

---

## Nonparametric Hamiltonian Monte Carlo (Appendix)

---

### A. Statistical PCF

In this section, we present a simply-typed statistical probabilistic programming language with (stochastic) branching and recursion, and its operational semantics.

This language serves two purposes for the NP-HMC algorithm. First, it is a purified universal probabilistic programming language (PPL) widely considered (Borgström et al., 2016; Vákár et al., 2019; Mak et al., 2021) which specifies tree-representable functions that satisfies Ass. 1, 2 and 3 (Prop. 7 and Lem. 12) and hence NP-HMC can be applied. Second, its (operational) semantics is used to prove correctness of NP-HMC in App. C.

#### A.1. Syntax

SPCF is a simply-typed higher-order universal PPL with branching and recursion. More formally, it is a statistical probabilistic version of call-by-value PCF (Scott, 1993; Sieber, 1990) with reals as the ground type. The terms and part of the typing system of SPCF are presented in Fig. 10. Free variables and closed terms are defined in the usual way. In the interest of readability, we sometimes use pseudocode (e.g. Listing 1) in the style of Python to express SPCF terms.

There are two probabilistic constructs of SPCF: the sampling construct `normal` draws from  $\mathcal{N}$ , the standard Gaussian distribution with mean 0 and variance 1; the scoring construct `score( $M$ )` enables conditioning on observed data by multiplying the weight of the current execution with the real number denoted by  $M$ . Note this is not limiting as the standard uniform distribution with endpoints 0 and 1 can be described as `cdfNormal(normal)` where `cdfNormal` is the cumulative distribution function (cdf) of the standard normal distribution. And any real-valued distribution with inverse cdf  $f$  can be described as `f(cdfNormal(normal))`.

*Remark 6.* The main difference between our variant of SPCF and the others (Vákár et al., 2019; Mak et al., 2021) is that our sampling construct draws from the *standard normal distribution* instead of the standard uniform distribution. This does not restrict nor extend our language and is only considered since the target (parameter) space of the standard HMC algorithm matches that of the support of a standard  $n$ -dimensional normal distribution.

**Types** (typically denoted  $\sigma, \tau$ ) and **terms** (typically  $M, N, L$ ):

$$\begin{aligned} \sigma, \tau &::= \mathbb{R} \mid \sigma \Rightarrow \tau \\ M, N, L &::= y \mid \underline{r} \mid \lambda y. M \mid M N \mid \text{if}(L \leq 0, M, N) \mid \underline{f}(M_1, \dots, M_\ell) \mid YM \mid \text{normal} \mid \text{score}(M) \end{aligned}$$

**Typing system:**

$$\frac{}{\Gamma \vdash \text{normal} : \mathbb{R}} \quad \frac{\Gamma \vdash M : \mathbb{R}}{\Gamma \vdash \text{score}(M) : \mathbb{R}} \quad \frac{\Gamma \vdash M : (\sigma \Rightarrow \tau) \Rightarrow (\sigma \Rightarrow \tau)}{\Gamma \vdash YM : \sigma \Rightarrow \tau}$$

Figure 10. Syntax of SPCF, where  $r \in \mathbb{R}$ ,  $x, y$  are variables, and  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  ranges over a set  $\mathcal{F}$  of partial, measurable primitive functions.

#### A.2. Operational Semantics

The small-step reduction of SPCF is standard (see Borgström et al. (2016)). We present it as a rewrite system of *configurations*, which are triples of the form  $\langle M, w, \mathbf{t} \rangle$  where  $M$  is a closed SPCF term,  $w \in \mathbb{R}_{\geq 0}$  is a *weight*, and  $\mathbf{t} \in \mathbb{T}$  a trace, as defined in Fig. 11.

**Values** (typically denoted  $V$ ), **redexes** (typically  $R$ ) and **evaluation contexts** (typically  $E$ ):

$$\begin{aligned} V &::= \underline{r} \mid \lambda y. M \\ R &::= (\lambda y. M) V \mid \text{if}(\underline{r} \leq 0, M, N) \mid \underline{f}(r_1, \dots, r_\ell) \mid \mathsf{Y}(\lambda y. M) \mid \text{normal} \mid \text{score}(\underline{r}) \\ E &::= [] \mid E M \mid (\lambda y. M) E \mid \text{if}(E \leq 0, M, N) \mid \underline{f}(r_1, \dots, r_{i-1}, E, M_{i+1}, \dots, M_\ell) \mid \mathsf{Y} E \mid \text{score}(E) \end{aligned}$$

**Redex contractions:**

$$\begin{aligned} \langle (\lambda y. M) V, w, \mathbf{t} \rangle &\longrightarrow \langle M[V/y], w, \mathbf{t} \rangle \\ \langle \underline{f}(r_1, \dots, r_\ell), w, \mathbf{t} \rangle &\longrightarrow \begin{cases} \langle \underline{f}(r_1, \dots, r_\ell), w, \mathbf{t} \rangle & \text{if } (r_1, \dots, r_\ell) \in \text{Dom}(f), \\ \text{fail} & \text{otherwise.} \end{cases} \\ \langle \mathsf{Y}(\lambda y. M), w, \mathbf{t} \rangle &\longrightarrow \langle \lambda z. M[\mathsf{Y}(\lambda y. M)/y] z, w, \mathbf{t} \rangle && \text{(for fresh variable } z) \\ \langle \text{if}(\underline{r} \leq 0, M, N), w, \mathbf{t} \rangle &\longrightarrow \begin{cases} \langle M, w, \mathbf{t} \rangle & \text{if } r \leq 0, \\ \langle N, w, \mathbf{t} \rangle & \text{otherwise.} \end{cases} \\ \langle \text{normal}, w, \mathbf{t} \rangle &\longrightarrow \langle \underline{r}, w, \mathbf{t} \# [r] \rangle && \text{(for some } r \in \mathbb{R}) \\ \langle \text{score}(\underline{r}), w, \mathbf{t} \rangle &\longrightarrow \begin{cases} \langle \underline{r}, r \cdot w, \mathbf{t} \rangle & \text{if } r > 0, \\ \text{fail} & \text{otherwise.} \end{cases} \end{aligned}$$

**Evaluation contexts:**

$$\frac{\langle R, w, \mathbf{t} \rangle \longrightarrow \langle R', w', \mathbf{t}' \rangle}{\langle E[R], w, \mathbf{t} \rangle \longrightarrow \langle E[R'], w', \mathbf{t}' \rangle} \quad \frac{\langle R, w, \mathbf{t} \rangle \longrightarrow \text{fail}}{\langle E[R], w, \mathbf{t} \rangle \longrightarrow \text{fail}}$$

Figure 11. Operational small-step semantics of SPCF

In the rule for `normal`, a random value  $r \in \mathbb{R}$  is generated and recorded in the trace, while the weight remains unchanged: even though the program samples from a normal distribution, the weight does not factor in Gaussian densities as they are already accounted for by  $\mu_{\mathbb{T}}$ . In the rule for `score`( $\underline{r}$ ), the current weight is multiplied by  $r \in \mathbb{R}$ : typically this reflects the likelihood of the current execution given some observed data. Similarly to (Borgström et al., 2016) we reduce terms which cannot be reduced in a reasonable way (i.e. scoring with nonpositive constants or evaluating functions outside their domain) to fail.

We write  $\longrightarrow^+$  for the transitive closure of  $\longrightarrow$ , and  $\longrightarrow^*$  for the reflexive and transitive closure of  $\longrightarrow$ .

### A.2.1. VALUE AND WEIGHT FUNCTIONS.

Recall the measure space of traces  $\mathbb{T} := \bigcup_{n \in \mathbb{N}} \mathbb{R}^n$  is equipped with the standard disjoint union  $\sigma$ -algebra  $\Sigma_{\mathbb{T}} := \{\bigcup_{n \in \mathbb{N}} U_n \mid U_n \in \mathcal{B}_n\}$ , with measure given by summing the respective (higher-dimensional) normals  $\mu_{\mathbb{T}}(\bigcup_{n \in \mathbb{N}} U_n) := \sum_{n \in \mathbb{N}} \mathcal{N}_n(U_n)$ . Following Borgström et al. (2016), we write  $\Lambda$  to denote the set of all SPCF terms and view it as  $\bigcup_{n \in \mathbb{N}} (\text{SK}_n \times \mathbb{R}^n)$  where  $\text{SK}_n$  is the set of SPCF terms with exactly  $n$  numerals place-holders. The measurable space of terms is equipped with the  $\sigma$ -algebra  $\Sigma_{\Lambda}$  that is the Borel algebra of the countable disjoint union topology of the product topology of the discrete topology on  $\text{SK}_n$  and the standard topology on  $\mathbb{R}^n$ . Similarly the subspace  $\Lambda_v^0$  of closed values inherits the Borel algebra on  $\Lambda$ .

Let  $M$  be a closed SPCF term. Its **value function**  $\text{value}_M : \mathbb{T} \rightarrow \Lambda_v^0 \cup \{\perp\}$  returns, given a trace, the output value of the program, if the program terminates in a value. The **weight function**  $\text{weight}_M : \mathbb{T} \rightarrow \mathbb{R}_{\geq 0}$  returns the final weight of the corresponding execution. Formally:

$$\text{value}_M(\mathbf{t}) := \begin{cases} V & \text{if } \langle M, 1, [] \rangle \longrightarrow^* \langle V, w, \mathbf{t} \rangle \\ \perp & \text{otherwise} \end{cases} \quad \text{weight}_M(\mathbf{t}) := \begin{cases} w & \text{if } \langle M, 1, [] \rangle \longrightarrow^* \langle V, w, \mathbf{t} \rangle \\ 0 & \text{otherwise} \end{cases}$$

It follows already from (Borgström et al., 2016) that the functions  $\text{value}_M$  and  $\text{weight}_M$  are measurable.

Finally, every closed SPCF term  $M$  has an associated *value measure*

$$\begin{aligned} \llbracket M \rrbracket : \Sigma_{\Lambda_v^0} &\longrightarrow \mathbb{R}_{\geq 0} \\ U &\longmapsto \int_{\text{value}_M^{-1}(U)} \text{weight}_M \, d\mu_{\mathbb{T}} \end{aligned}$$

This corresponds to the denotational semantics of SPCF in the  $\omega$ -quasi-Borel space model via computational adequacy (Vákár et al., 2019).

**Proposition 7.** *Every closed SPCF term has a tree representable weight function.*

*Proof.* Assume  $M$  is a closed SPCF term and  $\mathbf{q} \in \text{Supp}^n(\text{weight}_M)$ . The reduction of  $M$  must be  $\langle M, 1, [] \rangle \longrightarrow^* \langle V, w, \mathbf{q} \rangle$  for some value  $V$  and weight  $w > 0$ . Assume for contradiction that there is some  $k < n$  where  $\langle M, 1, [] \rangle \longrightarrow^* \langle V', w', \mathbf{q}^{1\dots k} \rangle$  for some value  $V'$  and weight  $w' > 0$ . Since  $\mathbf{q}^{1\dots k}$  is a prefix of  $\mathbf{q}$  and  $\longrightarrow$  is deterministic if the trace is given, we must have  $\langle M, 1, [] \rangle \longrightarrow^+ \langle V', w', \mathbf{q}^{1\dots k} \rangle \longrightarrow^+ \langle V, w, \mathbf{q} \rangle$ , which contradicts the fact that  $V'$  is a value.  $\square$

### A.3. Almost-sure Termination

**Definition 8.** We say that a SPCF term  $M$  *terminates almost surely* if  $M$  is closed and  $\mu_{\mathbb{T}}(\{\mathbf{t} \in \mathbb{T} \mid \exists V, w. \langle M, 1, [] \rangle \longrightarrow^* \langle V, w, \mathbf{t} \rangle\}) = 1$ ;

The following proposition is used in Prop. 24 to support the correctness proof.

**Proposition 9.** *The value measure  $\llbracket M \rrbracket$  of a closed almost surely terminating SPCF term  $M$  which does not contain  $\text{score}(-)$  as a subterm is probabilistic.*

One of the main contribution of (Mak et al., 2021) is to find a suitable class of primitive functions such that their main theorem (Lem. 10) holds.

For our purposes, we take the set of *analytic functions* with co-domain  $\mathbb{R}$  as our class  $\mathcal{F}$  of primitive functions which, as shown in Example 3 of (Mak et al., 2021), satisfies the conditions for which the following lemma holds.

**Lemma 10** (Mak et al. (2021), Theorem 3). *Let  $M$  be an SPCF term which terminates almost surely. Then its weight function  $\text{weight}_M$  and value function  $\text{value}_M$  are differentiable almost everywhere.*

**Definition 11.** We say that a SPCF term  $M$  is *integrable* if  $M$  is closed and its value measure is finite, i.e.  $\llbracket M \rrbracket(\Lambda_v^0) < \infty$ ;

We conclude with the following lemma which shows that NP-HMC is an adequate inference algorithm for closed SPCF terms.

**Lemma 12.** *The weight function of a closed integrable almost surely terminating SPCF term satisfies Assumptions 1, 2 and 3 of the NP-HMC algorithm.*

*Proof.* Let  $M$  be a closed integrable almost surely terminating SPCF term, and  $w$  be its weight function.  $w$  is tree representable by Prop. 7. Integrability of  $w$  (Assumption 1) is given as an assumption, and  $w$  is almost everywhere continuously differentiable (Assumption 2) by Lem. 10.

Assume for contradiction that Assumption 3 does not hold. i.e. There is a non-null set  $U$  of infinite real-valued sequence where  $w$  is zero on all prefixes of sequences in  $U$ . Let  $U_p := \{\mathbf{q}^{1\dots k} \mid \mathbf{q} \in U, k \in \mathbb{N}\}$  be the set of prefixes of sequences in  $U$ . Since  $U$  is non-null,  $U_p$  must also be non-null. Moreover,  $w$  is zero on all traces in  $U_p$ . By the definition of weight function,  $\mathbf{q} \in U_p$  implies  $\langle M, 1, [] \rangle \not\longrightarrow^* \langle V, w', \mathbf{q} \rangle$  for some  $V$  and  $w'$ . Hence, the probability of a non-terminating run of  $M$  is non-zero and  $M$  is not almost surely terminating.  $\square$

*Remark 13.* The weight function as defined in App. A.2.1 is the input density function of the target distribution to which an inference algorithm typically samples from. In this paper, we call this function the “weight function” when considering semantics following (Culpepper & Cobb, 2017; Vákár et al., 2019; Mak et al., 2021), and use the notion “density” when referring it in an inference algorithm similar to (Zhou et al., 2019; 2020; Cusumano-Towner et al., 2020).

## B. Hamiltonian Monte Carlo Algorithm and its Variants

Hamiltonian Monte Carlo (HMC) algorithm (Duane et al., 1987; Cances et al., 2007; Neal, 2011) is a Markov chain Monte Carlo inference algorithm that generates samples from a continuous (finite) distribution  $\nu$  on the measure space  $(\mathbb{R}^n, \mathcal{B}_n, \text{Leb}_n)$ , where  $\mathcal{B}_n$  denotes the Borel  $\sigma$ -algebra.

### B.1. HMC Algorithm

To generate a Markov chain  $\{\mathbf{q}_i\}_{i \in \mathbb{N}}$  of samples from  $\nu$ , HMC simulates the *Hamiltonian* motion of a particle on the negative logarithm of the density function of  $\nu$  with some auxiliary momentum. Hence regions with high probability in  $\nu$  have low potential energy and are more likely to be visited by the simulated particle. In each iteration, the particle is given some random momentum. We formalise the algorithm here.

