Deep Involutive Generative Models for Neural MCMC

Span Spanbauer
MIT

Cameron Freer
MIT

Vikash Mansinghka
MIT

We introduce deep involutive generative models, a new architecture for deep generative modeling, and use them to define *Involutive Neural MCMC*, a new approach to fast neural MCMC. An involutive generative model represents a probability kernel $G(\phi \mapsto \phi')$ as an involutive (i.e., self-inverting) deterministic function $f(\phi, \pi)$ on an enlarged state space containing auxiliary variables π. We show how to make these models volume-preserving, and how to use deep volume-preserving involutive generative models to make valid Metropolis–Hastings updates based on an auxiliary variable scheme with an easy-to-calculate acceptance ratio. We prove that deep involutive generative models and their volume-preserving special case are universal approximators for probability kernels. This result implies that with enough network capacity and training time, they can be used to learn arbitrarily complex MCMC updates. We define a loss function and optimization algorithm for training parameters given simulated data. We also provide initial experiments showing that Involutive Neural MCMC can efficiently explore multi-modal distributions that are intractable for Hybrid Monte Carlo, and can converge faster than A-NICE-MC, a recently introduced neural MCMC technique.

1 Introduction

Markov Chain Monte Carlo (MCMC) methods are a class of very general techniques for statistical inference [4]. MCMC has seen widespread use in many domains, including cosmology [10], localization [18], phylogenetics [20], and computer vision [14]. For the Metropolis–Hastings class of MCMC algorithms, one must specify a proposal distribution $q(\phi \mapsto \phi')$. Convergence will be slow if the proposal distribution is poorly tuned to the posterior distribution $p(\phi'|D)$. Conversely, if one could use a perfectly-tuned proposal distribution — ideally, the exact posterior $q(\phi \mapsto \phi') = p(\phi'|D)$ — then MCMC would converge in a single step.

Neural MCMC refers to an emerging class of deep learning approaches [21, 22, 16] that attempt to learn good MCMC proposals from data. Neural MCMC approaches can be guaranteed to converge, as the number of MCMC iterations increases, to the correct distribution — unlike neural variational inference [17, 13, 19], which can suffer from biased approximations. Recently, Song et al. [21] suggested that involutive neural proposals are desirable but difficult to achieve:

> "If our proposal is deterministic, then $f_\theta(f_\theta(x, v)) = (x, v)$ should hold for all (x, v), a condition which is difficult to achieve." [21]

Contributions. This paper presents a solution to the problem of learning involutive proposals posed by [21]. Specifically, it presents the following contributions:

1. This paper introduces involutive neural networks, a new class of neural networks that is guaranteed to be involutive by construction; we also show how to constrain the Jacobian of these networks to have magnitude 1, that is, to preserve volume.
2. This paper uses involutive networks to define involutive generative models, a new class of auxiliary variable models, and shows that the volume-preserving ones can be used as Metropolis–Hastings proposals.
3. This paper proves that volume-preserving involutive generative models are universal approximators for transition kernels, justifying their use for black-box learning of good MCMC proposals.
4. This paper describes a new, lower-variance estimator for the Markov-GAN training objective [21] that we use to train involutive generative models.
5. This paper shows that *Involutive Neural MCMC* can improve on the convergence rate of A-NICE-MC, a state-of-the-art neural MCMC technique.
6. This paper illustrates *Involutive Neural MCMC* on a simple problem.

We motivate our approach by showing that several common Metropolis–Hastings proposals are special cases of involutive proposals (Section 2). We then show that by using a class of exactly involutive neural network architectures (Section 3) satisfying an appropriate universality condition (Section 4) and using adversarial training (Section 5), we can find involutive proposals that empirically converge extremely rapidly (Section 6).

