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 $\pi(x) \approx \tilde{\pi}(x; \hat{f}) \times \text{const}$. 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

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

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

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

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

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

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

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

can be rewritten as

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

where

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

Assuming C is invertible, 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

$$K \approx LL^T,$$

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[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 $-\sum_{\ell=1}^{d} \ddot{\phi}^{\ell}_{x_{t+1}} \in \mathbb{R}^m$ 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, 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,

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

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

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

This is the inverse of the rank-d perturbed previous matrix $\tilde{C}_t$. 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

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

2. Perform rank-d update to obtain updated Cholesky factorisation $\tilde{L}_{t+1} \tilde{L}_{t+1}^T = \tilde{C}_{t+1}$.
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 $\dot{\phi}^{\ell}_{x_i}$.

[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.