#### B.1.1. HAMILTONIAN DYNAMICS

Say  $\rho : \mathbb{R}^n \rightarrow \mathbb{R}$  is the (not necessarily normalized) probability density function of  $\nu$ . The simulated particle has two types of energies: **potential energy**  $U : \mathbb{R}^n \rightarrow \mathbb{R}$  given by  $U(\mathbf{q}) := -\log \rho(\mathbf{q})$  and **kinetic energy**  $K : \mathbb{R}^n \rightarrow \mathbb{R}$  given by  $K(\mathbf{p}) := -\log \text{pdf}_D(\mathbf{p})$  where  $D$  is some momentum distribution, typically a  $n$ -dimensional normal distribution. Henceforth, we take  $K(\mathbf{p}) := \sum_{i=1}^n \frac{p_i^2}{2}$ .

The **Hamiltonian**  $H : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  of a system is defined quite simply to be the sum of the potential and kinetic energies, i.e.

$$H(\mathbf{q}, \mathbf{p}) := U(\mathbf{q}) + K(\mathbf{p}).$$

The trajectories  $\{(\mathbf{q}^t, \mathbf{p}^t)\}_{t \geq 0}$ , where  $\mathbf{q}^t$  and  $\mathbf{p}^t$  are the position and momentum of the particle at time  $t$  respectively, defined by the Hamiltonian  $H$ , can be determined by the **Hamiltonian equations**:

$$\frac{d\mathbf{q}(t)}{dt} := \frac{\partial H}{\partial \mathbf{p}}(\mathbf{q}(t), \mathbf{p}(t)) = \nabla K(\mathbf{p}(t)) = \mathbf{p}(t) \quad \text{and} \quad \frac{d\mathbf{p}(t)}{dt} := -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{q}(t), \mathbf{p}(t)) = -\nabla U(\mathbf{q}(t)).$$

with initial conditions  $(\mathbf{q}(0), \mathbf{p}(0)) = (\mathbf{q}^0, \mathbf{p}^0)$ .

The **canonical distribution** (also called Boltzmann-Gibbs distribution)  $\pi$  on the measure space  $(\mathbb{R}^n \times \mathbb{R}^n, \Sigma_{\mathbb{R}^n \times \mathbb{R}^n}, \text{Leb}_{2n})$  corresponding to  $H$  is given by the probability density function

$$\zeta(\mathbf{q}, \mathbf{p}) := \frac{1}{Z} \exp(-H(\mathbf{q}, \mathbf{p})) = \frac{1}{Z} \exp(-U(\mathbf{q}) - K(\mathbf{p})) \quad \text{where } Z := \int_{\mathbb{R}^n} \rho \, d\text{Leb}_n$$

#### B.1.2. THE ALGORITHM

Since computers cannot simulate continuous motions like Hamiltonian, the equations of motion are generally numerically integrated by the **leapfrog** method (also called the velocity-Verlet algorithm (Verlet, 1967)):

$$\begin{aligned} \mathbf{p}^{n+1/2} &= \mathbf{p}^n - \epsilon/2 \cdot \nabla U(\mathbf{q}^n) \\ \mathbf{q}^{n+1} &= \mathbf{q}^n + \epsilon \cdot \mathbf{p}^{n+1/2} \\ \mathbf{p}^{n+1} &= \mathbf{p}^{(n+1)/2} - \epsilon/2 \cdot \nabla U(\mathbf{q}^{n+1}) \end{aligned}$$

where  $\epsilon$  is the time step.

The **integrator**  $\Psi_n : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n \times \mathbb{R}^n$  as given in Alg. 4, takes a state  $(\mathbf{q}, \mathbf{p})$  and performs  $L$  leapfrog steps with initial condition  $(\mathbf{q}^0, \mathbf{p}^0) := (\mathbf{q}, \mathbf{p})$  and time step  $\epsilon$ , and return the state  $(\mathbf{q}^L, -\mathbf{p}^L)$ .

**Proposition 14** (Bou-Rabee & Sanz-Serna (2018), Theorem 4.1 and 4.2). *The integrator  $\Psi_n$  is volume preserving (i.e.  $\Psi_{n*} \text{Leb}_{2n} = \text{Leb}_{2n}$ ) and reversible (i.e.  $\Psi_n = \Psi_n^{-1}$ ) on  $\mathbb{R}^n \times \mathbb{R}^n$ .*

*Proof.* Let  $\phi_k^P, \phi_k^Q : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n}$  be the transition of momentum and position variables with step size  $k$  respectively, i.e.  $\phi_k^P(\mathbf{q}, \mathbf{p}) = (\mathbf{q}, \mathbf{p} - k \nabla U(\mathbf{q}))$ , and  $\phi_k^Q(\mathbf{q}, \mathbf{p}) = (\mathbf{q} + k \nabla K(\mathbf{p}), \mathbf{p})$ . Hence, we can write the integrator  $\Psi_n$  as the composition  $S \circ \phi_{\epsilon/2}^P \circ \phi_{\epsilon}^Q \circ \phi_{\epsilon/2}^P$ , where  $S(\mathbf{q}, \mathbf{p}) := (\mathbf{q}, -\mathbf{p})$ .

**Algorithm 4** HMC Integrator  $\Psi_n$ 

**Input:** current state  $(\mathbf{q}_0, \mathbf{p}_0)$ , potential energy  $U$ , step size  $\epsilon$ , number of steps  $L$

**Output:** new state  $(\mathbf{q}, \mathbf{p})$

---

```

 $(\mathbf{q}, \mathbf{p}) = (\mathbf{q}_0, \mathbf{p}_0)$                                 {initialise}
for  $i = 0$  to  $L$  do
     $\mathbf{p} = \mathbf{p} - \frac{\epsilon}{2} \nabla U(\mathbf{q})$                         {1/2 momentum step}
     $\mathbf{q} = \mathbf{q} + \epsilon \mathbf{p}$                                 {1 position step}
     $\mathbf{p} = \mathbf{p} - \frac{\epsilon}{2} \nabla U(\mathbf{q})$                         {1/2 momentum step}
end for
 $\mathbf{p} = -\mathbf{p}$ 
return  $(\mathbf{q}, \mathbf{p})$ 
    
```

---

**Algorithm 5** HMC Step

**Input:** current sample  $\mathbf{q}_0$ , potential energy  $U$ , step size  $\epsilon$ , number of steps  $L$

**Output:** next sample  $\mathbf{q}$

---

```

 $\mathbf{p}_0 \sim \mathcal{N}_n$                                         {Kick}
 $(\mathbf{q}, \mathbf{p}) = \Psi_n((\mathbf{q}_0, \mathbf{p}_0), U, \epsilon, L)$         {Integrate}
if  $\mathcal{U}(0, 1)^a < \min\{1, \frac{\zeta(\mathbf{q}, \mathbf{p})}{\zeta(\mathbf{q}_0, \mathbf{p}_0)}\}$  then
    return  $\mathbf{q}$                                         {MH acceptance ratio}
else
    return  $\mathbf{q}_0$ 
end if
    
```

---

$^a \mathcal{U}(0, 1)$  is the standard uniform distribution.

It is easy to see that  $(\phi_k^P)^{-1} = S \circ \phi_k^P \circ S$  and  $(\phi_k^Q)^{-1} = S \circ \phi_k^Q \circ S$ . Hence,  $\Psi_n^{-1} = (\phi_{\epsilon/2}^P)^{-1} \circ (\phi_\epsilon^Q)^{-1} \circ (\phi_{\epsilon/2}^P)^{-1} \circ S = S \circ \phi_{\epsilon/2}^P \circ \phi_\epsilon^Q \circ \phi_{\epsilon/2}^P = \Psi_n$  and  $\Psi_n$  is reversible.

Similarly it is easy to see that the shear transformations  $\phi_k^P$ ,  $\phi_k^Q$  and momentum flip  $S$  preserves measure on  $\mathbb{R}^{2n}$ , i.e.  $\phi_k^P(D)$ ,  $\phi_k^Q(D)$ ,  $S(D)$  and  $D$  have the same measure for all measurable set  $D$  in  $\mathbb{R}^{2n}$ . Hence,  $(\Psi_n \star \text{Leb}_{2n})(D) = \text{Leb}_{2n}(\Psi_n^{-1}(D)) = \text{Leb}_{2n}(\Psi_n(D)) = \text{Leb}_{2n}(D)$  and  $\Psi_n$  is volume preserving.  $\square$

Alg. 5 shows how HMC generates a sample from the current one  $\mathbf{q}_0$ . It first performs leapfrog steps on  $(\mathbf{q}_0, \mathbf{p}_0)$  via the integrator  $\Psi_n$  with a randomly chosen initial momentum  $\mathbf{p}_0$ . The result  $(\mathbf{q}, \mathbf{p})$  of  $\Psi_n$  is then accepted with probability  $\min\{1, \frac{\zeta(\mathbf{q}, \mathbf{p})}{\zeta(\mathbf{q}_0, \mathbf{p}_0)}\}$ . Note that if Hamiltonian is preserved (i.e.  $H(\mathbf{q}, \mathbf{p}) = H(\mathbf{q}_0, \mathbf{p}_0)$ ), the acceptance probability is one and the proposal will always be accepted.

A Markov chain  $\{\mathbf{q}_i\}_{i \in \mathbb{N}}$  is generated by iterating Alg. 5.

### B.1.3. CORRECTNESS

The HMC algorithm is only effective if its generated Markov chain  $\{\mathbf{q}_i\}_{i \in \mathbb{N}}$  does converge to the target distribution  $\nu$ . Here we consider the typical convergence result of the total variation norm for the probability measure generated.

Formally, we say a Markov chain  $\{\mathbf{q}_i\}_{i \in \mathbb{N}}$  converges to the target distribution  $\nu$  on  $\mathbb{R}^n$  if

$$\forall \mathbf{q} \in \mathbb{R}^n, \quad \lim_{m \rightarrow \infty} \|Q^m(\mathbf{q}, -) - \nu\| = 0,$$

where  $Q^m(\mathbf{q}, A)$  is the probability for which the Markov chain is in  $A \in \mathcal{B}_n$  after  $m$  steps starting at  $\mathbf{q} \in \mathbb{R}^n$  and  $\|\cdot\|$  denotes the total variation norm on  $\mathbb{R}^n$  (i.e.  $\|\mu\| := \sup_{A \in \mathcal{B}_n} \mu(A) - \inf_{A \in \mathcal{B}_n} \mu(A)$ ).

Here we present the necessary conditions to prove such a result for the HMC algorithm. Let  $Q : \mathbb{R}^n \times \mathcal{B}_n \rightarrow \mathbb{R}_{\geq 0}$  be the transition kernel specified by Alg. 5, so that  $Q(\mathbf{q}, A)$  is the probability for which the next sample returned by Alg. 5 is in  $A \in \mathcal{B}_n$  given the current sample is  $\mathbf{q} \in \mathbb{R}^n$ . We write  $Q^m$  to be  $m$  compositions of  $Q$ . (i.e.  $Q^0(\mathbf{q}, A) := [\mathbf{q} \in A]$ ; for  $k > 0$ ,  $Q^{k+1}(\mathbf{q}, A) := \int_{\mathbb{R}^n} Q^k(\mathbf{q}', A) Q(\mathbf{q}, d\mathbf{q}')$ ).

First, we make sure that  $\nu$  is the invariant distribution of the Markov chain.

**Proposition 15** (Bou-Rabee & Sanz-Serna (2018), Theorem 5.2).  $\nu$  is invariant against  $Q$ .

While showing  $\nu$  is the invariant distribution for the Markov chain is relatively simple, we would be wrong to think that convergence follows trivially. In fact, as shown in the following example, the Markov chain can easily be periodic.

**Example 16** (Bou-Rabee & Sanz-Serna (2018), Example 5.1). Consider the case where the target distribution is a (unnormalised) one-dim. normal distribution. In particular say the potential energy is  $U(q) := q^2/2$ . Then, the Hamiltonian flow ( $H(q, p) = U(q) + K(p) = q^2/2 + p^2/2$ ) is a rotation in the  $(q, p)$ -plane with period  $2\pi$ . If the duration of the simulation is  $\pi$ , the exact flow returns  $q_1 = -q_0$ .

There are known conditions for which HMC converges to the right distribution (Schütte, 1999). Here we follow the treatment given by Cances et al. (2007).

Results from (Tierney, 1994; Borgström et al., 2016) tell us that it is enough to show that the transition kernel  $Q$  is **strongly  $\nu$ -irreducible**: for all  $a$  and  $B$ ,  $\nu(B) > 0$  implies  $Q(a, B) > 0$ .

**Lemma 17** (Cances et al. (2007), Lemma 2 and 3 (Strong irreducibility)). *Assume  $U$  is continuously differentiable, bounded above on  $\mathbb{R}^n$  and  $\nabla U$  is globally Lipschitz. Then the transition kernel  $Q$  is strongly  $\nu$ -irreducible.*

**Lemma 18** (Borgström et al. (2016), Lemma 33 (Aperiodicity)). *A strongly  $\nu$ -irreducible transition kernel is also  $\nu$ -aperiodic.*

**Lemma 19** (Tierney (1994), Theorem 1 and Corollary 2). *If the transition kernel  $Q$  with invariant distribution  $\nu$  is  $\nu$ -irreducible and  $\nu$ -aperiodic, then for all  $\mathbf{q}$ ,  $\lim_{n \rightarrow \infty} \|Q^n(\mathbf{q}, -) - \nu\| = 0$ .*

**Theorem 20.** *If  $U$  is continuously differentiable, bounded above on  $\mathbb{R}^n$  and  $\nabla U$  is globally Lipschitz, the Markov chain generated by iterating Alg. 5 converges to the target distribution  $\nu$ .*

## B.2. HMC Variants

### B.2.1. REFLECTIVE/REFRACTIVE HMC

Reflective/refractive HMC (RHMC) (Afshar & Domke, 2015) is an extension of HMC that improves its behaviour for discontinuous density functions. Standard HMC is correct for such distributions as well, but the acceptance probability may be very low and convergence extremely slow.

We need to quickly discuss what discontinuities mean in our setting: In addition to discontinuities of each  $U_n : \mathbb{R}^n \rightarrow \mathbb{R}$  itself, we also regard it as a discontinuity when  $\mathbf{q}$  leaves the support of  $U_n$ , since this means that a different branch in the tree representing function is chosen. The set of these discontinuities is  $\partial \text{Supp}(w)$ , i.e. the boundary of the support of the density function.

Fortunately, the extension of RHMC to our nonparametric setting is straightforward. The algorithm is described in Alg. 6. The only relevant difference is the need for an extend call in the algorithm.

The rest of the algorithm is the same as (Afshar & Domke, 2015): It uses two additional functions that deal with the discontinuities of  $U$ : `decompose` and `nextBoundary`. Just like in (Afshar & Domke, 2015), we assume that these are given to the algorithm because their implementation depends on the kind of discontinuities in the density function. In the original paper, they only consider discontinuities that are given by affine subspaces.

The function `nextBoundary`( $\mathbf{q}, \mathbf{p}, T, U$ ) takes a position  $\mathbf{q} \in \mathbb{R}^n$ , a momentum  $\mathbf{p} \in \mathbb{R}^n$ , a time limit  $T > 0$ , and family of potential energies  $\{U_n\}_{n \in \mathbb{N}}$ . It then checks whether a particle starting at  $\mathbf{q}$  moving with momentum  $\mathbf{p}$  will hit a discontinuity of  $U$  in time  $\leq T$ . If so, it returns the time  $t$  of “impact”, the position  $\mathbf{q}_<$  just before the discontinuity and  $\mathbf{q}_>$  just after the discontinuity.

The function `decompose`( $\mathbf{q}, \mathbf{p}, U$ ) takes a position  $\mathbf{q}$  on the discontinuity, a momentum  $\mathbf{p}$ , and  $U$  as before. It then decomposes the momentum  $\mathbf{p}$  into a component  $\mathbf{p}_\parallel$  that is parallel to the discontinuity and  $\mathbf{p}_\perp$  that is perpendicular to it.