July 2, 2020

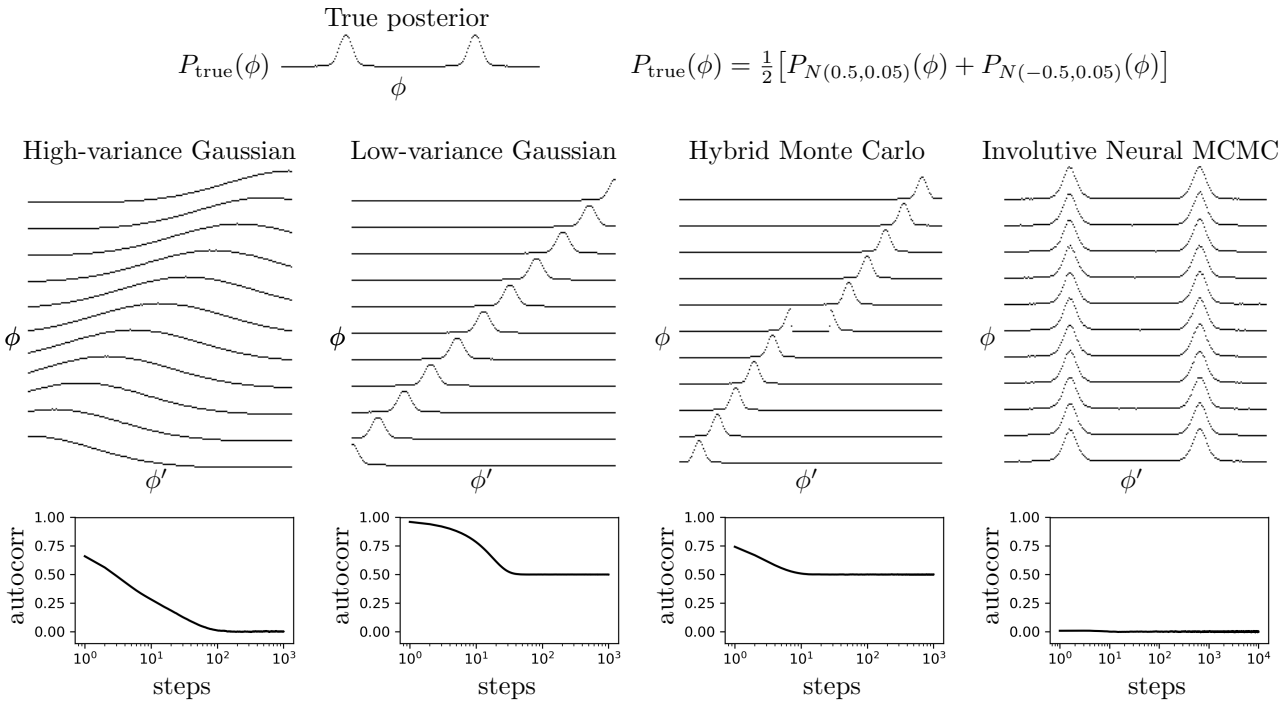

Figure 1: Consider the problem of using MCMC to sample from a mixture of two Gaussians. Here, each trace shows the distribution (rescaled for clarity) of proposed state transitions ϕ' from a given initial state ϕ. High-variance Gaussian proposals are a poor approximation of the posterior, and hence converge slowly. Low-variance Gaussian proposals fail to mix between the two modes because proposals between the modes will be rejected with high probability. Hybrid Monte Carlo converges quickly within a mode, but also fails to mix between modes. Proposals from Involutive Neural MCMC nearly match the posterior from every state, mixing and nearly converging in a single step.

2 Background

Recall that the speed of convergence of a Metropolis–Hastings algorithm is highly dependent on how well the proposal distributions match the posterior.

In order to use a given proposal distribution, one typically constructs a transition which satisfies the *detailed balance* condition, which (in an ergodic setting) ensures convergence to the posterior. Satisfying this condition for a general proposal is hard, which has led researchers to use smaller classes of proposals for which this problem is tractable. Our method, *Involutive Neural MCMC*, satisfies detailed balance for a universal class of proposal distributions, drawn from a generative model specified by a volume-preserving involutive function. Our method builds on previous work on invertible neural networks [1], for example the architecture we use in our constructive proof of universality makes use of additive coupling layers [7] which have been cascaded [8] [12]. We now describe several existing classes of proposals, and observe that each can be viewed as an involutive proposal.

The canonical example of a class of proposal distributions is the collection of shifts by a multivariate Gaussian. These immediately satisfy detailed balance due to their symmetry, that is, the probability $P(\phi \mapsto \phi')$ of a forward transition is equal to the probability $P(\phi' \mapsto \phi)$ of a backward transition. However, multivariate Gaussians are usually poor approximations of the posterior, leading to slow convergence. We observe that these proposals can be viewed as involutive proposals: choose the auxiliary variable π to be a sample from the multivariate Gaussian, and define the state transition to be $(\phi, \pi) \mapsto (\phi + \pi, -\pi)$.

Another example class of proposal distributions is those generated by Hamiltonian dynamics in the Hybrid Monte-Carlo algorithm [9]. These proposals can be shown to satisfy detailed balance because they are involutive: proposals for Hybrid Monte Carlo are obtained by simulating a particle for a certain amount of time and then negating its momentum; if one performs this operation twice, the particle will end in its initial state.

Recently, researchers have begun using neural networks to parameterize classes of proposal distributions, leading to *neural MCMC* algorithms. The A-NICE-MC

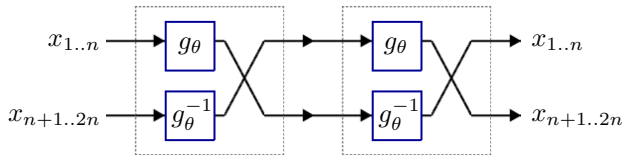

Figure 2: System diagram showing that two composed involutive function blocks forms the identity operation.

method involves choosing a symmetric class of proposals parameterized by an invertible neural network: its Metropolis–Hastings proposal assigns $1/2$ probability to the output of the network, and $1/2$ to the output of the inverse of the network. This proposal is symmetric, and hence satisfies detailed balance. However, one can also view it as involutive. Specifically, let f be an invertible neural network, and $\pi \sim N(0,1)$ the auxiliary variable. Define the state transition to be

$$(\phi, \pi) \mapsto \begin{cases} (f(\phi), -\pi) & \text{if } \pi > 0, \\ (f^{-1}(\phi), -\pi) & \text{otherwise.} \end{cases}$$

We have seen that all of these examples are special cases of involutive proposals. We now introduce a class of exactly involutive neural network architectures (Section 3) and show that they satisfy a universality condition, and so may be used to approximate any involutive proposal arbitrarily well (Section 4).

3 Involutive Neural Networks

In this section, we describe how to build deep neural networks which are exactly involutive by construction. To do this, we first describe three kinds of smaller involutive building blocks, and then we describe how to compose these blocks to form a deep involutive network:

- **Involutive function blocks**, which are fairly general nonlinear maps, but do not fully mix information, in that every element of the output is independent of half of the elements of the input.
- **Involutive permutation blocks**, which are linear maps and cannot be optimized, but can mix information.
- **Involutive matrix blocks**, which are linear maps, but can be optimized and can mix information.

By composing these blocks in a particular way, we can create deep networks which have high capacity and are exactly involutive. We further show that each involutive block is either volume-preserving or can be made so, leading to high-capacity volume-preserving involutive networks appropriate for use as MCMC transition kernels. In Section 4, we prove that these deep involutive networks are universal in a particular sense.

Let $a^\frown b$ denote the concatenation of vectors a and b, and write $a_{j..k}$ to denote the restriction of the vector a to its terms indexed by $j, j+1, \ldots, k$. Let Id_n denote the $n \times n$ identity matrix.

