandwidth σ, and the regularisation parameter λ. KMC's performance depends on the quality of the approximate infinite dimensional exponential family model in (4) or (6), hence a principled approach is to use the score matching objective function in (9) in Appendix A.1 to choose σ, λ pairs via cross-validation (after burn-in). Earlier adaptive kernel-based MCMC methods [12] did not address parameter choice.

6 Experiments

We start by quantifying performance of KMC finite 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 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 the number of data and basis functions, which are set to be equal $n = m$, and of dimension d. In dimensions up to $d \approx 100$, we are able to obtain acceptance probabilities comparable to plain HMC with the finite estimator obtained 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 the performance of HMC as it sees more data. We compare KMC, HMC, an isotropic random walk (RW), and KAMH on the 8-dimensional nonlinear banana-shaped target from [12, 10]. To only quantify mixing, both KMC and KAMH used the same set of fixed burn-in samples. We quantify performance on estimating the target's mean, which is exactly $\mathbf{0}$. We tuned the scaling of KAMH and RW to achieve 23% acceptance. We set HMC parameters to achieve 80% acceptance and then used the same parameters for KMC. We ran all samplers for 2000+200 iterations from a random start point (chosen from burn-in samples), discarded the burn-in and computed 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 repeated the experiment for an increasing number of burn-in samples and basis functions $m = n$. Figure 3 shows the results as a function of $m = n$. KMC clearly outperforms RW and KAMH, and eventually achieves performance close to HMC as $n = m$ grows.

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 on the UCI Glass dataset [23]. Note that HMC is

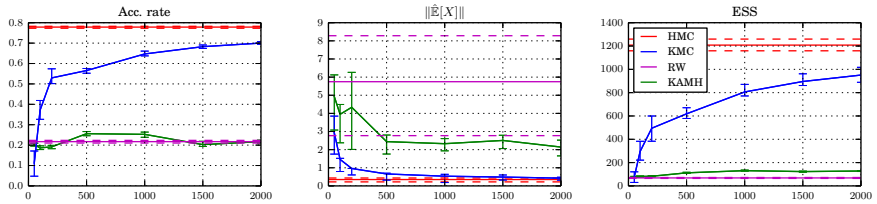

Figure 3: Results for 8-dimensional synthetic Banana. As the number of seen data increases, KMC performs close to HMC – outperforming KAMH and RW. 80% error bars over 30 runs.

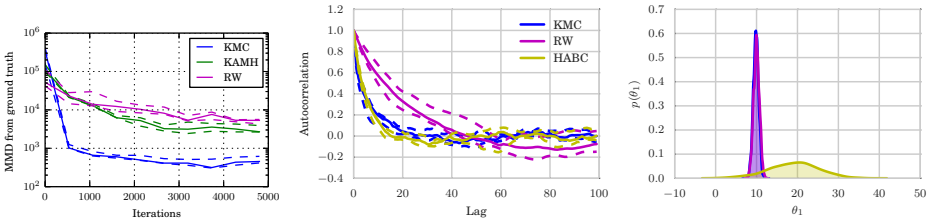

Figure 4: **Left:** Results for 9-dimensional marginal posterior over length scales of a GPC model applied to the UCI Glass dataset. The plots shows convergence of all mixed moments up to order 3 to a previously generated heavily thinned benchmark sample used as ground truth (lower MMD is better). **Middle/right:** ABC-MCMC auto-correlation and marginal θ_1 posterior for a 10-dimensional skew normal likelihood. While KMC mixes as well as HABC, it does not suffer from any bias (overlaps with RW, while HABC is significantly different) and requires fewer simulations per proposal.

unavailable for this problem, due to the intractability of the marginal data likelihood given the hyperparameters. Our experimental protocol mostly follows [12, Section 5.1], but uses only 1200+5000 MCMC iterations. We compare convergence in terms of all mixed moments of order up to 3 to a set of benchmark samples (MMD [24], lower is better). KMC randomly used 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 described settings were sufficient. Both KMC and KAMH used 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, with most of the time spent on estimating the marginal likelihood, in line with [12, Sec. 5.1].

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 obtained by simulation, 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 [25] with gradients based on stochastic finite differences. We remark that this requires to simulate from the likelihood in *every* leapfrog step, and that the additional bias from the Gaussian likelihood approximation can be problematic. In contrast, KMC does not require simulations to construct a proposal, but rather 'invests' simulations into an accept/reject step (3) that ensures convergence to the *original* ABC target. Figure 4 (right) compares performance of RW, HABC (sticky random numbers and SPAS, [8, Sec. 4.3, 4.4]), and KMC on a 10-dimensional skew-normal distribution $p(y|\theta) = 2\mathcal{N}(\theta, I)\,\Phi(\langle \alpha, y \rangle)$ with $\theta = \alpha = \mathbf{1} \cdot 10$. KMC mixes as well as HABC, but HABC suffers from a severe bias; we also reduce by a factor $L = 50$ the number of simulations per proposal.