The basic idea of the algorithm is inspired by reflection and refraction in physics. We simulate the trajectory of a particle according to Hamiltonian dynamics. When hitting a discontinuity, we compute the potential difference. If the kinetic energy is big enough to overcome it, refraction occurs: the perpendicular component of  $\mathbf{p}$  is scaled down. Otherwise, the particle is reflected.

The only difference to the original algorithm in (Afshar & Domke, 2015) is the call to `extend`. Why is it necessary? When hitting a discontinuity (and only then!), we may have to switch to a different branch on the tree representing the density function. Hence we may have to extend the position  $\mathbf{q}_>$  just after the discontinuity, which is why we call `extend` on it.

### B.2.2. LAPLACE MOMENTUM AND DISCONTINUOUS HMC

The Hamiltonian Monte Carlo method usually uses Gaussian momentum because it corresponds to the physical interpretation of kinetic energy being  $\frac{1}{2} \sum_i \mathbf{p}_i^2$  for a momentum vector  $\mathbf{p}$ . Nishimura et al. (2020) propose to use Laplace momentum where the kinetic energy for a momentum vector  $\mathbf{p}$  is given by  $\sum_i |\mathbf{p}_i|$ . This means that the momentum vector must follow a Laplace distribution, denoted as  $\mathcal{L}(0, 1)$ , with density proportional to  $\prod_i \exp(-|\mathbf{p}_i|)$ . Hamilton’s equations have to be changed to

$$\frac{d\mathbf{q}}{dt} = \text{sign}(\mathbf{p}), \quad \frac{d\mathbf{p}}{dt} = -\nabla_{\mathbf{q}} U.$$

**Algorithm 6** NP-RHMC Integrator  $\Psi_{\text{NP-R}}$ 

**Input:** current state  $(\mathbf{q}_0, \mathbf{p}_0)$ , family of potential energies  $\{U_n\}_{n \in \mathbb{N}}$ , step size  $\epsilon$ , number of steps  $L$

**Output:** new state  $(\mathbf{q}, \mathbf{p})$  computed according to Hamiltonian dynamics, extended initial state  $(\mathbf{q}_0, \mathbf{p}_0)$

---

```

 $(\mathbf{q}, \mathbf{p}) = (\mathbf{q}_0, \mathbf{p}_0)$  {initialise}
for  $i = 0$  to  $L$  do
     $\mathbf{p} = \mathbf{p} - \frac{\epsilon}{2} \nabla U_{|q_0|}(\mathbf{q})$  {1/2 momentum step}
     $t = 0$  {start of position step}
    while nextBoundary $(\mathbf{q}, \mathbf{p}, \epsilon - t, U)$  exists do
         $(t', \mathbf{q}_<, \mathbf{q}_>) = \text{nextBoundary}(\mathbf{q}, \mathbf{p}, \epsilon - t, U)$ 
         $t = t + t'$ 
         $((\mathbf{q}', \mathbf{p}'), (\mathbf{q}'_0, \mathbf{p}'_0)) = \text{extend}((\mathbf{q}_>, \mathbf{p}), (\mathbf{q}_0, \mathbf{p}_0), i\epsilon + t, U)$ 
         $\Delta U = (U_{|q'|}(\mathbf{q}') - U_{|q_<|}(\mathbf{q}_<))$ 
        if  $\|\mathbf{p}_\perp\|^2 > 2\Delta U$  then
             $(\mathbf{p}_\parallel, \mathbf{p}_\perp) = \text{decompose}(\mathbf{q}', \mathbf{p}', U)$ 
             $\mathbf{p}_\perp = \sqrt{\|\mathbf{p}_\perp\|^2 - 2\Delta U} \frac{\mathbf{p}_\perp}{\|\mathbf{p}_\perp\|}$  {refraction}
             $\mathbf{q} = \mathbf{q}'$ 
        else
             $(\mathbf{p}_\parallel, \mathbf{p}_\perp) = \text{decompose}(\mathbf{q}_<, \mathbf{p}, U)$ 
             $\mathbf{p}_\perp = -\mathbf{p}_\perp$  {reflection}
             $\mathbf{q} = \mathbf{q}_<$ 
        end if
         $\mathbf{p} = \mathbf{p}_\perp + \mathbf{p}_\parallel$ 
    end while
     $\mathbf{q} = \mathbf{q} + (\epsilon - t)\mathbf{p}$  {rest of position step}
     $\mathbf{p} = \mathbf{p} - \frac{\epsilon}{2} \nabla U_{|q|}(\mathbf{q})$  {1/2 momentum step}
end for
 $\mathbf{p} = -\mathbf{p}$ 
return  $((\mathbf{q}, \mathbf{p}), (\mathbf{q}_0, \mathbf{p}_0))$ 

```

---

**Algorithm 7** extend for NP-DHMC

**Input:** current state  $(\mathbf{q}, \mathbf{p})$ , initial state  $(\mathbf{q}_0, \mathbf{p}_0)$ , time  $t$ , family of potential energies  $U = \{U_n\}_{n \in \mathbb{N}}$  family of potential energies  $\{U_n\}_{n \in \mathbb{N}}$ , step size  $\epsilon$ , number of steps  $L$

**Output:** extended current state  $(\mathbf{q}, \mathbf{p})$ , extended initial state  $(\mathbf{q}_0, \mathbf{p}_0)$

---

```

while  $\mathbf{q} \notin \text{Dom}(U_{|q|})$  do
     $x_0 \sim \mathcal{N}(0, 1)$ 
    if  $|q| + 1 \in C$  then
         $y_0 \sim \mathcal{N}(0, 1)$  {Gaussian for continuous params}
    else
         $y_0 \sim \mathcal{L}(0, 1)$  {Laplace for discontinuous ones}
    end if
     $(x, y) = (x_0 + ty_0, y_0)$  {update to current time  $t$ }
     $(\mathbf{q}_0, \mathbf{p}_0) = (\mathbf{q}_0 \# [x_0], \mathbf{p}_0 \# [y_0])$ 
     $(\mathbf{q}, \mathbf{p}) = (\mathbf{q} \# [x], \mathbf{p} \# [y])$  {increment dimension}
end while
return  $((\mathbf{q}, \mathbf{p}), (\mathbf{q}_0, \mathbf{p}_0))$ 

```

---

**Algorithm 8** NP-DHMC Integrator  $\Psi_{\text{NP-Dis}}$ 

**Input:** current state  $(\mathbf{q}_0, \mathbf{p}_0)$ , family of potential energies  $\{U_n\}_{n \in \mathbb{N}}$ , step size  $\epsilon$ , number of steps  $L$ , discontinuous coordinates  $D$

**Output:** new state  $(\mathbf{q}, \mathbf{p})$  computed according to Hamiltonian dynamics, extended initial state  $(\mathbf{q}_0, \mathbf{p}_0)$

---

```

 $(\mathbf{q}, \mathbf{p}) = (\mathbf{q}_0, \mathbf{p}_0)$  {initialise}
 $\mathbf{q}' = \mathbf{q}_0$ 
 $\mathbf{p}' = \mathbf{p}_0$ 
 $N = |q_0|$ 
for  $i = 0$  to  $L$  do
     $\mathbf{p}_C = \mathbf{p}_C - \frac{\epsilon}{2} \nabla_{q_C} U_N(\mathbf{q})$ 
     $\mathbf{q}_C = \mathbf{q}_C + \frac{\epsilon}{2} \mathbf{p}_C$ 
    for  $j \in \text{randomlyPermute}(D)$  do
        if  $j < |q|$  then
             $\{|q| \text{ may have changed, so must check } j < |q|\}$ 
             $((\mathbf{q}, \mathbf{p}), (\mathbf{q}', \mathbf{p}')) = \text{coordIntegrator}((\mathbf{q}, \mathbf{p}), (\mathbf{q}', \mathbf{p}'), j, i\epsilon, \epsilon)$ 
        end if
    end for
     $N = |q|$ 
     $\mathbf{q}_C = \mathbf{q}_C + \frac{\epsilon}{2} \mathbf{p}_C$ 
     $\mathbf{p}_C = \mathbf{p}_C - \frac{\epsilon}{2} \nabla_{q_C} U_N(\mathbf{q})$ 
end for
 $\mathbf{p} = -\mathbf{p}$ 
return  $((\mathbf{q}, \mathbf{p}), (\mathbf{q}', \mathbf{p}'))$ 

```

---

**function** coordIntegrator $((\mathbf{q}, \mathbf{p}), (\mathbf{q}', \mathbf{p}'), j, t, \epsilon)$

```

 $\mathbf{q}^* = \mathbf{q}$ 
 $q_j^* = q_j^* + \epsilon \text{sign}(p_j)$ 
 $((\mathbf{q}^*, \mathbf{p}^*), (\mathbf{q}'^*, \mathbf{p}'^*)) = \text{extend}((\mathbf{q}^*, \mathbf{p}^*), (\mathbf{q}', \mathbf{p}'), t, U)$ 
 $\Delta U = U(\mathbf{q}^*) - U(\mathbf{q})$ 
if  $|p_j| > \Delta U$  then
     $(\mathbf{q}, \mathbf{p}) = (\mathbf{q}^*, \mathbf{p}^*)$  {enough kinetic energy to jump}
     $(\mathbf{q}', \mathbf{p}') = (\mathbf{q}'^*, \mathbf{p}'^*)$ 
     $p_j = p_j - \text{sign}(p_j)\Delta U$ 
else
     $p_j = -p_j$  {not enough kinetic energy, reflect}
end if
return  $((\mathbf{q}, \mathbf{p}), (\mathbf{q}', \mathbf{p}'))$ 

```

---



Note that the time derivative of  $\mathbf{q}$  only depends on the sign of the  $p_i$ 's. Hence, if the sign does not change, the change of  $\mathbf{q}$  can be computed, irrespective of the intermediate values of  $U_{|\mathbf{q}|}(\mathbf{q})$ . The integrator of discontinuous HMC (Nishimura et al., 2020) takes advantage of this for “discontinuous parameters”, i.e. parameters that  $U$  is not continuous in. Thus it can jump through multiple discontinuities of  $U$  without evaluating it at every boundary.

We adapt the integrator from (Nishimura et al., 2020) to NP-HMC. Following them, we assume for simplicity that each coordinate of the position space either corresponds to a continuous or discontinuous parameter, irrespective of which path is chosen. The set  $C$  records all the continuous parameters and  $D = \mathbb{N} \setminus C$  the discontinuous ones. We use a Gaussian distribution for the continuous parameters of the momentum vector and a Laplace distribution for the discontinuous parameters. Our integrator updates the continuous coordinates by half a step size just as before, but then the discontinuous ones are updated coordinate by coordinate, a technique called *operator splitting*. Afterwards, the continuous coordinates are updated by half a step size again. Algorithm 8 contains all the details.

Again, the main difference to the original algorithm is a call to `extend`. Note we also have to modify the `extend` function itself (given in Alg. 7) because some momentum coordinates have to be sampled from a Laplace distribution, and not a Gaussian as before.

### B.3. Efficiency Improvements

As touched upon in the main text, our implementation includes various performance improvements compared to the pseudocode presentation of NP-HMC.

(i) The `extend` function (Alg. 3) as presented may seem inefficient. While it terminates almost surely (thanks to Assumption 3), the expected number of iterations may be infinite. In practice, however, the density function  $w$  will arise from a probabilistic program, such as Listing 1. Therefore, to evaluate  $w$ , it would be natural to run the program. The length of  $\mathbf{q}$  returned by `extend` is exactly the number of `sample` statements encountered during the program’s execution. In particular, if the program has finite expected running time, then the same is true of `extend`.

(ii) On top of that, efficient implementations of NP-HMC will interleave the execution of the program with `extend`, by gradually extending  $\mathbf{q}$  (if necessary) at every encountered `sample` statement. This way, `extend` increases the running time only by a small constant factor.

(iii) In a similar vein, we do not have to compute the sum  $w_{\leq n}(\mathbf{q}) = \sum_{k=1}^n w(\mathbf{q}^{1\dots k})$  each time  $U_n = -\log w_{\leq n}$  is accessed. By the prefix property, only one of the summands of  $w_{\leq n}(\mathbf{q})$  is actually nonzero. Moreover, if  $w$  is given by a probabilistic program, then the weight computed during the execution of the program on  $\mathbf{q}$  is exactly this nonzero summand, assuming that the trace  $\mathbf{q}$  is long enough for a successful run (which the `extend` function ensures).

(iv) Another notable way our implementation differs from the algorithm presented above is that it not only extends a trace  $\mathbf{q}$  in `extend` (if necessary), but also trims it (if necessary) to the unique prefix  $\mathbf{q}'$  of  $\mathbf{q}$  with positive  $w(\mathbf{q}')$ . The dimension of  $\mathbf{p}$  is adjusted accordingly. This seems to work much better for certain examples, such as the geometric distribution described in Sec. 5. The reason is most likely that the unused suffix (which may have been adapted to the state before the current call of `extend`) is a hindrance when trying to extend to a different state later on.

## C. Proof of Correctness

In this section, we show that the NP-HMC algorithm is correct, in the sense that the Markov chain generated by iterating Alg. 1 converges to the target distribution  $\nu : A \mapsto \frac{1}{Z} \int_A w \, d\mu_{\mathbb{T}}$  where  $Z := \int_{\mathbb{T}} w \, d\mu_{\mathbb{T}}$ .

Henceforth, we assume that the density function  $w$  of the target distribution  $\nu$  is tree-representable and satisfies Assumptions 1, 2 and 3.

### C.1. An Equivalent Algorithm

We write Alg. 1 as the program `NPHMCstep` (Alg. 2 as `NPint` and Alg. 3 as `extend`) in Listing 2. We present input sample as `q0`; the density function as `w` and define potential energy  $U$ , which is a family of partial functions, as a function `U`, such that `U(n)` is a partial function denoting  $U_n$ ; step size as `ep`; and number of steps as `L`. We also assume the following primitive functions are implemented: `normal` is the sampling construct in the language which samples a real number from the standard normal distribution  $\mathcal{N}_1$ . `domain(f)` gives the domain of the partial function `f`. `pdfN(x, n)` gives the probability density of `x`



on the standard  $n$ -dimensional normal distribution. `cdfn`( $x$ ) gives the cumulative distribution of  $x$  on the standard normal distribution. `grad`( $f, x$ ) gives the gradient of the partial function  $f$  at  $x$  if defined and `None` if not.

The program `NPHMC` generates a Markov chain on  $\mathbb{T}$  by iterating `NPHMCstep`.

Instead of a direct proof, we consider an auxiliary program `eNPHMC` equivalent to `NPHMC` (in the sense of Prop. 22), which does not increase the dimension dynamically; instead it finds the smallest  $N$  such that all intermediate positions during the  $L$  leapfrog steps stay in the domain of  $U_N$ , and performs leapfrog steps as in standard HMC.

The program `eNPHMC` is given in Listing 4, which iterates `eNPHMCstep` to generate a Markov chain on `states` and then marginalise it using the helper function `supported` to obtain a Markov chain on  $\mathbb{T}$ . The program `validstate` determines whether the input state  $(q_0, p_0)$  goes beyond the domain of the potential energy  $U$  in  $L$  leapfrog steps, and the program `HMCint` is the leapfrog integrator of the standard HMC algorithm.

*Remark 21.* Programs in Listings 2 to 4 are given in Python syntax, but they can be translated into SPCF. First, note we can represent pairs and lists using Church encoding as follows:

$$\begin{aligned} \text{Pair}(\sigma, \tau) &:= \sigma \rightarrow \tau \rightarrow (\sigma \rightarrow \tau \rightarrow R) \rightarrow R & \text{List}(\sigma) &:= (\sigma \rightarrow R \rightarrow R) \rightarrow (R \rightarrow R) \\ \langle M, N \rangle &\equiv \lambda z.z.MN & [M_1, \dots, M_\ell] &\equiv \lambda f.x.f M_1(f M_2 \dots (f M_\ell)) \end{aligned}$$

Hence a state  $(q, p) \in \mathbb{R}^\ell \times \mathbb{R}^\ell$  can be encoded as a value  $[\langle \underline{q}_1, \underline{p}_1 \rangle, \dots, \langle \underline{q}_\ell, \underline{p}_\ell \rangle]$  with type `List(Pair(R, R))`.