3.1 Involutive function blocks

Involutive function blocks enable the application of fairly general nonlinear functions to the input data. In the typical case, they are parameterized by an invertible neural network [1], which is itself parameterized by two neural networks of arbitrary architecture.

Definition 3.1 (Involutive function block)**.** Let $n \in \mathbb{N}$ and let $g\colon \mathbb{R}^n \to \mathbb{R}^n$ be a bijection.

Define the **involutive function block** $I_F^{2n,g}\colon \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ by $I_F^{2n,g}(x) \coloneqq g^{-1}(x_{n+1..2n})^\frown g(x_{1..n})$.

Observe that $(I_F^{2n,g} \circ I_F^{2n,g})(x) = (g^{-1} \circ g)(x_{1..n})^\frown (g \circ g^{-1})(x_{n+1..2n}) = x$, where $\circ$ denotes function composition, and so the function $I_F^{2n,g}$ is indeed an involution. See Fig. 2 for a system diagram depicting this fact. Further note that n and g are uniquely determined by the function $I_F^{2n,g}$, and so when this function is induced by some neural net (i.e., when g is itself induced by a neural net), we may think of $I_F^{2n,g}$ as a neural net with the same parameters as the neural net inducing g. In this case we will sometimes elide the distinction between the function and this neural net, or refer to the function induced by the neural net by the same symbol.

If the parameter g is volume-preserving, that is, the determinant of its Jacobian has magnitude 1, then we observe that the resulting involutive function block is also volume-preserving by considering the properties of the determinant of block matrices. In our experiments we obtain volume-preserving involutive function blocks by parameterizing g using NICE additive coupling layers [7] which have been cascaded [8] [12], since these NICE layers preserve volume.

3.2 Involutive permutation blocks

One can see from Fig. 2, as previously noted, that each output of an involutive function block is independent of half of the inputs. In order to create more general involutive networks without this property, we mix information by applying an involutive permutation.

Definition 3.2 (Involutive permutation block)**.** Let $n \in \mathbb{N}$ and let σ be an involution on the set $\{1, \ldots, n\}$. Let $\boldsymbol{\sigma}$ denote the matrix defined by $\boldsymbol{\sigma}_{ij} e_j = e_{\sigma(i)}$ for $i, j \in \{1, \ldots, n\}$, where the e_i are basis vectors.

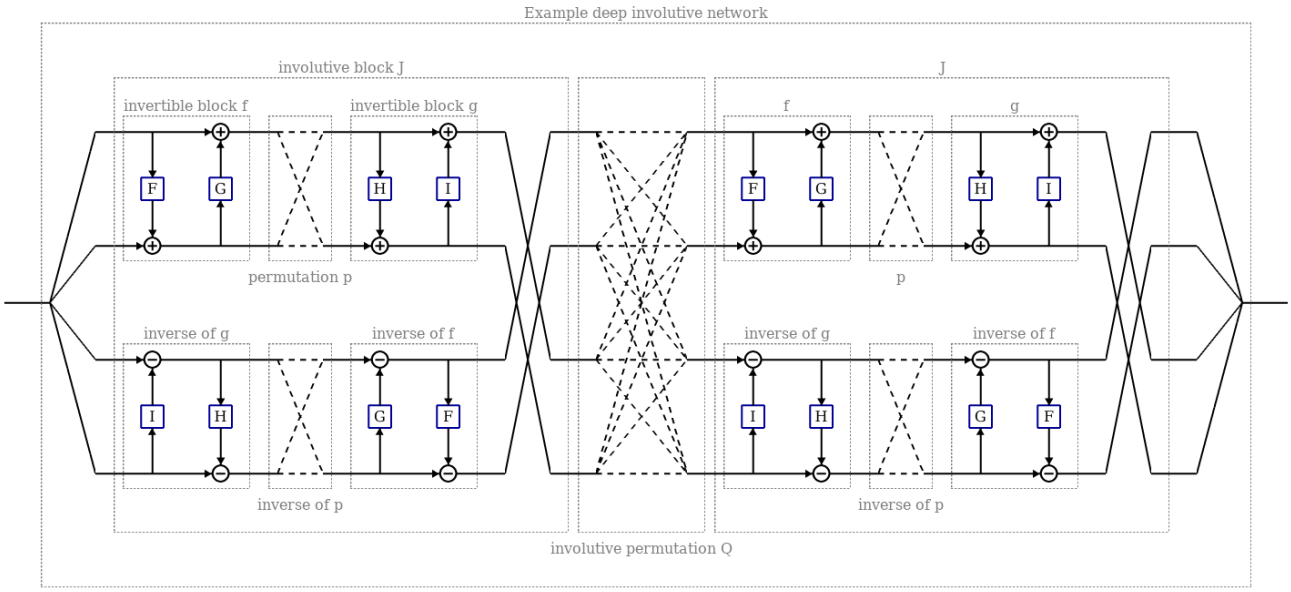

Figure 3: System diagram of a typical deep involutive network. The functions F, G, H, and I are arbitrary functions, usually induced by neural networks.

Define the **involutive permutation block** $I_P^{n,\sigma}\colon \mathbb{R}^n \to \mathbb{R}^n$ by $I_P^{n,\sigma}(x) \coloneqq \boldsymbol{\sigma} x$.

Note that n and σ are uniquely determined by the function $I_P^{n,\sigma}$. We may also think of this function as a linear layer with no parameters in a neural net.

Observe that involutive permutation blocks, as permutations, are volume preserving.

One may use any involutive permutation σ: we use a specific choice of σ in the proof of universality, and we use uniformly random involutive permutations in the experiments. An algorithm for sampling uniformly from the space of n-dimensional involutive permutations is described in [3].