7 Discussion

We have introduced KMC, a kernel-based gradient free adaptive MCMC algorithm, that mimics HMC's behaviour by estimating target gradients in an RKHS. In experiments, KMC outperforms

random walk based sampling methods in up to moderately high dimensions ($d \leq 100$), including the recent kernel-based KAMH [12]. KMC is particularly useful when gradients of the target density are unavailable, as in PM-MCMC or ABC-MCMC, where HMC is not an option. We have proposed two efficient empirical estimators, each with different strengths and weaknesses, and have 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 experimental 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] P. Marjoram, J. Molitor, V. Plagnol, and S. 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. Likelihood-free Markov chain Monte Carlo. *Handbook of Markov chain Monte Carlo*, 2010.

[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] C.E. Rasmussen. Gaussian Processes to Speed up Hybrid Monte Carlo for Expensive Bayesian Integrals. *Bayesian Statistics 7*, pages 651–659, 2003.

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

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

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

[20] A. Hyvärinen. Some extensions of score matching. *Computational Statistics & Data Analysis*, 51:2499–2512, 2007.

[21] 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.

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

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

[24] A. Gretton, K. Borgwardt, B. Schölkopf, A. J. Smola, and M. Rasch. A kernel two-sample test. *JMLR*, 13:723–773, 2012.

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

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

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

[28] 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.

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

[30] 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

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

$$\log \tilde{\pi}_Z(x; f) := \log \tilde{\pi}(x; f) - \log Z(f), \tag{8}$$

where f is a collection of parameters of yet unspecified dimension (c.f. natural parameters f of (2)), 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[6] $\{x_i\}_{i=1}^n \sim \pi$, such that $\pi(x) \approx \tilde{\pi}(x; \hat{f}) \times \text{const}$. From [14, Eq. 2], the criterion being optimised is the expected squared distance between score functions,

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

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

$$\hat{J}(f) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\ell=1}^{d} \left[\partial_\ell \psi_\ell(x_i; f) + \frac{1}{2} \psi_\ell^2(x_i; f) \right], \tag{9}$$

where

$$\psi_\ell(x; f) = \frac{\partial \log \tilde{\pi}(x; f)}{\partial x_\ell} \qquad \text{and} \qquad \partial_\ell \psi_\ell(x; f) = \frac{\partial^2 \log \tilde{\pi}(x; f)}{\partial x_\ell^2}. \tag{10}$$

Both proposed approximate estimators of the infinite dimensional exponential family model (2) from [13] are based on minimising (9) using approximate version of the scores (10).

A.2 Lite Estimator

Proof of Proposition 1

The proof below extends the model in [20, Section 4.1]. We assume the model log-density (8) takes the dual form in Proposition 1, then directly implement score functions (10) and derive a matrix expression of the empirical score matching objective (9), which can be minimised with a linear solve.

Proof. As assumed the log unnormalised density takes the form

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

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

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

The score functions from (10) are then given by

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

[6] 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.

and

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

Substituting this into (9) yields

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

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

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

Define

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

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 $D_x := \text{diag}(x)$, then the following two relations hold

$$K(\alpha \odot x) = KD_x\alpha$$
$$(K\alpha) \odot x = D_xK\alpha.$$

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

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

can be rewritten as

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

where

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

Assuming C is invertible, this is minimised by

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

□

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

Linear Computational Costs via Low-rank Approximations and Conjugate Gradient

Solving the linear system in (5) 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 [26, Alg. 5.12], that is a standard way to achieve linear computational costs for kernel methods. We rewrite the kernel matrix

$$K \approx LL^T,$$

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[7]. Given such a representation of K, we can rewrite any matrix-vector product as

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

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) allows to solve (5) 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 (8) takes the primal form in a finite dimensional feature space as in Proposition 2, then again directly implement score functions (10) and minimise the empirical score matching objective (9) 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 (10) then can be written as the simple linear form

$$\psi_\ell(\xi;\theta) = \theta^T \dot{\phi}_x^\ell \quad \text{and} \quad \partial_\ell \psi_\ell(\xi;\theta) = \theta^T \ddot{\phi}_x^\ell, \tag{11}$$