Now we look at all the primitive functions used in the programs. It is easy to see that `cdfn`, `pdfn` and `log` are analytic functions. `len`, `append` and `sum` can be defined on Church lists. `grad` can be defined using the simple numerical differentiation method using analytic functions like subtraction and division. We can change `domain` in such a way that it takes  $q$  and  $w$  as inputs and tests whether `sum`(`[w(q[:i]) for i in range(len(q))]`) is zero (instead of testing whether  $q$  is in the domain of  $U(\text{len}(q))$ ).

Now we give a formal definition of equivalence. We say two SPCF programs are *equivalent* if they induce the same value and weight functions, as specified in App. A.2.1.

**Proposition 22.** `NPHMC` and `eNPHMC` are equivalent.

*Proof.* We give an informal explanation here.

First note that `NPHMCstep` is a Markov process on samples, and `eNPHMCstep` on states. However, it is easy to see that some minor changes to `NPHMCstep` and `NPHMC` make `NPHMCstep` a Markov process on states. Precisely, the following does not alter the meaning of program `NPHMC`:

- (1) Given a state  $(q_0, p_0)$  in `NPHMCstep`, apply `supported` to  $q_0$  at the start of initialisation and return the state  $(q_0, p_0)$  or  $(q, p)$  at the MH acceptance step.
- (2) In `NPHMC`, add the marginalisation step just like in `eNPHMC`.

Hence, it is enough to show that all steps in programs `NPHMCstep` and `eNPHMCstep` are equivalent, i.e. they give the same weight and value functions.

After the modification, `NPHMCstep` and `eNPHMCstep` have the same initialisation and MH acceptance step. So it remains to show that the NP-HMC integration as described in `NPint` behaves the same as searching for a valid initial state (step 2) and HMC integration (step 3) in `eNPHMCstep`.

In `NPHMCstep`,  $((q, p), (q_0, p_0)) = \text{NPint}((q_0, p_0), U, \text{ep}, L)$  “integrates” from the initial state  $(q_0, p_0)$  until it goes beyond the domain of  $U(\text{len}(q_0))$ , at which moment it `extends`.

While in `eNPHMCstep`, it increments the dimension of the state  $(q_0, p_0)$  until it has *just* enough dimension to “integrate” for time  $\text{ep} * L$  through  $U(\text{len}(q_0))$  without going beyond the domain of  $U(\text{len}(q_0))$ . This ensures the state  $(q_0, p_0)$  is safe to be an input to the standard HMC integrator `HMCint`.

Notice that given the same values for the samples, the resulting initial state  $(q_0, p_0)$  in `NPHMCstep` would be the same as that in `eNPHMCstep`. Hence, the proposal state  $(q, p)$  in both programs would be the same.  $\square$

*Remark 23.* The discussion in the proof of Prop. 22 argues informally that `NPHMC` and `eNPHMC` are equivalent. We outline a formal proof here. To show that `NPHMC` and `eNPHMC` are equivalent, we first demonstrate that one program can be obtained

Listing 2. Python code for **NPHMC**

```

def extend((q,p),(q0,p0),t,U):
    while q not in domain(U(len(q))):
        x0 = normal
        y0 = normal
        x = x0 + t*y0
        y = y0
        q0.append(x0)
        p0.append(y0)
        q.append(x)
        p.append(y)
    return ((q,p),(q0,p0))

def NPint((q0,p0),U,ep,L):
    q = q0
    p = p0
    for i in range(L):
        p = p - ep/2*grad(U(len(q0)),q)
        q = q + ep*p
        ((q,p),(q0,p0)) =
            extend((q,p),(q0,p0),i*ep,U)
        p = p - ep/2*grad(U(len(q0)),q)
    return ((q,p),(q0,p0))

def NPHMCstep(q0,w,ep,L):
    # initialisation
    p0 = [normal for i in range(len(q0))]
    U = lambda n: lambda q:
        -log(sum([w(q[:i]) for i in range(n)]))
    # NP-HMC integration
    ((q,p),(q0,p0)) = NPint((q0,p0),U,ep,L)
    # MH acceptance
    if cdfN(normal) < accept((q,p),(q0,p0),w):
        return supported(q,w)
    else:
        return supported(q0,w)

def NPHMC(q0,w,ep,L,M):
    S = [q0]
    for i in range(M):
        S.append(NPHMCstep(S[i],w,ep,L))
    return S
    
```

Listing 3. Python code for helper functions

```

# the MH acceptance ratio
def accept((q,p),(q0,p0),w):
    N = len(q)
    N_trunc = lambda q':
        sum([w(q'[:i]) for i in range(N)])
    weight = (N_trunc(q)*pdfN((q,p),2N))/
        (N_trunc(q0)*pdfN((q0,p0),2N))
    return min(1,weight)

# the w-supported prefix of q
def supported(q,w):
    k = 1
    while w(q[:k]) == 0 and k < len(q):
        k += 1
    return q[:k]
    
```

 Listing 4. Python code for **eNPHMC**

```

def validstate((q0,p0),U,ep,L):
    q = q0
    p = p0
    for i in range(L):
        p = p - ep/2*grad(U,q)
        q = q + ep*p
        if q not in domain(U):
            return False
        p = p - ep/2*grad(U,q)
    return True

def HMCint((q0,p0),U,ep,L):
    q = q0
    p = p0
    for i in range(L):
        p = p - ep/2*grad(U,q)
        q = q + ep*p
        p = p - ep/2*grad(U,q)
    # momentum flip
    p = -p
    return (q,p)

def eNPHMCstep((q0,p0),w,ep,L):
    # initialisation (step 1)
    q0 = supported(q0,w)
    p0 = [normal for i in range(len(q0))]
    U = lambda n: lambda q:
        -log(sum([w(q[:i]) for i in range(n)]))
    # search (step 2)
    while not validstate((q0,p0),U(len(q0)),ep,L):
        x0 = normal
        y0 = normal
        q0.append(x0)
        p0.append(y0)
    # HMC integration (step 3)
    (q,p) = HMCint((q0,p0),U(len(q0)),ep,L)
    # MH acceptance (step 4)
    if cdfN(normal) < accept((q,p),(q0,p0),w):
        return (q,p)
    else:
        return (q0,p0)

def eNPHMC(q0,w,ep,L,M):
    mc = [(q0,0)]
    for i in range(M):
        mc.append(eNPHMCstep(mc[i],w,ep,L))
    # marginalisation
    S = [supported(q,w) for (q,p) in mc]
    return S
    
```

form another by a series of meaning-preserving transformations (i.e. transformations that preserves the value and weight functions). After that we show that the convergence result (Thm. 5) is invariant over equivalent programs.

Since **NPHMC** and **eNPHMC** are equivalent, it is enough to show that **eNPHMC** is correct, i.e. generates a Markov chain that converges to the target distribution. We present a three-step proof.

1. We first identify the invariant distribution  $\pi$  of the Markov chain  $\{(\mathbf{q}^{(i)}, \mathbf{p}^{(i)})\}_{i \in \mathbb{N}}$  generated by iterating **eNPHMCstep**. (Eq. (1))
2. We then show that the *marginalised* chain  $\{f(\mathbf{q}^{(i)}, \mathbf{p}^{(i)})\}_{i \in \mathbb{N}}$  is invariant under the target distribution  $\nu$ , where  $f(\mathbf{q}, \mathbf{p})$  is the unique prefix of  $\mathbf{q}$  that has positive weight according to  $w$ . (Thm. 4)
3. Finally, we show this chain converges for a small enough step size  $\epsilon$ . (Thm. 5)

## C.2. Invariant Distribution

By iterating **eNPHMCstep**, a Markov chain  $\{(\mathbf{q}^{(i)}, \mathbf{p}^{(i)})\}_{i \in \mathbb{N}}$  is generated. We now analyse this Markov chain by studying its invariant distribution  $\pi$  and transition kernel.

Let  $(\mathbb{S}, \Sigma_{\mathbb{S}}, \mu_{\mathbb{S}})$  be the *state space* where  $\mathbb{S} := \uplus_{n \in \mathbb{N}} (\mathbb{R}^n \times \mathbb{R}^n)$ ,  $\Sigma_{\mathbb{S}} := \{\uplus_{n \in \mathbb{N}} U_n \mid U_n \in \mathcal{B}_{2n}\}$  and  $\mu_{\mathbb{S}}(\uplus_{n \in \mathbb{N}} U_n) := \sum_{n \in \mathbb{N}} (\mathcal{N}_n \times \mathcal{N}_n)(U_n)$ . It is easy to see that all output states in **eNPHMCstep**, and hence all elements of the Markov chain, is in  $\mathbb{S}$ .

However not all states have a positive weight. In fact not even the union of the support of invariant distributions of the fixed dimension HMC on each of the truncations works. This is because if **eNPHMCstep** returns  $(\mathbf{q}, \mathbf{p}) \in \mathbb{R}^{2k}$ , then it cannot return states of the form  $(\mathbf{q} + \mathbf{q}', \mathbf{p} + \mathbf{p}') \in \mathbb{R}^{2n}$ , which is a valid returning state for the fixed dimension HMC. Hence we define a subset of states which precisely capture all possible returning states of **eNPHMCstep**, and define a distribution on it.

We say a state  $(\mathbf{q}, \mathbf{p})$  is  $(\epsilon, L)$ -*valid* (or simply *valid* whenever the parameters  $\epsilon$  and  $L$  are clear from the context) if a particle starting from the state  $(\mathbf{q}, \mathbf{p})$  does not “fall beyond” the domain of  $U_{|\mathbf{q}|} := -\log w_{\leq |\mathbf{q}|}$  in the course of  $L$  discrete leapfrog steps of size  $\epsilon$ , and the states  $(\mathbf{q}^{1 \dots k}, \mathbf{p}^{1 \dots k})$  are not  $(\epsilon, L)$ -valid for all  $k < n$ .

Let  $\mathbb{S}^{\text{valid}}$  denote the set of all valid states and  $\mathbb{S}_n^{\text{valid}} := \mathbb{S}^{\text{valid}} \cap (\mathbb{R}^n \times \mathbb{R}^n)$  denote the the set of all  $n$ -dimension valid states. The program **validstate** verifies valid states, i.e **validstate** always returns True when the input state is valid.

Let  $\pi$  be a distribution on  $\mathbb{S}$  with density  $\zeta$  (with respect to  $\mu_{\mathbb{S}}$ ) given by

$$\zeta(\mathbf{q}, \mathbf{p}) := \begin{cases} \frac{1}{Z} w_{\leq |\mathbf{q}|}(\mathbf{q}) & \text{if } (\mathbf{q}, \mathbf{p}) \in \mathbb{S}^{\text{valid}}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Since the the position component of all valid states must have a  $w$ -supported prefix, the set of valid states can be written as

$$\mathbb{S}^{\text{valid}} = \bigcup_{n=1}^{\infty} \bigcup_{m=n}^{\infty} \{(\mathbf{q} + \mathbf{x}, \mathbf{y}) \in \mathbb{S}_m^{\text{valid}} \mid \mathbf{q} \in \text{Supp}^n(w), \mathbf{x} \in \mathbb{R}^{m-n}, \mathbf{y} \in \mathbb{R}^m\},$$

and hence the distribution  $\pi$  can be written as

$$\begin{aligned} \pi : X &\mapsto \int_X [(\mathbf{q}, \mathbf{p}) \in \mathbb{S}^{\text{valid}}] \cdot \frac{1}{Z} w_{\leq |\mathbf{q}|}(\mathbf{q}) \mu_{\mathbb{S}}(d(\mathbf{q}, \mathbf{p})) = \int_X [(\mathbf{q}, \mathbf{p}) \in \mathbb{S}^{\text{valid}}] \cdot \frac{1}{Z} \sum_{n=1}^{|\mathbf{q}|} w(\mathbf{q}^{1 \dots n}) \mu_{\mathbb{S}}(d(\mathbf{q}, \mathbf{p})) \\ &= \sum_{n=1}^{\infty} \sum_{m=n}^{\infty} \int_{\mathbb{R}^n} \int_{\mathbb{R}^{m-n}} \int_{\mathbb{R}^m} [(\mathbf{q} + \mathbf{x}, \mathbf{y}) \in X \cap \mathbb{S}_m^{\text{valid}}] \cdot \frac{1}{Z} w(\mathbf{q}) \mathcal{N}_m(d\mathbf{y}) \mathcal{N}_{m-n}(d\mathbf{x}) \mathcal{N}_n(d\mathbf{q}) \end{aligned} \quad (2)$$

We claim that  $\pi$  is the *invariant distribution* of the Markov chain determined by **eNPHMCstep**. The rest of this subsection is devoted to a proof of the claim.

For any state  $(\mathbf{q}, \mathbf{p}) \in \mathbb{S}$ , we write  $\llbracket (\mathbf{q}, \mathbf{p}) \rrbracket$  to be the term  $[\langle \mathbf{q}_1, \mathbf{p}_1 \rangle, \dots, \langle \mathbf{q}_{|\mathbf{q}|}, \mathbf{p}_{|\mathbf{q}|} \rangle]$  of type  $\text{List}(\text{Pair}(\mathbb{R}, \mathbb{R}))$ . Take a SPCF term  $M$  of type  $\{x : \text{List}(\text{Pair}(\mathbb{R}, \mathbb{R}))\} \vdash M : \text{List}(\text{Pair}(\mathbb{R}, \mathbb{R}))$ . We define a function  $v_M : \mathbb{S} \times \mathbb{T} \rightarrow \mathbb{S}$  such that  $\llbracket v_M(\mathbf{s}, \mathbf{t}) \rrbracket = \text{value}_{M[\llbracket \mathbf{s} \rrbracket/x]}(\mathbf{t})$ . Then, the *transition kernel*  $k_M : \mathbb{S} \times \Sigma_{\mathbb{S}} \rightarrow \mathbb{S}$  of  $M$  given by

$$k_M(\mathbf{s}, U) := \int_{v_M(\mathbf{s}, -)^{-1}(U)} \text{weight}_{M[\llbracket \mathbf{s} \rrbracket/x]} d\mu_{\mathbb{T}}.$$

returns the probability of  $M$  returning a state in  $U$  given the input  $\mathbf{s}$ .

We say  $M$  leaves the distribution  $\mu$  on  $\mathbb{S}$  invariant if for all  $U \in \Sigma_{\mathbb{S}}$ ,  $\int_{\mathbb{S}} k_M(\mathbf{s}, U) \mu(d\mathbf{s}) = \mu(U)$ .

### C.2.1. INITIALISATION AND SEARCH (STEPS 1 AND 2)

Given  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}}$  and  $X \in \Sigma_{\mathbb{S}}$ , where  $w(\mathbf{q}_0^{1\dots n}) > 0$ , the initialisation (step 1) of **eNPHMCstep** returns a pair of the  $w$ -supported prefix of  $\mathbf{q}_0$  and a randomly drawn momentum. Hence, its transition kernel  $k_1$  is given by  $k_1((\mathbf{q}_0, \mathbf{p}_0), X) := \int_{\mathbb{T}} [(\mathbf{q}_0^{1\dots n}, \mathbf{t}) \in X] \mu_{\mathbb{T}}(d\mathbf{t})$ . Note that  $\mathbf{p}_0$  (of the input state  $(\mathbf{q}_0, \mathbf{p}_0)$ ) is ignored by **eNPHMCstep**.