3.3 Involutive matrix blocks

As an alternative to involutive permutations, we may use a different class of involutive matrices to mix information. Compared to involutive permutations, involutive matrix blocks have the advantage that they can be optimized. This is because they are parameterized by two arbitrary nonzero vectors of real numbers.

Definition 3.3 (Involutive matrix block)**.** Let $n \in \mathbb{N}$ and let $v, w \in \mathbb{R}^n \setminus \{0^n\}$.

Define the **involutive matrix block** $I_M^{n,v,w}\colon \mathbb{R}^n \to \mathbb{R}^n$ by $I_M^{n,v,w} \coloneqq \mathrm{Id}_n - \frac{2v\otimes w}{v\cdot w}$.

Involutive matrix blocks are in fact involutive; for completeness, we include the following proof, adapted from [15].

Lemma 3.4. *Every involutive matrix block is involutive.*

Proof. Let $n \in \mathbb{N}$ and $v, w \in \mathbb{R}^n \setminus \{0^n\}$. The product of $I_M^{n,v,w}$ with itself is the identity:

$$\left(\mathrm{Id}_n - 2\frac{v\otimes w}{v\cdot w}\right)\left(\mathrm{Id}_n - 2\frac{v\otimes w}{v\cdot w}\right) = \mathrm{Id}_n - 4\frac{v\otimes w}{v\cdot w} + 4\frac{(v\otimes w)(v\otimes w)}{(v\cdot w)^2} = \mathrm{Id}_n.$$

Hence $I_M^{n,v,w}$ is involutive. □

Not all involutive matrices are involutive matrix blocks; for example, the identity matrix is involutive but not an involutive matrix block.

Compared to involutive permutation blocks, involutive matrix blocks potentially allow for freer mixing between dimensions, and can also be optimized.

Note that n, v, and w are uniquely determined by the function $I_M^{n,v,w}$ if we constrain $|w| = 1$, and so without loss of generality we may think of it as a neural net with parameters v and w.

Observe that involutive matrix blocks are volume preserving by the matrix determinant lemma.

3.4 Deep involutive networks

In order to create deep networks which have high capacity and which are exactly involutive, we want to compose several involutive blocks. For the resulting

network to be involutive, we must compose them in particular ways.

Definition 3.5 (Involutive network)**.** Let $n \in \mathbb{N}$. We say that a neural network is an **invertible network of dimension** n if it induces a bijection from $\mathbb{R}^n$ to $\mathbb{R}^n$ and its inverse is also expressible as a neural network. Write $\mathbb{V}_n$ to denote the set of invertible neural networks of dimension n.

A neural network is an **involutive network of dimension** n if the function it induces is contained in the closure of the following operations. Write $\mathbb{I}_n$ for the set of all such functions.

- $I_F^{n,g} \in \mathbb{I}_n$ for $g \in \mathbb{V}_{n/2}$ when n is even;
- $I_P^{n,\sigma} \in \mathbb{I}_n$ for every involution σ on $\{1, \dots, n\}$;
- $I_M^{n,v,w} \in \mathbb{I}_n$ for every $v, w \in \mathbb{R}^n \setminus \{0^n\}$;
- $\mathcal{I} \circ \mathcal{J} \circ \mathcal{I} \in \mathbb{I}_n$ for every $\mathcal{I}, \mathcal{J} \in \mathbb{I}_n$;
- $g^{-1} \circ \mathcal{J} \circ g \in \mathbb{I}_n$ for every $\mathcal{J} \in \mathbb{I}_n$ and $g \in \mathbb{V}_n$.

It is immediate by induction that every involutive network is involutive. Furthermore, if every block in the network is volume preserving, then the network is volume-preserving as well.

Our proof of universality considers the special case of the involutive network $g^{-1} \circ h^{-1} \circ I_P^{n,\sigma} \circ h \circ g$ for particular n, g, h, and σ. However, the field of deep learning has found that despite the fact that traditional neural networks with a single hidden layer are universal [11], most functions of interest are learned more effectively by networks with more than one hidden layer. Therefore we recommend constructing deep involutive networks similarly, as a composition of many functions.

For a system diagram of the architecture for a typical involutive neural network, see Fig. 3.

4 Universality of Involutive Generative Models

When using a machine learning model, it is useful to know which class of functions the model can represent. In this section, we consider generative models built from deep involutive networks, and show that they are universal approximators in a certain sense. Specifically, we prove that these networks, which map a state and an auxiliary variable $(\phi, \pi) \in (\mathbb{R}^n, \mathbb{R}^m)$ to an output interpreted as another state and auxiliary variable (ϕ', π'), can serve as arbitrarily good generative models of any continuous function of a Gaussian on any compact subset of $\mathbb{R}^n$.

The theorem statement and proof are given in Section 4.1.

Our proof is constructive: for any desired transition T, we explicitly construct an involutive generative model such that transitions drawn from it are likely to be drawn, with arbitrarily high probability, from a distribution as close as desired to that of T. Moreover, both of these approximation parameters are explicit in the description of the generative model. We first describe a family of involutive functions that approximate the desired cumulative distribution function, and then make use of the universal approximation theorem [11], which has a constructive proof, to approximate these involutive functions by involutive neural nets.

As a consequence, involutive generative models simply match the expressive power of traditional neural generative models. There is, however, one key advantage: a standard generative model maps a state and auxiliary variable to a state: $(\phi, \pi) \mapsto \phi'$, whereas an involutive generative model produces an additional piece of information, an output auxiliary variable π' such that the model maps $(\phi', \pi') \mapsto (\phi, \pi)$.

In other words, it produces a value for the auxiliary variable such that the model makes a backwards transition $\phi' \mapsto \phi$. This immediately gives a lower bound on the backward transition probability (via the sampling distribution for π), and is a key property we use to easily generate Metropolis–Hastings transitions satisfying detailed balance as shown in Section 4.2.