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 (9), we arrive at

$$\begin{aligned} J(\theta) &= \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left[\partial_\ell \psi_\ell(x_i;\theta) + \frac{1}{2}\psi_\ell^2(x_i;\theta)\right] \\ &= \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left[\theta^T \ddot{\phi}_{x_i}^\ell + \frac{1}{2}\theta^T \left(\dot{\phi}_{x_i}^\ell \left(\dot{\phi}_{x_i}^\ell\right)^T\right)\theta\right] \\ &= \frac{1}{2}\theta^T C \theta - \theta^T b \end{aligned} \tag{12}$$

where

$$b := -\frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \ddot{\phi}_{x_i}^\ell \in \mathbb{R}^m \quad \text{and} \quad C := \frac{1}{n}\sum_{i=1}^n \sum_{\ell=1}^d \left(\dot{\phi}_{x_i}^\ell \left(\dot{\phi}_{x_i}^\ell\right)^T\right) \in \mathbb{R}^{m\times m}. \tag{13}$$

Assuming C is invertible (trivial for $n \geq m$), the objective is uniquely minimised by differentiating (12) wrt. θ, setting to zero, and solving for θ. This gives

$$\hat{\theta} := C^{-1} b. \tag{14}$$

□

Again, similar to [13], we in practice add a term $\lambda\|\theta\|_2^2$ for $\lambda \in \mathbb{R}^+$ to (12), 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 $\hat{\theta}_\lambda := (C + \lambda I)^{-1} b$.

Next, we give an example for 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.

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

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" [17]. 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 [17] for details and a uniform convergence bound. Note that it is possible to achieve logarithmic computational costs in d exploiting properties of Hadamard matrices [27].

The score functions 11 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 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

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

Constant Cost Updates

A convenient property of the finite feature space approximation is that its primal representation of the solution allows to update (13) 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 $-\sum_{\ell=1}^{d} \ddot{\phi}^{\ell}_{x_{t+1}} \in \mathbb{R}^m$ and $\sum_{\ell=1}^{d} \dot{\phi}^{\ell}_{x_{t+1}} (\dot{\phi}^{\ell}_{x_{t+1}})^T \in \mathbb{R}^{m \times m}$ to the moving averages of b and C respectively. Consequently, at iteration t, rather than fully re-computing (14) at the cost of $\mathcal{O}(tdm^2 + m^3)$ for every new point, we can use rank-d updates to construct the minimiser of (12) from the solution of the previous iteration. Assume we have computed the sum of all moving average terms,

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

from feature vectors derivatives $\ddot{\phi}^{\ell}_{x_i} \in \mathbb{R}^m$ 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

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

This is the inverse of the rank-d perturbed previous matrix $\bar{C}_t$. We can therefore construct this inverse using d successive applications of the Sherman-Morrison-Woodbury formula for rank-one updates [28], each using $\mathcal{O}(m^2)$ computation. Since $\bar{C}_t$ is positive definite[8], we can represent its inverse as a numerically much more stable Cholesky factorisation $\bar{C}_t = \bar{L}_t \bar{L}_t^T$. It is also possible to perform cheap rank-d updates of such Cholesky factors[9], see [28, 29]. Denote by $\bar{b}_t$ the sum of the moving average b. We solve (14) as

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

using cheap triangular back-substitution from $\bar{L}_t$, and never storing $\bar{C}_t^{-1}$ or $\bar{L}_t^{-1}$ 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.

Algorithmic Description:

1. Update sums

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

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

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

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 $c(x^{(0)}) := \epsilon^2 \sum_{i=0}^{L-1} \nabla f(x^{(i\epsilon)})/2$ and $d(x^{(0)}) := \epsilon(\nabla f(x^{(0)}) + \nabla f(x^{(L\epsilon)}))/2 + \epsilon \sum_{i=1}^{L-1} \nabla f(x^{(i\epsilon)})$, where $x^{(i\epsilon)}$ is the i-th point of the leapfrog integration from $x = x^{(0)}$.

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

[9] 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 of size ϵ, and wlog the identity mass matrix I. Following [21, 30], 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 $x_t \to -\infty$). We detail the case $\nabla f(x) \uparrow 0$ as $x \to \infty$ here, the other is analogous. 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

$$\begin{aligned} 0 < I_0^{x_t^\delta} &\leq \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{sL\epsilon p'} - 1] \exp\left(-\alpha_2 p' + \alpha_1\gamma_1 - \gamma_2^2/2\right) \mu(dp') + I_0^{\gamma_1/L\epsilon} \\ &= e^{c_1} \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{s_2 p'} - 1] e^{-\alpha_2 p'} \mu(dp') + I_0^{\gamma_1/L\epsilon}, \end{aligned} \tag{15}$$

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

$$I_{-x_t^\delta}^0 \leq e^{c_1} \int_{\gamma_1/L\epsilon}^{x_t^\delta} [e^{-s_2 p'} - 1] \mu(dp') < 0. \tag{16}$$

Combining (15) and (16) and rearranging as in [30, 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 $I_{x_t^\delta}^{\infty}$. □