If the input state  $(\mathbf{q}_0, \mathbf{p}_0)$  is not a valid state, we have  $k_1((\mathbf{q}_0, \mathbf{p}_0), X) = 0$ . This is required for technical reasons but is excluded in the program **eNPHMCstep** for ease of readability. At it stands in Listing 4, **eNPHMCstep** does not care whether the input state is valid as long as it has a prefix which is  $w$ -supported. To define such a transition kernel for **eNPHMCstep**, we can simply call **validstate** on the input state at the start of initialisation and fail this execution if the input state is not valid.

After that, given  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}$  and  $X \in \Sigma_{\mathbb{S}}$  where  $w(\mathbf{q}_0^{1\dots n}) > 0$ , step 2 of **eNPHMCstep** searches for a valid state by repeating drawing from the standard normal distribution. We can write its transition kernel  $k_2$  as  $k_2((\mathbf{q}_0, \mathbf{p}_0), X) := \int_{\mathbb{T}} [(\mathbf{q}_0 + \mathbf{t}^{\text{odd}}, \mathbf{p}_0 + \mathbf{t}^{\text{even}}) \in X \cap \mathbb{S}^{\text{valid}}] \mu_{\mathbb{T}}(d\mathbf{t})$  where  $\mathbf{t}^{\text{odd}}$  and  $\mathbf{t}^{\text{even}}$  are subsequences of  $\mathbf{t}$  containing the values of odd and even indexes respectively.

For any  $X \in \Sigma_{\mathbb{T}}$ , the (combined) transition kernel  $k_{1,2}$  of steps 1 and 2 of **eNPHMCstep** is given by

$$\begin{aligned} k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), X) &= \int_{\mathbb{T}} \int_{\mathbb{T}} [(\mathbf{q}_0^{1\dots n} + \mathbf{t}'^{\text{odd}}, \mathbf{t} + \mathbf{t}'^{\text{even}}) \in X \cap \mathbb{S}^{\text{valid}}] \mu_{\mathbb{T}}(d\mathbf{t}') \mu_{\mathbb{T}}(d\mathbf{t}) \\ &= \int_{\mathbb{R}^n} \sum_{m=n}^{\infty} \int_{\mathbb{R}^{m-n}} \int_{\mathbb{R}^{m-n}} [(\mathbf{q}_0^{1\dots n} + \mathbf{t}'', \mathbf{t} + \mathbf{t}'') \in X \cap \mathbb{S}^{\text{valid}}] \mathcal{N}_{m-n}(d\mathbf{t}'') \mathcal{N}_{m-n}(d\mathbf{t}') \mathcal{N}_n(d\mathbf{t}) \\ &= \sum_{m=n}^{\infty} \int_{\mathbb{R}^m} \int_{\mathbb{R}^{m-n}} [(\mathbf{q}_0^{1\dots n} + \mathbf{x}, \mathbf{y}) \in X \cap \mathbb{S}^{\text{valid}}] \mathcal{N}_{m-n}(d\mathbf{x}) \mathcal{N}_m(d\mathbf{y}) \end{aligned}$$

if  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}}$ ; and  $k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), X) = 0$  otherwise.

**Proposition 24.** *The transition kernel is probabilistic, i.e.  $k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), \mathbb{S}) = k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), \mathbb{S}^{\text{valid}}) = 1$  for any valid state  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}}$ .*

*Proof.* Let  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}}$ . We can see  $k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), -)$  as the value measure of steps 1 and 2 of **eNPHMCstep** (with the initial states substituted by  $\llbracket (\mathbf{q}_0, \mathbf{p}_0) \rrbracket$ ) which does not contain  $\text{score}(-)$  as a subterm. Moreover, Assumption 3 ensures step 2 almost always terminates and returns a valid state. Hence, Prop. 9 tells us that  $k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), -)$  is probabilistic and  $k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), \mathbb{S}) = k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), \mathbb{S}^{\text{valid}}) = 1$ .  $\square$

**Proposition 25.**  *$\pi$  is invariant with respect to step 1 and 2 of **eNPHMCstep**.*

*Proof.* We aim to show:  $\int_{\mathbb{S}} k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), X) \pi(d(\mathbf{q}_0, \mathbf{p}_0)) = \pi(X)$  for any measurable set  $X \in \Sigma_{\mathbb{S}}$ .

$$\begin{aligned} &\int_{\mathbb{S}} k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), X) \pi(d(\mathbf{q}_0, \mathbf{p}_0)) = \int_{\mathbb{S}^{\text{valid}}} k_{1,2}((\mathbf{q}_0, \mathbf{p}_0), X) \pi(d(\mathbf{q}_0, \mathbf{p}_0)) \\ &= \left\{ \text{Eq. (2), definition of } k_{1,2} \text{ and writing } (\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}} \text{ as } (\mathbf{q} + \mathbf{x}, \mathbf{y}) \text{ where } \mathbf{q} \in \text{Supp}(w) \right\} \\ &\quad \sum_{n=1}^{\infty} \sum_{m=n}^{\infty} \int_{\mathbb{R}^n} \int_{\mathbb{R}^{m-n}} \int_{\mathbb{R}^m} \left( \sum_{k=n}^{\infty} \int_{\mathbb{R}^k} \int_{\mathbb{R}^{k-n}} [(\mathbf{q} + \mathbf{x}', \mathbf{y}') \in X \cap \mathbb{S}^{\text{valid}}] \mathcal{N}_{k-n}(d\mathbf{x}') \mathcal{N}_k(d\mathbf{y}') \right) \\ &\quad \quad \left( [(\mathbf{q} + \mathbf{x}, \mathbf{y}) \in \mathbb{S}^{\text{valid}}] \cdot \frac{1}{Z} w(\mathbf{q}) \right) \mathcal{N}_m(d\mathbf{y}) \mathcal{N}_{m-n}(d\mathbf{x}) \mathcal{N}_n(d\mathbf{q}) \\ &= \left\{ \text{Rearranging (allowed because everything is nonnegative)} \right\} \\ &\quad \sum_{n=1}^{\infty} \sum_{k=n}^{\infty} \int_{\mathbb{R}^n} \int_{\mathbb{R}^{k-n}} \int_{\mathbb{R}^k} [(\mathbf{q} + \mathbf{x}', \mathbf{y}') \in X \cap \mathbb{S}^{\text{valid}}] \cdot \frac{1}{Z} w(\mathbf{q}) \\ &\quad \quad \left( \sum_{m=n}^{\infty} \int_{\mathbb{R}^{m-n}} \int_{\mathbb{R}^m} [(\mathbf{q} + \mathbf{x}, \mathbf{y}) \in \mathbb{S}^{\text{valid}}] \mathcal{N}_m(d\mathbf{y}) \mathcal{N}_{m-n}(d\mathbf{x}) \right) \mathcal{N}_k(d\mathbf{y}') \mathcal{N}_{k-n}(d\mathbf{x}') \mathcal{N}_n(d\mathbf{q}) \\ &= \left\{ \text{Definition of } k_{1,2} \text{ where } (\hat{\mathbf{q}}, \hat{\mathbf{p}}) \text{ is an arbitrary valid state such that } \hat{\mathbf{q}}^{1\dots n} = \mathbf{q} \right\} \end{aligned}$$

$$\begin{aligned}
 & \sum_{n=1}^{\infty} \sum_{k=n}^{\infty} \int_{\mathbb{R}^n} \int_{\mathbb{R}^{k-n}} \int_{\mathbb{R}^k} [(\mathbf{q} \# \mathbf{x}', \mathbf{y}') \in X \cap \mathbb{S}^{\text{valid}}] \cdot \frac{1}{Z} w(\mathbf{q}) \cdot k_{1,2}((\hat{\mathbf{q}}, \hat{\mathbf{p}}), \mathbb{S}^{\text{valid}}) \mathcal{N}_k(d\mathbf{y}') \mathcal{N}_{k-n}(d\mathbf{x}') \mathcal{N}_n(d\mathbf{q}) \\
 = & \quad \{ \text{Definition of } \zeta \text{ and Prop. 24 for some valid state } (\hat{\mathbf{q}}, \hat{\mathbf{p}}) \} \\
 & \int_X \zeta d\mu_{\mathbb{S}}
 \end{aligned}$$

□

### C.2.2. INTEGRATION AND ACCEPTANCE (STEPS 3 AND 4)

Let  $(\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}$  and  $X \in \Sigma_{\mathbb{S}}$ . Now we check that the HMC integration (step 3) and acceptance (step 4) preserve the invariant distribution  $\pi$ .

Similar to HMC, the transition kernel for steps 3 and 4 is given by

$$k_{3,4}((\mathbf{q}_0, \mathbf{p}_0), X) = \begin{cases} \alpha(\mathbf{q}_0, \mathbf{p}_0) \cdot [\Psi_{|\mathbf{q}_0|}(\mathbf{q}_0, \mathbf{p}_0) \in X] + (1 - \alpha(\mathbf{q}_0, \mathbf{p}_0)) \cdot [(\mathbf{q}_0, \mathbf{p}_0) \in X] & \text{if } (\mathbf{q}_0, \mathbf{p}_0) \in \mathbb{S}^{\text{valid}}, \\ 0 & \text{otherwise.} \end{cases}$$

where  $\alpha(\mathbf{q}_0, \mathbf{p}_0) = \min\{1, \frac{w_{\leq N}(\mathbf{q}) \cdot \varphi_{2N}(\mathbf{q}, \mathbf{p})}{w_{\leq N}(\mathbf{q}_0) \cdot \varphi_{2N}(\mathbf{q}_0, \mathbf{p}_0)}\}$  for  $N = |\mathbf{q}_0|$  and  $(\mathbf{q}, \mathbf{p}) = \Psi_N(\mathbf{q}_0, \mathbf{p}_0)$ .

**Proposition 26.** *The HMC integrator  $\Psi_n$  with respect to the potential energy  $U_n$  is volume preserving with respect to  $\text{Leb}_{2n}$  (i.e.  $\Psi_{n*} \text{Leb}_{2n} = \text{Leb}_{2n}$ ) and reversible (i.e.  $\Psi_n = \Psi_n^{-1}$ ) on  $\mathbb{S}_n^{\text{valid}}$ .*

*Proof.* Since measurable subsets of and states in  $\mathbb{S}_n^{\text{valid}}$  are also in the  $n$ -dimension Euclidean Space, and  $\Psi_n$  always map valid states to valid states, Prop. 14 is sufficient. □

**Proposition 27.**  *$\pi$  is invariant against integration and acceptance (steps 3 and 4) of **eNPHMCstep**.*

*Proof.* We aim to show:  $\int_{\mathbb{S}} k_{3,4}(x, X) \pi(dx) = \pi(X)$  for all  $X \in \Sigma_{\mathbb{S}}$ . By Prop. 26, for all  $n$ , HMC integrator  $\Psi_n$  is volume preserving against  $\text{Leb}_{2n}$  and reversible on  $\mathbb{S}_n^{\text{valid}}$ . Hence, we have

$$\begin{aligned}
 \int_{\mathbb{S}} k_{3,4}(x, X) \pi(dx) &= \int_{\mathbb{S}^{\text{valid}}} k_{3,4}(x, X) \pi(dx) = \sum_{n=1}^{\infty} \int_{\mathbb{S}_n^{\text{valid}}} k_{3,4}(x, X) \cdot \zeta(x) (\mathcal{N}_n \times \mathcal{N}_n)(dx) \\
 &= \int_X \zeta d\mu_{\mathbb{S}} + \sum_{n=1}^{\infty} \left( \int_{\mathbb{S}_n^{\text{valid}}} [\Psi_n(x) \in X \cap \mathbb{S}_n^{\text{valid}}] \cdot \alpha(x) \cdot \zeta(x) \cdot \varphi_{2n}(x) \text{Leb}_{2n}(dx) \right. \\
 &\quad \left. - \int_{\mathbb{S}_n^{\text{valid}}} [x \in X \cap \mathbb{S}_n^{\text{valid}}] \cdot \alpha(x) \cdot \zeta(x) \cdot \varphi_{2n}(x) \text{Leb}_{2n}(dx) \right)
 \end{aligned}$$

The second and third integrals are the same since the pushforward measure of  $\text{Leb}_{2n}$  along the integrator  $\Psi_n$  is the same as  $\text{Leb}_{2n}$  ( $\Psi_n$  is volume preserving on  $\mathbb{S}_n^{\text{valid}}$ ) for all  $n$  and  $\alpha(x) \cdot \zeta(x) \cdot \varphi_{2n}(x) = \alpha(\Psi_n(x)) \cdot \zeta(\Psi_n(x)) \cdot \varphi_{2n}(\Psi_n(x))$  for all  $x \in \mathbb{S}_n^{\text{valid}}$  (all  $\Psi_n$  are reversible on  $\mathbb{S}_n^{\text{valid}}$ ). □

Since the transition kernel  $P$  of **eNPHMCstep** is the composition of  $k_{1,2}$  and  $k_{3,4}$ , i.e.  $P(x, X) := \int_{\mathbb{S}} k_{3,4}(x', X) k_{1,2}(x, dx')$  for  $x \in \mathbb{S}$  and  $X \in \Sigma_{\mathbb{S}}$ , and both  $k_{1,2}$  and  $k_{3,4}$  are invariant against  $\pi$  (Propositions 25 and 27), we conclude with the following lemma.

**Lemma 28.**  *$\pi$  is the invariant distribution of the Markov chain generated by iterating **eNPHMCstep**.*

### C.3. Marginalised Markov Chains

It is important to notice that the Markov chain  $\{(\mathbf{q}_i, \mathbf{p}_i)\}_{i \in \mathbb{N}}$  generated by iterating **eNPHMCstep** with invariant distribution  $\pi$  is *not* the samples we are seeking. The chain we are in fact interested in is the *marginalised* chain  $\{f(\mathbf{q}_i, \mathbf{p}_i)\}_{i \in \mathbb{N}}$  where the measurable<sup>10</sup> function  $f$  finds the prefix of  $\mathbf{q}$  which is  $w$ -supported, formally defined as

$$\begin{aligned}
 f : \mathbb{S}^{\text{valid}} &\longrightarrow \mathbb{T} \\
 (\mathbf{q}, \mathbf{p}) &\longmapsto \mathbf{q}^{1 \dots n} \quad \text{for } \mathbf{q}^{1 \dots n} \in \text{Supp}(w).
 \end{aligned}$$

<sup>10</sup>For any measurable set  $A \in \Sigma_{\mathbb{T}}$ ,  $f^{-1}(A) = (\bigcup_{n=1}^{\infty} \bigcup_{m=n}^{\infty} ((A \cap \mathbb{R}^n) \times \mathbb{R}^{m-n}) \times \mathbb{R}^m) \cap \mathbb{S}^{\text{valid}}$  is measurable in  $\mathbb{S}$ .

This function is realised by the **supported** program in Listing 3.

In this section we show that this marginalised chain has the target distribution  $\nu$  as its invariant distribution. Let  $Q : \text{Supp}(w) \times \Sigma_{\mathbb{T}} \rightarrow \mathbb{R}_{\geq 0}$  be the transition kernel of this marginalised chain. We can write it as  $Q(f(x), A) = P(x, f^{-1}(A))$  for  $x \in \mathbb{S}^{\text{valid}}$  and  $A \in \Sigma_{\mathbb{T}}$ .

*Remark 29.* In the standard HMC algorithm, the function  $f$  would simply be the first projection, and it is trivial to check that the pushforward of the invariant distribution along the first projection is exactly the target distribution. Hence this step tends to be skipped in the correctness proof of HMC (Neal, 2011; Bou-Rabee & Sanz-Serna, 2018).