4.1 Proof of Universality

Theorem 4.1 (Involutive generative models are universal)**.** *Let $n \in \mathbb{N}$, and let random variables $\pi \sim N(0^{n+6}, \mathrm{Id}_{n+6})$ and $\pi' \sim N(0, 1)$.*

For all compact sets $\Omega \subseteq \mathbb{R}^n$, and all continuous functions $T \colon \mathbb{R}^n \times \mathbb{R} \to \Omega$ there exists a sequence $\{\widehat{\mathcal{I}}_m\}_{m \in \mathbb{N}}$ of involutive neural networks that induce continuous functions $\mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ such that for all $\phi \in \Omega$ the random variables $\widehat{\mathcal{I}}_m[\phi] \coloneqq \widehat{\mathcal{I}}_m(\phi^\frown \pi)_{1..n}$ converge in distribution to $T[\phi] \coloneqq T(\phi^\frown \pi')$, as $m \to \infty$.

We begin with an outline of the proof technique. For each $m \in \mathbb{N}$, we aim to define a involutive neural network $\widehat{\mathcal{I}}_m$. We begin by exhibiting an involutive function parametrized by a positive real $\epsilon_m < 1$ depending on m, such that when the auxiliary variable is treated as a random variable, the involutive function matches the cumulative distribution function of the desired state transition arbitrarily well as $\epsilon_m \to 0$. Then we show that this involutive function can be uniformly approximated arbitrarily well by an involutive neural network parameterized by some other positive real $\delta_m < 1$ depending on both ϵ_m and m, as $\delta_m \to 0$. (In the main proof of Theorem 4.1 below, we will impose tighter constraints on ϵ_m and δ_m.)

We will use Hornik's Universal Approximation Theorem to obtain such a uniform approximation.

Theorem 4.2 (Hornik [11, Thm. 3])**.** *Let a, b be positive integers, $F\colon \mathbb{R}^a \to \mathbb{R}^b$ be a continuous function, and $\psi\colon \mathbb{R} \to \mathbb{R}$ be a continuous bounded nonconstant activation function. For any compact set $\Omega \subseteq \mathbb{R}^a$ and $\delta > 0$ there is a neural network consisting of a single hidden layer with activation ψ that induces a continuous function $\widehat{F}_\delta\colon \mathbb{R}^a \to \mathbb{R}^b$ satisfying*

$$\max_{x\in\Omega} \left|F(x) - \widehat{F}_\delta(x)\right|_1 \leq \delta.$$

We now define and prove several facts about some objects that will be useful in the proof.

Define $R_{\epsilon_m}\colon \mathbb{R}^{n+3} \to \mathbb{R}^{n+3}$ by

$$R_{\epsilon_m}(x) \coloneqq \begin{cases} 0^{n+3} & \text{if } q \leq -\frac{1}{2} \\ (q+\frac{1}{2})(T(\phi,\pi)-\phi)^\frown 0^3 & \text{if } -\frac{1}{2} < q < \frac{1}{2} \\ (T(\phi,\pi)-\phi)^\frown 0^3 & \text{if } \frac{1}{2} \leq q, \end{cases}$$

where $\phi \coloneqq x_{1..n}$, $\pi \coloneqq \frac{x_{n+1}}{\epsilon_m}$, and $q \coloneqq x_{n+3} - x_{n+2}$.

Define $S \coloneqq \mathbb{R}^{n+3} \to \mathbb{R}^{n+3}$ by

$$S(x) \coloneqq \begin{cases} 0^{n+3} & \text{if } q \leq -\frac{1}{2} \\ (q+\frac{1}{2})\phi'^\frown 0^3 & \text{if } -\frac{1}{2} < q < \frac{1}{2} \\ \phi'^\frown 0^3 & \text{if } \frac{1}{2} \leq q, \end{cases}$$

where $\phi' \coloneqq x_{1..n}$ and $q \coloneqq x_{n+3} - x_{n+2}$.

Define $g_{\epsilon_m}\colon \mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ by

$$g_{\epsilon_m}(x) \coloneqq x \odot (1^n{}^\frown \epsilon_m 1^{n+6}) + (0^{n+2}{}^\frown 1^\frown 0^{n+2}{}^\frown 1),$$

where $\odot$ denotes pointwise multiplication. Note that its inverse satisfies

$$g_{\epsilon_m}^{-1}(x) = (x - (0^{n+2}{}^\frown 1^\frown 0^{n+2}{}^\frown 1)) \odot (1^n{}^\frown \tfrac{1}{\epsilon_m} 1^{n+6}).$$

Define $h_{\epsilon_m}\colon \mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ by

$$h_{\epsilon_m} \coloneqq \big(x_{1..n+3} + S(v)\big)^\frown v$$

where $v \coloneqq x_{n+4..2n+6} + R_{\epsilon_m}(x_{1..n+3})$.

Note that its inverse satisfies

$$h_{\epsilon_m}^{-1}(x) = w^\frown\big(x_{n+4..2n+6} - R_{\epsilon_m}(w)\big)$$

where $w \coloneqq x_{1..n+3} - S(x_{n+4..2n+6})$.

Let σ be the permutation of $\{1, \ldots, 2n+6\}$ that transposes $n+2$ with $n+3$ and transposes $2n+5$ with $2n+6$ (and leaves all other elements fixed).

Now consider the involutive function $\mathcal{I}_{\epsilon_m}\colon \mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ defined by $\mathcal{I}_{\epsilon_m} \coloneqq g_{\epsilon_m}^{-1} \circ h_{\epsilon_m}^{-1} \circ I_P^{2n+6,\sigma} \circ h_{\epsilon_m} \circ g_{\epsilon_m}$.