**Lemma 30.** Writing  $\mathbb{S}_{\leq n}^{\text{valid}} := \bigcup_{k=1}^n \mathbb{S}_k^{\text{valid}}$ , we let  $\pi_n$  be a probability distribution on measurable space  $(\mathbb{R}^{2n}, \mathcal{B}^{2n}, \mathcal{N}_{2n})$  given by

$$\pi_n(X) := \int_X \frac{1}{Z_n} w_{\leq n}(\mathbf{q}) \mathcal{N}_{2n}(d(\mathbf{q}, \mathbf{p})) \quad \text{where } Z_n := \int_{\mathbb{R}^{2n}} w_{\leq n} d\mathcal{N}_n \text{ and } X \in \mathcal{B}_{2n}.$$

- (1)  $\pi(\mathbb{S} \setminus \mathbb{S}_{\leq n}^{\text{valid}}) \rightarrow 0$  as  $n \rightarrow \infty$ .
- (2) For  $m \geq n$ ,  $Z_n \cdot \pi_n = Z_m \cdot e_*^{(m,n)} \pi_m$  on  $\mathbb{S}_n^{\text{valid}}$  where  $e^{(m,n)} : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}^n \times \mathbb{R}^n$  with  $e^{(m,n)}(\mathbf{q}, \mathbf{p}) = (\mathbf{q}^{1\dots n}, \mathbf{p}^{1\dots n})$ .
- (3)  $Z \cdot \pi = Z_n \cdot g_*^{(n)} \pi_n$  on  $\mathbb{S}_{\leq n}^{\text{valid}}$  where  $g^{(n)} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{S}_{\leq n}^{\text{valid}}$  such that  $g^{(n)}(\mathbf{q}, \mathbf{p}) = (\mathbf{q}^{1\dots k}, \mathbf{p}^{1\dots k}) \in \mathbb{S}_{\leq n}^{\text{valid}}$ .

*Proof.* (1)  $\pi$  is an invariant distribution, and hence it is probabilistic. The sum  $\sum_{n=1}^{\infty} \pi(\mathbb{S}_n^{\text{valid}})$  which equals  $\pi(\bigcup_{n=1}^{\infty} \mathbb{S}_n^{\text{valid}}) = \pi(\mathbb{S}^{\text{valid}})$  must converge. Hence  $\pi(\mathbb{S} \setminus \mathbb{S}_{\leq n}^{\text{valid}}) = \sum_{i=n+1}^{\infty} \pi(\mathbb{S}_i^{\text{valid}}) \rightarrow 0$  as  $n \rightarrow \infty$ .

(2) Simple to show.

(3) Let  $X$  be a measurable subset of  $\mathbb{S}_{\leq n}^{\text{valid}}$ . Then,

$$\begin{aligned} Z \cdot \pi(X) &= \sum_{k=1}^n Z_k \cdot \pi_k(X \cap \mathbb{S}_k^{\text{valid}}) = Z_n \sum_{k=1}^n e_*^{(n,k)} \pi_n(X \cap \mathbb{S}_k^{\text{valid}}) \\ &= Z_n \cdot \pi_n\left(\bigcup_{k=1}^n \{(\mathbf{q}, \mathbf{p}) \in \mathbb{R}^{2n} \mid (\mathbf{q}^{1\dots k}, \mathbf{p}^{1\dots k}) \in X \cap \mathbb{S}_k^{\text{valid}}\}\right) \\ &= Z_n \cdot g_*^{(n)} \pi_n(X). \end{aligned}$$

□

**Theorem 4.** Given Assumptions 1, 2 and 3, the target distribution  $\nu$  is the invariant distribution of the Markov chain generated by iterating Alg. 1.

*Proof.* For any  $A \in \Sigma_{\mathbb{T}}$ , if (1)  $\nu = f_* \pi$  on  $\mathbb{T}$  and (2)  $\mu_{\mathbb{T}} = f_* \mu_{\mathbb{S}}$  on  $\text{Supp}(w)$ , then

$$\begin{aligned} \nu(A) &= f_* \pi(A) = \int_{\mathbb{S}} P(x, f^{-1}(A)) \mu_{\mathbb{S}}(dx) && \text{(Lem. 28)} \\ &= \int_{\mathbb{S}^{\text{valid}}} P(x, f^{-1}(A)) \mu_{\mathbb{S}}(dx) = \int_{\mathbb{S}^{\text{valid}}} Q(f(x), A) \mu_{\mathbb{S}}(dx) \\ &= \int_{\text{Supp}(w)} Q(q, A) f_* \mu_{\mathbb{S}}(dq) = \int_{\text{Supp}(w)} Q(q, A) \mu_{\mathbb{T}}(dq) = \int_{\mathbb{T}} Q(q, A) \mu_{\mathbb{T}}(dq). \end{aligned}$$

Hence it is enough to show (1) and (2).

(1) Let  $A \subseteq \mathbb{R}^n$  be a measurable set on  $\mathbb{T}$  and  $\delta > 0$ . Then partitioning  $f^{-1}(A) = \{(\mathbf{q}, \mathbf{p}) \in \mathbb{S}^{\text{valid}} \mid \mathbf{q}^{1\dots n} \in A\}$  using  $\mathbb{S}_k^{\text{valid}}$ , we have for sufficiently large  $m$ ,

$$\begin{aligned} f_* \pi(A) &= \pi\left(\bigcup_{k=1}^m f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\right) + \pi\left(\bigcup_{k=m+1}^{\infty} f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\right) \\ &< \frac{Z_m}{Z} \cdot g_*^{(m)} \pi_m\left(\bigcup_{k=1}^m f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\right) + \delta && \text{(by Lem. 30 (1) and (3))} \\ &\leq \frac{Z_m}{Z} \cdot \pi_m(A \times \mathbb{R}^{m-n} \times \mathbb{R}^m) + \delta \\ &= \nu(A) + \delta. \end{aligned}$$



For any measurable set  $A \in \Sigma_{\mathbb{T}}$ , we have  $f_*\pi(A) = \sum_{n=1}^{\infty} f_*\pi(A \cap \mathbb{R}^n) \leq \sum_{n=1}^{\infty} \nu(A \cap \mathbb{R}^n) = \nu(A)$ . Since both  $\nu$  and  $\pi$  are probability distributions, we also have  $\nu(A) = 1 - \nu(\mathbb{T} \setminus A) \leq 1 - f_*\pi(\mathbb{T} \setminus A) = 1 - (1 - f_*\pi(A)) = f_*\pi(A)$ . Hence  $f_*\pi = \nu$  on  $\mathbb{T}$ .

(2) Similarly, let  $A \subseteq \text{Supp}^n(w)$  be a measurable set on  $\mathbb{T}$  and  $\delta > 0$ . Then for sufficiently large  $m$ , we must have  $\mu_{\mathbb{S}}(\bigcup_{k=m+1}^{\infty} \mathbb{S}_k^{\text{valid}}) = \mu_{\mathbb{S}}(\mathbb{S}^{\text{valid}} \setminus \mathbb{S}_{\leq m}^{\text{valid}}) < \delta$ . Hence,

$$\begin{aligned} f_*\mu_{\mathbb{S}}(A) &= \mu_{\mathbb{S}}\left(\bigcup_{k=1}^m f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\right) + \mu_{\mathbb{S}}\left(\bigcup_{k=m+1}^{\infty} f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\right) \\ &< \sum_{k=1}^m \mathcal{N}_{2k}(f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}) + \delta \\ &= \sum_{k=1}^m \mathcal{N}_{2m}(\{(q, p) \in \mathbb{R}^{2m} \mid (q^{1\dots k}, p^{1\dots k}) \in f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\}) + \delta \\ &= \mathcal{N}_{2m}(\bigcup_{k=1}^m \{(q, p) \in \mathbb{R}^{2m} \mid (q^{1\dots k}, p^{1\dots k}) \in f^{-1}(A) \cap \mathbb{S}_k^{\text{valid}}\}) + \delta \\ &\leq \mathcal{N}_{2m}(A \times \mathbb{R}^{m-n} \times \mathbb{R}^m) + \delta \\ &= \mu_{\mathbb{T}}(A) + \delta. \end{aligned}$$

Then the proof proceeds as in (1). □

#### C.4. Convergence

Last but not least, we check for the convergence of the marginalised chain to the target distribution  $\nu$ .

As shown in Ex. 16, it is not trivial that the standard HMC algorithm converges. The same can be said of the NP-HMC algorithm. Recall the conditions on the transition kernel to ensure convergence.

**Lemma 19** (Tierney (1994), Theorem 1 and Corollary 2). *If the transition kernel  $Q$  with invariant distribution  $\nu$  is  $\nu$ -irreducible and  $\nu$ -aperiodic, then for all  $q$ ,  $\lim_{n \rightarrow \infty} \|Q^n(q, -) - \nu\| = 0$ .*

Recall  $Q$  is the transition kernel of the Markov chain generated by iterating Alg. 1 on  $\text{Supp}(w)$ . In Thm. 4, we have shown that  $Q$  has invariant distribution  $\nu$ . Hence, most of this section is devoted to searching for sufficient conditions (Def. 35) in order to show that the transition kernel  $Q$  is  $\nu$ -irreducible (Lem. 36) and aperiodic (Lem. 37). We conclude in Thm. 5 that this Markov chain converges to the target distribution  $\nu$ .

We start by extending the result in (Cances et al., 2007) in two ways:

1. The density function is only continuously differentiable *almost everywhere*.
2. The position space is the target space  $\mathbb{T}$ .

Let  $\mathcal{U}$  be the collection of measurable subsets of  $\mathbb{T}$  with the property that their boundary has measure zero. Formally,  $\mathcal{U} := \{A \in \Sigma_{\mathbb{T}} \mid \mu_{\mathbb{T}}(\partial A) = 0\}$ . Not every set in  $\Sigma_{\mathbb{T}}$  satisfies this property. A typical example would be the fat Cantor set. It is easy to see that  $\mathcal{U}$  is closed under complementation. Moreover, for any non-null set  $A$  in  $\mathcal{U}$ , its interior  $\overset{\circ}{A}$  is non-empty.

We assume the density function  $w : \mathbb{T} \rightarrow \mathbb{R}_{\geq 0}$  is continuously differentiable on a non-null set  $A \in \mathcal{U}$ . We start by showing that the Markov chain can almost surely move between  $w$ -supported elements in  $A$ .

**Lemma 31.** *Assume  $w$  is continuously differentiable on a non-null set  $A \in \mathcal{U}$  and  $\{U_n\}$  is uniformly bounded above (i.e. there is an upper bound  $M$ , where  $U_n(q) < M$  for all  $q \in \text{Dom}(U_n)$  for all  $n \in \mathbb{N}$ ). For almost all  $a, b \in A \cap \text{Supp}(w)$ , there exists some  $k \geq \max\{|a|, |b|\}$  and  $p \in \mathbb{R}^k$  such that  $\text{proj}_1(\Psi_k(a + \mathbf{0}^{1\dots k-|a|}, p))^{1\dots |b|} = b$ , where  $\text{proj}_1(q, p) = q$ .*

*Proof.* Define a function  $V$  on the sequence space  $\mathbb{R}^{\omega}$ , which is a Fréchet space with a family of semi-norms  $\{\|\cdot\|_k\}_{k \in \mathbb{N}}$  where  $\|\mathbf{x}\|_k = |\mathbf{x}_k|$ , as

$$\begin{aligned} V : \quad \mathbb{R}^{\omega} &\longrightarrow \mathbb{R}_{\geq 0} \\ \mathbf{x} &\longmapsto -\log \sum_{k=1}^{\infty} w(\mathbf{x}^{1\dots k}). \end{aligned}$$



$V$  is well-defined thanks to Assumption 3. Since  $w$  is continuously differentiable on  $A$ ,  $V$  is continuously differentiable on the non-empty open set  $\hat{A} := \bigcup_{n=1}^{\infty} (\hat{A} \cap \mathbb{R}^n) \times \mathbb{R}^{\omega}$ . Moreover,  $V$  must be bounded above, say by some  $M$ .

Now we consider the minimization of the function  $S_{\epsilon} : (\mathbb{R}^{\omega})^{L+1} \rightarrow \mathbb{R}^{\omega}$  where  $\epsilon$  is the leapfrog step size,

$$(S_{\epsilon}(\mathbf{q}^0, \dots, \mathbf{q}^L))_k := \epsilon \sum_{i=0}^{L-1} \left( \frac{1}{2} \left( \frac{\mathbf{q}_k^{i+1} - \mathbf{q}_k^i}{\epsilon} \right)^2 - \frac{V(\mathbf{q}^{i+1}) + V(\mathbf{q}^i)}{2} \right) \quad \text{for all } k \in \mathbb{N}$$

where  $\mathbf{q}^0 = \mathbf{a} \# \mathbf{0}$  and  $\mathbf{q}^L = \mathbf{b} \# \mathbf{0}$ . Since  $V$  is bounded above by  $M$ , for all  $\phi \in (\mathbb{R}^{\omega})^{L+1}$ , each component of  $S_{\epsilon}(\phi) \in \mathbb{R}^{\omega}$  is bounded below by  $-\epsilon(L-1)M$  (i.e.  $\forall k \in \mathbb{N}, S_{\epsilon}(\phi)_k > -\epsilon(L-1)M$ ). Hence,  $S_{\epsilon}$  is bounded below. By the completeness of  $\mathbb{R}^{\omega}$ ,  $\inf S_{\epsilon} \in \mathbb{R}^{\omega}$  exists.

Consider a minimising sequence  $\{\phi_n\}_{n \in \mathbb{N}}$  on  $(\mathbb{R}^{\omega})^{L+1}$  where  $S_{\epsilon}(\phi_{n+1})_k < S_{\epsilon}(\phi_n)_k$  for all  $n, k \in \mathbb{N}$  and  $S_{\epsilon}(\phi_n) \rightarrow \inf S_{\epsilon}$  as  $n \rightarrow \infty$ . Writing the sequence as  $\{(\mathbf{q}^{0,n}, \dots, \mathbf{q}^{L,n})\}_{n \in \mathbb{N}}$ , we say it is bounded on  $(\mathbb{R}^{\omega})^{L+1}$  if and only if for each  $i = 0, \dots, L$ ,  $\{\mathbf{q}^{i,n}\}_{n \in \mathbb{N}}$  is a bounded set on  $\mathbb{R}^{\omega}$  which is equivalent to saying that for each  $i = 0, \dots, L$  and for all  $k \in \mathbb{N}$ ,  $\{\|\mathbf{q}^{i,n}\|_k\}_{n \in \mathbb{N}}$  is bounded on  $\mathbb{R}$ . It is easy to see that for all  $n \in \mathbb{N}$  and  $i = 1, \dots, L$ ,  $\|\mathbf{q}^{i+1,n} - \mathbf{q}^{i,n}\|_k \leq 2\epsilon S_{\epsilon}(\phi_0) + 2\epsilon^2 LM$  and  $\|\mathbf{q}^{1,n}\|_k \leq 2\epsilon S_{\epsilon}(\phi_0) + 2\epsilon^2 LM + \|\mathbf{q}^0\|_k$ , so for any  $i = 0, \dots, L$  and  $k \in \mathbb{N}$ ,  $\{\|\mathbf{q}^{i,n}\|_k\}_{n \in \mathbb{N}}$  is bounded and hence the sequence  $\{\phi_n\}_{n \in \mathbb{N}}$  is bounded. Moreover, its closure  $\Phi := \overline{\{\phi_n\}_{n \in \mathbb{N}}}$  is bounded and closed.

Note that the Fréchet space  $\mathbb{R}^{\omega}$  is a quasi-complete nuclear space and has the Heine–Borel property, i.e. all closed and bounded set is compact. So, the set  $\Phi$  is compact. Moreover, since  $\mathbb{R}^{\omega}$  is completely metrisable, the compact set  $\Phi$  is also sequentially compact, i.e. every sequence in  $\Phi$  has a subsequence converging to a point in  $\Phi$ . Hence  $\{\phi_n\}_{n \in \mathbb{N}} \subseteq \Phi$  must have a subsequence  $\{\phi_{n_k}\}_{k \in \mathbb{N}}$  which converges to some point  $\bar{\phi}$  in  $\Phi$ .