Let A_{ϵ_m} denote the event that $\pi_3 - \pi_2 > -\frac{1}{2\epsilon_m}$ and $\pi_{n+6} - \pi_{n+5} > -\frac{1}{2\epsilon_m}$ and $|\pi| < \frac{1}{\epsilon_m}$ all hold.

Note that

$$\lim_{\epsilon_m \to 0} P(A_{\epsilon_m}) = 1. \qquad (\dagger)$$

Lemma 4.3. *Conditioned on the event A_{ϵ_m}, we have*

$$\mathcal{I}_{\epsilon_m}(\phi^\frown\pi)_{1..n} = T(\phi, \pi_1) + \epsilon_m \pi_{4..n+3}.$$

Proof. The event A_{ϵ_m} fully determines the branches of R_{ϵ_m} and S that are taken in the evaluation of $\mathcal{I}_{\epsilon_m}(\phi^\frown\pi)$, which enables us to simplify the expression $\mathcal{I}_{\epsilon_m}(\phi^\frown\pi)_{1..n}$ to the stated form. □

For $\phi \in \Omega$, define the random variable $\mathcal{I}_{\epsilon_m}[\phi] \coloneqq \mathcal{I}_{\epsilon_m}(\phi^\frown\pi)_{1..n}$. The following lemma is immediate.

Lemma 4.4. *Conditioned on the event A_{ϵ_m}, the random variable $\mathcal{I}_{\epsilon_m}[\phi]$ converges in distribution to $T[\phi]$ as $\epsilon_m \to 0$.* □

We will show that conditioned on the event A_{ϵ_m}, and for appropriately small ϵ_m and δ_m, we can approximate $\mathcal{I}_{\epsilon_m}$ arbitrarily well with an involutive neural network $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}$.

Since Ω is a compact subset of $\mathbb{R}^n$, it is bounded, and hence contained in a ball of finite radius $r \in \mathbb{R}$. Let $\Omega^+ \subseteq \mathbb{R}^{n+3}$ be the closure of the ball of radius $7r+10$, and let $\Omega^{++} \subseteq \mathbb{R}^{n+3}$ be the closure of the ball of radius $14r+21$. The sets Ω^+ and Ω^{++} are closed and bounded, hence compact subsets of $\mathbb{R}^{n+3}$.

Since R_{ϵ_m} and S are continuous, by Theorem 4.2, for any $\delta_m > 0$ there are neural networks each with a single hidden layer and sigmoid activation that induce continuous functions $\widehat{R}_{\epsilon_m,\delta_m}\colon \mathbb{R}^{n+3} \to \mathbb{R}^{n+3}$ and $\widehat{S}_{\delta_m}\colon \mathbb{R}^{n+3} \to \mathbb{R}^{n+3}$, satisfying

$$\max_{x\in\Omega^{++}} \left|R_{\epsilon_m}(x) - \widehat{R}_{\epsilon_m,\delta_m}(x)\right|_1 \leq \delta_m$$

and

$$\max_{x\in\Omega^{++}} \left|S(x) - \widehat{S}_{\delta_m}(x)\right|_1 \leq \delta_m.$$

That is, $\widehat{R}_{\epsilon_m,\delta_m}$ and $\widehat{S}_{\delta_m}$ converge uniformly to R_{ϵ_m} and S, respectively, on Ω^{++} as $\delta_m \to 0$.

Define $\widehat{h}_{\epsilon_m,\delta_m}\colon \mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ by

$$\widehat{h}_{\epsilon_m,\delta_m} \coloneqq \big(x_{1..n+3} + \widehat{S}_{\delta_m}(v')\big)^\frown v'$$

where $v' \coloneqq x_{n+4..2n+6} + \widehat{R}_{\epsilon_m,\delta_m}(x_{1..n+3})$.

Note that its inverse satisfies

$$\widehat{h}_{\epsilon_m,\delta_m}^{-1}(x) = w'^\frown\big(x_{n+4..2n+6} - \widehat{R}_{\epsilon_m,\delta_m}(w')\big)$$

where $w' \coloneqq x_{1..n+3} - \widehat{S}_{\delta_m}(x_{n+4..2n+6})$.

Now form the involutive neural network that induces a function $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m} : \mathbb{R}^{2n+6} \to \mathbb{R}^{2n+6}$ defined as follows,

$$\widehat{\mathcal{I}}_{\epsilon_m,\delta_m} := g_{\epsilon_m}^{-1} \circ \widehat{h}_{\epsilon_m,\delta_m}^{-1} \circ I_P^{2n+6,\sigma} \circ \widehat{h}_{\epsilon_m,\delta_m} \circ g_{\epsilon_m},$$

by composing the layers or neural nets corresponding to each term in the function definition.

Lemma 4.5. *As $\delta_m \to 0$, the function $\widehat{h}_{\epsilon_m,\delta_m}$ converges uniformly to h_{ϵ_m} on $\Omega \times B_3(0^{n+6})$ and its inverse $\widehat{h}_{\epsilon_m,\delta_m}^{-1}$ converges to $h_{\epsilon_m}^{-1}$ on $\Omega^+ \times \Omega^+$.*

Proof. First observe that $|S(x)| \le |x|$ and $|R_{\epsilon_m}(x)| \le 2r$, so that $|h_{\epsilon_m}(x)| \le 3|x| + 4r$.

For $x \in \Omega \times B_3(0^{n+6})$, we have $|x_{n+4..2n+6} + \widehat{R}_{\epsilon_m,\delta_m}(x_{1..n+3})| < |\widehat{R}_{\epsilon_m,\delta_m}(x_{1..n+3})| + 3 < |R_{\epsilon_m}(x_{1..n+3})| + 4 < 2r + 4$, and hence we have $x_{n+4..2n+6} + \widehat{R}_{\epsilon_m,\delta_m}(x_{1..n+3}) \in \Omega^{++}$. Thus for $x \in \Omega \times B_3(0^{n+6})$, all applications of $\widehat{R}_{\epsilon_m,\delta_m}$ and $\widehat{S}_{\delta_m}$ in $\widehat{h}_{\epsilon_m,\delta_m}(x)$ are on points in Ω^{++} (where $\widehat{R}_{\epsilon_m,\delta_m}$ and $\widehat{S}_{\delta_m}$ converge uniformly to R_{ϵ_m} and S), and so $\widehat{h}_{\epsilon_m,\delta_m}$ converges to h_{ϵ_m} on $\Omega \times B_3(0^{n+6})$ as $\delta_m \to 0$. Furthermore, the convergence is uniform, since it is formed from sums and compositions of uniformly converging functions.

For $x \in \Omega^+ \times \Omega^+$ we have $|x_{1..n+3} - \widehat{S}_{\delta_m}(x_{n+4..2+6})| < |\widehat{S}_{\delta_m}(x_{n+4..2+6})| + 7r + 10 < |S(x_{n+4..2+6})| + 7r + 11 < 14r + 21$, and hence we have $x_{n+4..2n+6} - \widehat{S}_{\delta_m}(x_{1..n+3}) \in \Omega^{++}$. Thus for $x \in \Omega^+ \times \Omega^+$, all applications of $\widehat{R}_{\epsilon_m,\delta_m}$ and $\widehat{S}_{\delta_m}$ in $\widehat{h}_{\epsilon_m,\delta_m}^{-1}(x)$ are on points in Ω^{++} (where $\widehat{R}_{\epsilon_m,\delta_m}$ and $\widehat{S}_{\delta_m}$ converge to R_{ϵ_m} and S) and so $\widehat{h}_{\epsilon_m,\delta_m}^{-1}$ converges to $h_{\epsilon_m}^{-1}$ on $\Omega^+ \times \Omega^+$ as $\delta_m \to 0$. □

Note that by Lemma 4.5, we have

$$\max_{x \in \Omega \times B_{1/\epsilon_m}(0^{n+6})} \left| \widehat{h}_{\epsilon_m,\delta_m}(g(x)) - h_{\epsilon_m}(g(x)) \right| < 1 \quad (\ddagger)$$

for sufficiently small δ_m.

Lemma 4.6. *Consider the random variable $x := \phi^\frown\pi$. Conditioned on the event A_{ϵ_m}, the function $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}(x)$ converges pointwise to $\mathcal{I}_{\epsilon_m}(x)$, as $\delta_m \to 0$.*

Proof. Condition on A_{ϵ_m} and assume δ_m is sufficiently small that ($\ddagger$) holds. Then notice that $x \in \Omega \times B_{1/\epsilon_m}(0^{n+6})$, so that $g(x) \in \Omega \times B_3(0^{n+6})$, and hence $\left|I_P^{2n+6,\sigma}(\widehat{h}_{\epsilon_m,\delta_m}(g(x)))\right| = \left|\widehat{h}_{\epsilon_m,\delta_m}(g(x))\right| < |h_{\epsilon_m}(g(x))| + 1 < 3|g(x)| + 4r + 1 < 7r + 10$. Thus $I_P^{2n+6,\sigma}\left(\widehat{h}_{\epsilon_m,\delta_m}(g(x))\right) \in \Omega^+ \times \Omega^+$. Hence when evaluating $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}(x)$, the inputs to both $\widehat{h}_{\epsilon_m,\delta_m}$ and $\widehat{h}_{\epsilon_m,\delta_m}^{-1}$ are in the domains where their respective convergence properties stated in Lemma 4.5 hold. Therefore each function occurring in the definition of $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}$ convergence pointwise to the corresponding function in the definition of $\mathcal{I}_{\epsilon_m}$. Further, all such functions are continuous, and so the result holds. □

For $\phi \in \Omega$, define the random variable $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi] := \widehat{\mathcal{I}}_{\epsilon_m,\delta_m}(\phi^\frown\pi)_{1..n}$. Let $\xi_m \sim N(0^n, \epsilon_m \mathrm{Id}_n)$ be an independently chosen Gaussian.

The following result is immediate from Lemma 4.6.

Lemma 4.7. *Conditioned on the event A_{ϵ_m}, the random variable $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]$ converges in distribution to $\mathcal{I}_{\epsilon_m}[\phi]$ as $\delta_m \to 0$.* □

We now prove Theorem 4.1.

Proof of Theorem 4.1. Fix $m \in \mathbb{N}$; we will define $\widehat{\mathcal{I}}_m$ such that the sequence $\left\{\widehat{\mathcal{I}}_m\right\}_{m \in \mathbb{N}}$ has the desired convergence property using Lemmas 4.4 and 4.7.

We may decompose the CDF of $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]$ in terms of A_{ϵ_m} and $\bar{A}_{\epsilon_m}$:

$$\begin{aligned} F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]}(\phi') = P(A_{\epsilon_m}) \cdot F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]|A_{\epsilon_m}}(\phi') \\ + (1 - P(A_{\epsilon_m})) \cdot F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]|\bar{A}_{\epsilon_m}}(\phi'). \end{aligned}$$

Now we form a sequence $\left\{\widehat{\mathcal{I}}_m[\phi]\right\}_{m \in \mathbb{N}}$ and demonstrate that $\lim_{m\to\infty} F_{\widehat{\mathcal{I}}_m[\phi]}(\phi') = F_{T[\phi]}(\phi')$ at all points of continuity ϕ' of $T[\phi]$.