We claim that  $\bar{\phi}$  is almost surely in  $\hat{A}^{L+1}$ . We show that the set  $(\mathbb{R}^{\omega})^{L+1} \setminus \hat{A}^{L+1}$  has measure zero. First note that by Assumption 2,  $w$  is continuously differentiable almost everywhere and hence  $\mathbb{T} \setminus A$  is a null set. Moreover, by the definition of  $A \in \mathcal{U}$ ,  $\mathbb{T} \setminus \hat{A}$  is also a null set. Then this implies the set of infinite sequences with no prefixes in  $\hat{A}$  has measure zero, i.e.  $\mathbb{R}^{\omega} \setminus \hat{A}$  is a null set. Hence  $(\mathbb{R}^{\omega})^{L+1} \setminus \hat{A}^{L+1} = \{(\mathbf{q}^0, \dots, \mathbf{q}^L) \in (\mathbb{R}^{\omega})^{L+1} \mid \exists i. \mathbf{q}^i \notin \hat{A}\} = \bigcup_{i=0}^L (\mathbb{R}^{\omega})^i \times (\mathbb{R}^{\omega} \setminus \hat{A}) \times (\mathbb{R}^{\omega})^{L-i}$  has zero measure.

Since  $\bar{\phi}$  is constrained by  $\mathbf{q}^0 = \mathbf{a} \# \mathbf{0}$  and  $\mathbf{q}^L = \mathbf{b} \# \mathbf{0}$ , there can only be a null set of  $\mathbf{a}, \mathbf{b} \in A \cap \text{Supp}(w)$  which induces  $\bar{\phi}$  in the null set  $(\mathbb{R}^{\omega})^{L+1} \setminus \hat{A}^{L+1}$ . Hence  $\bar{\phi}$  is almost surely in  $\hat{A}^{L+1}$ .

Assume  $\bar{\phi}$  is in  $\hat{A}^{L+1}$ . Since  $V$  is continuously differentiable on  $\hat{A}$ , so is  $S_{\epsilon}$  on  $\hat{A}^{L+1}$ . By the continuity of  $S_{\epsilon}$ , we have  $\inf S_{\epsilon} = \lim_{k \rightarrow \infty} S_{\epsilon}(\phi_{n_k}) = S_{\epsilon}(\lim_{k \rightarrow \infty} \phi_{n_k}) = S_{\epsilon}(\bar{\phi})$ , so  $S_{\epsilon}$  attains its infimum on  $\hat{A}^{L+1}$ .

By Prop. 32, the gradient of  $S_{\epsilon}$  at its infimum  $\bar{\phi} = (\bar{\mathbf{q}}^0, \dots, \bar{\mathbf{q}}^L)$  is  $\mathbf{0}$ . Hence  $\bar{\mathbf{q}}^0 = \mathbf{a} \# \mathbf{0}$ ,  $\bar{\mathbf{q}}^L = \mathbf{b} \# \mathbf{0}$  and

$$\bar{\mathbf{q}}^{i+1} = 2\bar{\mathbf{q}}^i - \bar{\mathbf{q}}^{i-1} - \epsilon^2 \nabla V(\bar{\mathbf{q}}^i) \quad \text{for } i = 1, \dots, L-1$$

which is the solution to the leapfrog steps. In other words, the infimum  $\bar{\phi}$  gives a path from  $\mathbf{a} \# \mathbf{0}$  to  $\mathbf{b} \# \mathbf{0}$  via the leapfrog trajectory with initial momentum  $\mathbf{p} = \frac{1}{\epsilon}(\bar{\mathbf{q}}^1 - \mathbf{a} \# \mathbf{0}) + \frac{\epsilon}{2} \nabla V(\mathbf{a} \# \mathbf{0})$ .

Last but not least, let  $k$  be the maximum of  $k_i$ 's where  $w(\bar{\mathbf{q}}^{1 \dots k_i}) > 0$  for all  $i = 0, \dots, L$ . Then it is easy to see that  $\text{proj}_1(\Psi_k(\mathbf{a} \# \mathbf{0}^{1 \dots k-|a|}), \mathbf{p}^{1 \dots k})^{1 \dots |b|} = \mathbf{b}$ .  $\square$

**Proposition 32.** *Let  $f : \mathbb{R}^{\omega} \rightarrow \mathbb{R}^{\omega}$  be a function with infimum at  $x_0 \in \mathbb{R}^{\omega}$  and is continuously differentiable on  $A \subseteq \mathbb{R}^{\omega}$  where  $x_0 \in A$ , then  $\nabla f(x_0)$  is the zero map, i.e.  $\nabla f(x_0)(h) = \mathbf{0}$  for all  $h \in \mathbb{R}^{\omega}$ .*

*Proof.* First note that  $f$  is continuously differentiable at  $x_0 \in A$  means that for any  $\epsilon > 0$  there exists an  $\delta > 0$  such that for any  $k \in \mathbb{N}$  and  $x \in \mathbb{R}^{\omega}$  such that  $\|x - x_0\|_k < \delta$  implies  $\frac{\|f(x) - f(x_0) - L(x - x_0)\|_{\ell}}{\|x - x_0\|_k} < \epsilon$  for all  $\ell \in \mathbb{N}$ , where  $L : \mathbb{R}^{\omega} \rightarrow \mathbb{R}^{\omega}$  is the bounded linear map defined as  $L := (Df)(x_0)$ .<sup>11</sup>

Assume for contradiction that  $L$  is not a zero map. i.e. There exists some  $h \in \mathbb{R}^{\omega}$  such that  $Lh \neq \mathbf{0}$ . Let  $k$  be the coordinate such that  $(Lh)_k \neq 0$  and  $\epsilon > 0$ .

Since  $x_0$  is an infimum of  $f$ ,  $f(x)_{\ell} \geq f(x_0)_{\ell}$  for all  $\ell \in \mathbb{N}$  and  $x \in \mathbb{R}^{\omega}$ . Moreover,  $f$  is continuously differentiable at  $x_0$  so there exists an  $\delta > 0$  such that for any  $x \in \mathbb{R}^{\omega}$ ,  $\|x - x_0\|_k < \delta$  implies  $\frac{\|f(x) - f(x_0) - L(x - x_0)\|_{\ell}}{\|x - x_0\|_k} < \epsilon$  for all  $\ell \in \mathbb{N}$ .

<sup>11</sup>This can be easily seen by substituting  $h$  by  $\frac{x - x_0}{\|x - x_0\|_k}$  in the standard definition of continuously differentiable functions  $f$  on  $A \subseteq \mathbb{R}^{\omega}$ .

Consider the sequence  $\{y_n\}_{n \in \mathbb{N}}$  defined as  $y_n := x_0 - \frac{1}{n} \frac{Lh}{\|Lh\|_k} \cdot h$ . The distance between  $y_n$  and  $x_0$  is  $\|y_n - x_0\|_k = \left\| \frac{-1}{n} \frac{Lh}{\|Lh\|_k} \cdot h \right\|_k = \frac{1}{n} \|h\|_k$ . So for large enough  $n$ ,  $\|y_n - x_0\|_k < \delta$ .

Hence,

$$0 \leq \frac{(f(y_n) - f(x_0))_k}{\|y_n - x_0\|_k} < \frac{L(y_n - x_0)_k}{\|y_n - x_0\|_k} + \epsilon = \frac{n}{\|h\|_k} \cdot \left( \frac{-1}{n} \frac{(Lh)^2}{\|Lh\|_k} \right)_k + \epsilon = -\frac{\|Lh\|_k}{\|h\|_k} + \epsilon$$

which implies  $\|Lh\|_k < \|h\|_k \epsilon$ . Since  $\epsilon$  is arbitrary, we have  $\|Lh\|_k \leq 0$  which implies  $(Lh)_k = 0$  and contradicts our assumption.  $\square$

Now we show that the Markov chain can move to any measurable set with positive measure on  $A$  from almost all  $w$ -supported element in  $A$ .

**Lemma 33.** *Assuming  $w$  is continuously differentiable on a non-null set  $A \in \mathcal{U}$  and  $\{U_n\}$  is uniformly bounded above (i.e. there is an upper bound  $M$ , where  $U_n(\mathbf{q}) < M$  for all  $\mathbf{q} \in \text{Dom}(U_n)$  for all  $n \in \mathbb{N}$ ) and  $\nabla U_n$  is Lipschitz on  $A \cap \text{Dom}(U_n)$ . For almost all  $\mathbf{a} \in A \cap \text{Supp}(w)$  and measurable subset  $B \subseteq A$ ,  $\nu(B) > 0$  implies  $Q(\mathbf{a}, B) > 0$ .*

*Proof.* It is enough to prove the statement for a non-null measurable set  $B \subseteq A \cap \mathbb{R}^n$  where all elements of  $B$  have positive weight since all measurable subset  $B$  of  $A$  with  $\nu(B) > 0$  must contains such a subset. Moreover we restrict  $B$  to the elements where the statement in Lem. 31 always hold w.r.t.  $\mathbf{a}$ .

Say  $m = |\mathbf{a}|$  and  $M = \max\{m, n\}$ . Let  $I_{\mathbf{a}}(B) = \{\mathbf{p} \in \mathbb{R}^k \mid k \geq M \text{ and all intermediate leapfrog steps starting from } (\mathbf{a} + \mathbf{0}^{1 \dots k-m}, \mathbf{p}) \in \mathbb{S}^{\text{valid}} \text{ are in } A \cap \text{Dom}(U_k) \text{ and } \text{proj}_1(\Psi_k(\mathbf{a} + \mathbf{0}^{1 \dots k-m}, \mathbf{p}))^{1 \dots n} \in B\}$ . It is enough to show that  $\sum_{k=M}^{\infty} \text{Leb}_k(I_{\mathbf{a}}(B) \cap \mathbb{R}^k) > 0$ .

Let  $\theta : I_{\mathbf{a}}(B) \rightarrow B$  be the function where  $\theta(\mathbf{p})$  gives the next sample in  $B$  after  $L$  HMC leapfrog steps starting with initial state  $(\mathbf{a} + \mathbf{0}^{1 \dots |\mathbf{p}|-m}, \mathbf{p})$ . By Lem. 31,  $\theta$  is surjective.

We write  $I_{\mathbf{a}}^k(B) = I_{\mathbf{a}}(B) \cap \mathbb{R}^k$  and show that  $\theta_k : I_{\mathbf{a}}^k(B) \rightarrow B$  is Lipschitz. By assumption for any  $\mathbf{p}^0 \in I_{\mathbf{a}}^k(B)$ , all the intermediate positions are in  $\text{Dom}(U_k) \cap A$ . Hence, we can write  $\theta_k(\mathbf{p}) := \text{proj}_1(\Psi_k(\mathbf{a} + \mathbf{0}^{1 \dots k-m}, \mathbf{p})) = \mathbf{q}^L$  as

$$\mathbf{q}^0 + \epsilon L \mathbf{p}^0 - \epsilon^2 \left( \frac{L}{2} \nabla U_k(\mathbf{q}^0) + \sum_{k=1}^{L-1} k \nabla U_k(\mathbf{q}^{L-k}) \right).$$

Let  $\mathbf{p}, \mathbf{p}' \in I_{\mathbf{a}}^k(B)$ , and  $\mathbf{q}^i, \mathbf{q}'^i$  be the position of the state after  $i$  leapfrog steps with momentum kick  $\mathbf{p}, \mathbf{p}'$  respectively. Then,

$$\begin{aligned} |\theta_k(\mathbf{p}) - \theta_k(\mathbf{p}')| &= |\mathbf{q}^L - \mathbf{q}'^L| \leq \epsilon L |\mathbf{p} - \mathbf{p}'| + \epsilon^2 \sum_{i=1}^{L-1} i |\nabla U_k(\mathbf{q}^{L-i}) - \nabla U_k(\mathbf{q}'^{L-i})| \\ &\leq \epsilon L |\mathbf{p} - \mathbf{p}'| + \epsilon^2 \sum_{i=1}^{L-1} i |\mathbf{q}^{L-i} - \mathbf{q}'^{L-i}| \quad (U_k \text{ is Lipschitz on } A \cap \text{Dom}(U_k)) \end{aligned}$$

hence  $|\theta_k(\mathbf{p}) - \theta_k(\mathbf{p}')| \leq c |\mathbf{p} - \mathbf{p}'|$  for some constant  $c$  and  $\theta_k$  is Lipschitz.

Assume for contradiction that  $\sum_{k=M}^{\infty} \text{Leb}_k(I_{\mathbf{a}}(B) \cap \mathbb{R}^k) = 0$  which means that for all  $k \geq M$ ,  $\text{Leb}_k(I_{\mathbf{a}}^k(B)) = 0$ . However,

$$\begin{aligned} \text{Leb}_n(B) &= \text{Leb}_n(\theta(I_{\mathbf{a}}(B))) = \text{Leb}_n\left(\theta\left(\bigcup_{k=M}^{\infty} I_{\mathbf{a}}^k(B)\right)\right) = \text{Leb}_n\left(\bigcup_{k=M}^{\infty} \theta_k(I_{\mathbf{a}}^k(B))\right) \\ &\leq \sum_{k=M}^{\infty} \text{Leb}_n(\theta_k(I_{\mathbf{a}}^k(B))) \leq \sum_{k=M}^{\infty} \text{Lip}(\theta_k)^{3N} \cdot \text{Leb}_n(I_{\mathbf{a}}^k(B)) = 0 \end{aligned}$$

implies that  $\text{Leb}_n(B) = 0$  which gives a contradiction.  $\square$

**Lemma 34.** *Assuming  $w$  is continuously differentiable on a non-null set  $A$  where  $A \in \mathcal{U}$  and  $\{\nabla U_n\}$  is uniformly bounded above and below (i.e. there are bounds  $M_1, M_2$ , where  $M_1 \leq \nabla U_n(\mathbf{q}) \leq M_2$  for all  $\mathbf{q} \in \text{Dom}(\nabla U_n)$  for all  $n \in \mathbb{N}$ ). Then there exists a step size  $\epsilon$  such that for any sequence  $\mathbf{q} \in \text{Supp}(w)$ ,  $\nu(A) > 0$  implies  $Q(\mathbf{q}, A) > 0$ .*

*Proof.* Let  $\mathbf{q} \in \mathbb{R}^m$  be  $w$ -supported. Since  $A \in \mathcal{U}$ , its interior  $\overset{\circ}{A}$  is a non-empty open set. Hence for some  $n$ , there is a non-empty open subset  $\prod_{i=1}^n (a_i, b_i)$  of  $A \cap \mathbb{R}^n$ .

Now we consider the conditions on the starting momentum  $\mathbf{p}^0$  in order for the position  $\mathbf{q}^L$  at the end of the trajectory of the leapfrog steps to be in  $A$  assuming that the position of the intermediate states never leave the domain of  $U_k$  for some  $k \geq M := \max\{m, n\}$ .

$$\begin{aligned} \mathbf{q}^L \in \prod_{i=1}^n (a_i, b_i) \times \mathbb{R}^{k-n} &\Leftrightarrow \forall i = 1, \dots, n \quad \mathbf{q}_i^0 + \epsilon L \mathbf{p}_i^0 - \epsilon^2 \left( \frac{L}{2} \nabla U_k(\mathbf{q}^0) + \sum_{k=1}^{L-1} k \nabla U_k(\mathbf{q}^{L-k}) \right) \in (a_i, b_i) \\ &\Leftrightarrow \forall i = 1, \dots, n \quad \mathbf{p}_i^0 \in \left( \frac{1}{\epsilon L} (a_i - \mathbf{q}_i^0 + \frac{(\epsilon L)^2}{2} M_2), \frac{1}{\epsilon L} (b_i - \mathbf{q}_i^0 + \frac{(\epsilon L)^2}{2} M_1) \right) =: I_i \end{aligned}$$

For any  $\mathbf{p} \in \prod_{i=1}^n I_i$ , the union  $\bigcup_{k=M}^{\infty} \{\mathbf{p}' \in \mathbb{R}^{k-n} \mid (\mathbf{q} + \mathbf{0}^{k-m}, \mathbf{p} + \mathbf{p}') \in \mathbb{S}^{\text{valid}}\}$  is non-null. This is because the measure of the union can be seen as the value measure of the almost surely terminating probabilistic program which given  $\mathbf{q} \in \text{Supp}^m(w)$  and  $\mathbf{p} \in \prod_{i=1}^n I_i$  returns  $\mathbf{p}' \in \mathbb{R}^{k-n}$  such that  $(\mathbf{q} + \mathbf{0}^{k-m}, \mathbf{p} + \mathbf{p}')$  is a valid state,

For  $\epsilon < \frac{1}{L} \sqrt{\frac{2(b_i - a_i)}{M_2 - M_1}}$  for all  $i$ , the intervals  $\{I_i\}$  are non-empty and hence  $Q(\mathbf{q}, A) \geq \sum_{k=M}^{\infty} \mathcal{N}_k(\{\mathbf{p}' \in \mathbb{R}^{k-n} \mid (\mathbf{q} + \mathbf{0}^{k-m}, \mathbf{p} + \mathbf{p}') \in \mathbb{S}^{\text{valid}}, \mathbf{p} \in \prod_{i=1}^n I_i\}) > 0$ .  $\square$

**Definition 35.** We gather all the conditions so far.

- (C1)  $w$  is continuously differentiable on a non-null set  $A$  with measure-zero boundary.
- (C2)  $w|_{\text{Supp}(w)}$  is bounded below by a positive constant.
- (C3) For each  $n$ , the function  $\frac{\nabla w_{\leq n}}{w_{\leq n}}$  is uniformly bounded from above and below on  $\text{Supp}(w_{\leq n}) \cap A$ .
- (C4) For each  $n$ , the function  $\frac{\nabla w_{\leq n}}{w_{\leq n}}$  is Lipschitz continuous on  $\text{Supp}(w_{\leq n}) \cap A$ .

Note that

- (C1) implies  $w$  is continuously differentiable on a non-null set  $A \in \mathcal{U}$ .
- (C2) implies  $\{U_n\}$  is uniformly bounded above (i.e. there is an upper bound  $M$ , where  $U_n(\mathbf{q}) < M$  for all  $\mathbf{q} \in \text{Dom}(U_n)$  for all  $n \in \mathbb{N}$ ).
- (C3) implies  $\{\nabla U_n\}$  is uniformly bounded above and below (i.e. there are bounds  $M_1, M_2$ , where  $M_1 \leq \nabla U_n(\mathbf{q}) \leq M_2$  for all  $\mathbf{q} \in \text{Dom}(\nabla U_n)$  for all  $n \in \mathbb{N}$ ).
- (C4) implies  $\nabla U_n$  is Lipschitz on  $A \cap \text{Dom}(U_n)$ .

Now we are ready to prove irreducibility.

**Lemma 36 (Irreducible).** *If Assumptions (C1)–(C4) are satisfied, there exists a step size  $\epsilon$  such that for any sequence  $\mathbf{q} \in \text{Supp}(w)$  and measurable set  $B \in \Sigma_{\mathbb{T}}$ ,  $\nu(B) > 0$  implies  $Q^i(\mathbf{q}, B) > 0$  for  $i \in \{1, 2\}$ .*

*Proof.* Let  $A$  be the non-null set in  $\mathcal{U}$  where  $w$  is continuously differentiable on  $A$  and  $\mu_{\mathbb{T}}(\mathbb{T} \setminus A) = 0$  and Lem. 33 holds for all elements in  $A$ . Such  $A$  must exist by Assumption 2 and (C1).

First note that  $\nu(A \cap B) > 0$ . Otherwise, we must have  $\nu((\mathbb{T} \setminus A) \cap B) > 0$ . But this implies  $\mu_{\mathbb{T}}(\mathbb{T} \setminus A) \geq \mu_{\mathbb{T}}((\mathbb{T} \setminus A) \cap B) > 0$  which contradicts the assumption.

We do case analysis on  $\mathbf{q} \in \mathbb{T}$ .

- If  $\mathbf{q} \in A$ , then by Lem. 33,  $Q(\mathbf{q}, A \cap B) > 0$ .
- If  $\mathbf{q} \notin A$ , then by Lem. 34,  $Q(\mathbf{q}, A) > 0$  and so

$$\begin{aligned} Q^2(\mathbf{q}, B) &\geq Q^2(\mathbf{q}, A \cap B) \\ &= \int_{\mathbb{T}} Q(\mathbf{q}', A \cap B) Q(\mathbf{q}, d\mathbf{q}') \\ &\geq \int_A Q(\mathbf{q}', A \cap B) Q(\mathbf{q}, d\mathbf{q}') > 0. \end{aligned}$$

$\square$

**Lemma 37** (Aperiodic). *If Assumptions (C1)–(C4) are satisfied,  $Q$  is aperiodic.*

*Proof.* Assume for contradiction that  $Q$  is not aperiodic. Then, there exists disjoint  $B_0, \dots, B_d$  for  $d \geq 1$  such that  $\nu(B_0) > 0$  and  $x \in B_i$  implies  $Q(x, B_{(i+1) \bmod (d+1)}) = 1$  for all  $i = 0, \dots, d$ .

Let  $A$  be the non-null set in  $\mathcal{U}$  where  $w$  is continuously differentiable on  $A$  and  $\mu_{\mathbb{T}}(\mathbb{T} \setminus A) = 0$  and Lem. 34 holds for all elements in  $A$ . Such  $A$  must exist by Assumption 2 and (C1). Let  $C_i := B_i \cap A$  for all  $i = 0, \dots, d$ . Hence,  $\nu(C_0) > 0$  and  $x \in C_i$  implies  $Q(x, C_{(i+1) \bmod (d+1)}) = 1$  for all  $i = 0, \dots, d$ .

Let  $x \in C_0$  be a  $w$ -supported sequence. Such an  $x$  must exist as  $\nu(C_0) > 0$ . Then,  $Q(x, C_1) = 1$  implies  $Q(x, C_0) \leq Q(x, \mathbb{T} \setminus C_1) = 0$  which contradicts with Lem. 36 as  $x \in A$ .

□

Finally by Tierney’s Theorem (Lem. 19), the  $\nu$ -irreducible (Lem. 36) and  $\nu$ -aperiodic (Lem. 37) transition kernel  $Q$  with invariant distribution  $\nu$  (Thm. 4) converges to  $\nu$ .

**Theorem 5.** *If Assumptions (C1)–(C4) are satisfied in addition to Assumptions 1, 2 and 3, the Markov chain generated by iterating Alg. 1 converges to the target distribution  $\nu$ .*

## D. Experiments

### D.1. Details on the Experimental Setup

For our experimental evaluation, we implemented the algorithms in Python, using PyTorch for tensor and gradient computations. The source code for our implementation and experiments is available at <https://github.com/fzaiser/nonparametric-hmc> and archived as (Zaiser & Mak, 2021).

**Inference algorithms** The four inference algorithms we compared were:

1. NP-DHMC (ours): the nonparametric adaptation of (Nishimura et al., 2020), explained in App. B.2, using the efficiency improvements from App. B.3.
2. Lightweight Metropolis-Hastings (LMH),
3. Particle Gibbs (PGibbs) and
4. Random walk lightweight Metropolis-Hastings (RMH).

We used the Anglican implementations of the latter three algorithms.

**Models** For NP-DHMC, the models were given to the algorithm as probabilistic programs in the form of a Python function with a context argument for NP-DHMC. The context allows probabilistic primitives and records the trace and weight for the inference algorithms. This way, evaluating the density function  $w$  amounts to running the probabilistic programs. For LMH, PGibbs, and RMH, the Python models were translated to Clojure programs using Anglican’s probabilistic programming constructs. The pseudocode for the geometric example and the random walk example can be found in the main text. The Gaussian and Dirichlet process mixture model is explained there as well, using statistical notation. Sampling from  $DP(\alpha, \text{Uniform}([0, 1]^3))$  is implemented using the stick-breaking procedure (Sethuraman, 1994). We use a cutoff of  $\epsilon = 0.01$  for the stick size as explained in the text. In pseudocode, it looks as follows:

```
def dp(alpha, H):
    stick = 1.0
    beta = 0.0
    cumulative_product = 1.0
    weights = []
    means = []
    while stick > 0.01:
        cumulative_product *= 1 - beta
        beta = sample(Beta(1, alpha))
        theta = sample(H)
        weights.append(beta * cumulative_product)
```

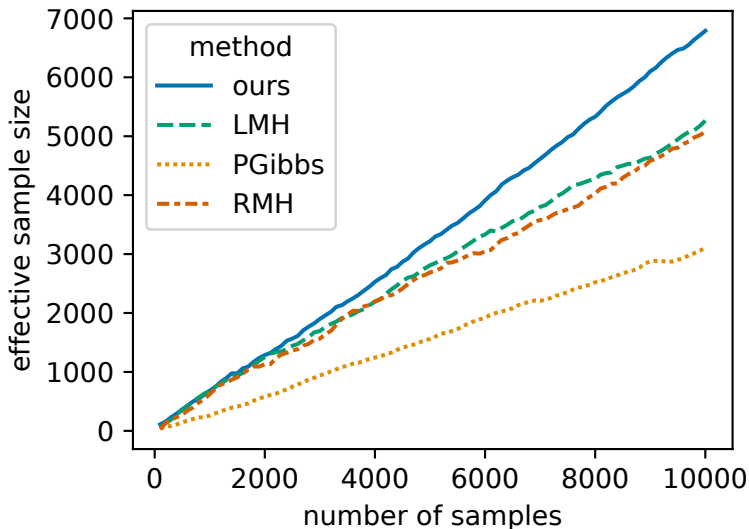


Figure 12. ESS for the random walk example in terms of number of samples

```

means.append(theta)
stick -= beta * cumulative_product
return weights, means
    
```

**ESS computation** For the random walk example, we computed the effective sample size. For this we used NumPyro’s (Bingham et al., 2019) `diagnostics.effective_sample_size` function. It is designed to estimate the effective sample size for MCMC samplers using autocorrelation (Gelman et al., 2014). For importance samples used as the ground truth, we used the importance weights directly to compute the ESS: given importance weights  $w_1, \dots, w_n$ , the ESS is  $\frac{(\sum_{i=1}^n w_i)^2}{\sum_{i=1}^n w_i^2}$ . We also computed the (autocorrelation-based) MCMC ESS for the importance samples and we obtained very similar results.

**Hyperparameter choices** We produced 10 runs with 1000 samples each for every example except the last, Dirichlet process mixture model (DPMM). For the DPMM example, we only produced 100 samples in each run because of the forbidding computational cost. We set the number of burn-in samples that are discarded to 10% of the total number of samples, i.e. 100 samples for each run. Since each run of the DPMM only had 100 samples, we set the burn-in higher there, namely to 50. We did not vary this hyperparameter much because higher values did not seem to make a difference. For the number of leapfrog steps we tried values  $L \in \{5, 20, 50, 100\}$ , and for the step size we tried values  $\epsilon \in \{0.01, 0.05, 0.1, 0.5\}$ . Generally, the simple geometric distribution example already works for very rough hyperparameters ( $L = 5, \epsilon = 0.1$ ). Finer steps work as well, but are not necessary. However, more complex models generally require finer steps (GMM:  $L = 50, \epsilon = 0.05$ ). The other inference algorithms we tested don’t have any hyperparameters that need to be set.

**Thinning** Since NP-DHMC performs more computation than its competitors for each sample because it evaluates the density function in each of the  $L$  leapfrog steps, not just once like the other inference algorithms. To equalise the computation budgets, we generate  $L$  times as many samples for each competitor algorithm, and apply thinning (taking every  $L$ -th sample) to get a comparable sample size.

## D.2. Additional Plots and Data

In addition to the ESS and LPPD computations, we also plotted both as a variable of the number of samples computed. The results can be seen in Fig. 12 and 13. As we can see, NP-DHMC performs the best consistently over the course of the inference, not just in terms of the final result.

**Running time** We report the wall-clock times for the different algorithms. Experiments were carried out on a computer with an Intel Core i7-8700 CPU @ 3.20 GHz x 12 and 16 GB RAM, running Ubuntu 20.04. The results are presented in Table 2.

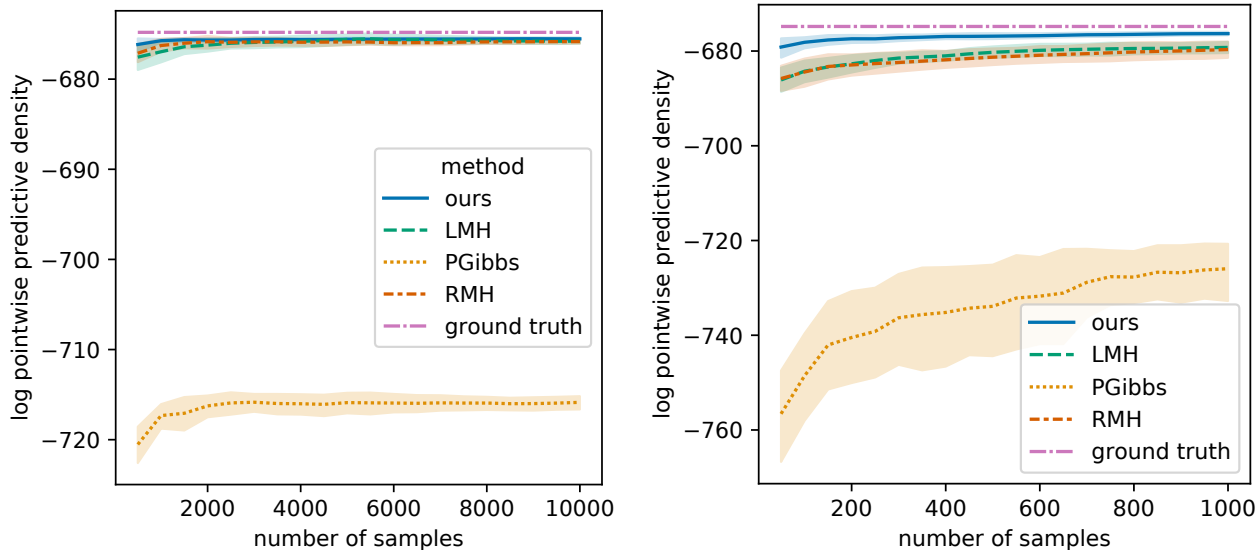


Figure 13. LPPD for the GMM and DP mixture model in terms of the number of samples from 10 runs. The shaded area is one standard deviation. These are the full plots of Fig. 8 and 9, respectively.

Table 2. Running times for the different inference algorithms in seconds per sample.

method	ours	LMH	PGibbs	RMH	Pyro HMC	Pyro NUTS
geometric example	0.0418	0.0003	0.0001	0.0005	n/a	n/a
random walk example	0.2266	0.0077	0.0051	0.0095	$\approx 0.41$	$\approx 5.7$
GMM example	0.1879	1.6572	1.6835	1.6376	n/a	n/a
DPMM example	1.8516	2.1491	1.7855	2.0584	n/a	n/a

NP-DHMC is significantly slower than the competition in the geometric and random walk examples, faster for GMM and comparable for DPMM. Due to the nature of the coordinate integrator of discontinuous HMC (Nishimura et al., 2020), NP-DHMC has to run the model  $L \times d$  times per sample where  $d$  is the number of discontinuous variables in the model. We could improve the algorithm by only updating a subset of the discontinuous variables per iteration. In addition, NP-DHMC computes gradients and simulates Hamiltonian dynamics, which is computationally expensive. On the random walk example we also ran Pyro HMC and NUTS, as mentioned before. Both of them were a lot slower than our implementation, which speaks to the fact that HMC methods simply have an unavoidable performance overhead. Finally, the implementation of NP-DHMC is a research prototype, so it is not optimal and there is a lot of room for improvement.