By Lemma 4.4 choose ϵ_m such that

$$|F_{T[\phi]}(\phi') - (F_{\mathcal{I}_{\epsilon_m}[\phi]|A_{\epsilon_m}})(\phi')| < \tfrac{1}{3m}$$

holds for all points of continuity ϕ' of $T[\phi]$ and $P(A_{\epsilon_m}) < \frac{1}{3m}$ holds, which is possible by (†).

By Lemma 4.7 choose δ_m so that

$$\left|(F_{\mathcal{I}_{\epsilon_m}[\phi]|A_{\epsilon_m}})(\phi') - F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]|A_{\epsilon_m}}(\phi')\right| < \tfrac{1}{3m}$$

holds for all points of continuity ϕ' of $T[\phi]$ and ($\ddagger$) holds. This is possible because conditioned on A_{ϵ_m}, the random variable $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]$ converges in distribution to $\mathcal{I}_{\epsilon_m}[\phi]$ and every point of continuity of $T[\phi]$ is also a point of continuity of $\mathcal{I}_{\epsilon_m}[\phi]$.

Now define $\widehat{\mathcal{I}}_m := \widehat{\mathcal{I}}_{\epsilon_m,\delta_m}$. Observe that for any point of continuity ϕ' of $T[\phi]$, we have

$$\begin{aligned} &\left|F_{\widehat{\mathcal{I}}_m[\phi]}(\phi') - F_{T[\phi]}(\phi')\right| \\ &\quad = \left|F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]}(\phi') - F_{T[\phi]}(\phi')\right| \\ &\quad < \left|F_{\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}[\phi]|A_{\epsilon_m}}(\phi') - F_{T[\phi]}(\phi')\right| + \tfrac{1}{3m} \\ &\quad < \left|F_{\mathcal{I}_{\epsilon_m}[\phi]|A_{\epsilon_m}}(\phi') - F_{T[\phi]}(\phi')\right| + \tfrac{2}{3m} \\ &\quad < \tfrac{1}{m}. \end{aligned}$$

Finally, considering this fact for all $m \in \mathbb{N}$, we see that $\widehat{\mathcal{I}}_m[\phi]$ converges in distribution to $T[\phi]$. □

Observe that universality holds even for the special case of volume-preserving involutive networks: the architecture $\widehat{\mathcal{I}}_{\epsilon_m,\delta_m}$ used in the constructive proof, defined to be

$$\widehat{\mathcal{I}}_{\epsilon_m,\delta_m} \coloneqq g_{\epsilon_m}^{-1} \circ \widehat{h}_{\epsilon_m,\delta_m}^{-1} \circ I_P^{2n+6,\sigma} \circ \widehat{h}_{\epsilon_m,\delta_m} \circ g_{\epsilon_m},$$

has a Jacobian whose determinant has magnitude 1, since $\widehat{h}_{\epsilon_m,\delta_m}$ has the structure of an additive coupling layer and the constant Jacobians of $g_{\epsilon_m}^{-1}$ and g_{ϵ_m} cancel.

Having established the universality of deep volume-preserving involutive generative models, we now review the following known result showing that volume-preserving involutive functions can be used as valid proposals within an MCMC algorithm.

4.2 Volume-preserving involutive functions lead to Metropolis–Hastings proposals satisfying detailed balance

The proof of detailed balance for Hybrid Monte Carlo relies on the fact that Hamiltonian dynamics composed with negating momentum is involutive and volume preserving. It is also known that volume-preserving involutive functions lead to Metropolis–Hastings proposals satisfying detailed balance (see, e.g., [5]), but we were unable to find a published argument. For completeness we describe here a procedure for using any volume-preserving involutive function as a proposal, and prove its correctness.

Our goal is to use a volume-preserving involutive function f_θ to construct a Markov process with $P_S(\phi)$ as a stationary distribution. To do this, we will find a transition such that the transition probabilities $P_M(\phi \mapsto \phi')$ satisfy the detailed balance condition:

$$P_S(\phi)P_M(\phi \mapsto \phi') = P_S(\phi')P_M(\phi' \mapsto \phi)$$

We follow the original derivation of Hybrid Monte Carlo [9], since the structure of the proof is similar.

In order to make a transition, we do the following.

1. Introduce an auxiliary random variable π with probability density P_G.
2. Propose a transition drawn from $P_H((\phi,\pi) \mapsto (\phi',\pi'))$ according to the volume-preserving involutive function f_θ:
$$P_H((\phi,\pi) \mapsto (\phi',\pi')) = \delta[(\phi',\pi') - f_\theta((\phi,\pi))].$$
3. Accept or reject that transition according to the Metropolis–Hastings acceptance criterion
$$P_A((\phi,\pi) \mapsto (\phi',\pi')) = \min\Big(1, \frac{P_S(\phi')P_G(\pi')P_H((\phi',\pi') \mapsto (\phi,\pi))}{P_S(\phi)P_G(\pi)P_H((\phi,\pi) \mapsto (\phi',\pi'))}\Big).$$
4. Marginalize over the auxiliary variable π.

Formally, we define our transition probability by

$$P_M(\phi \mapsto \phi') = \int \Big(P_G(\pi)\, P_H((\phi,\pi) \mapsto (\phi',\pi')) \; P_A((\phi,\pi) \mapsto (\phi',\pi'))\Big) d\pi d\pi'.$$

We now show that this transition satisfies the detailed balance condition.

Lemma 4.8. *Applying f_θ within a Dirac δ distribution leaves the δ unchanged:*

$$\delta[x - y] \;=\; \delta[f_\theta(x) - f_\theta(y)]$$

Proof. For arbitrary F we have

$$F(y) = \int_\Omega \delta[x-y]F(x)\,dx$$

and

$$\begin{aligned}
&\int_\Omega \delta[f_\theta(x)-f_\theta(y)]F(x)\,dx \\
&\quad= \int_{f_\theta(\Omega)} \delta[u - f_\theta(y)]F(f_\theta^{-1}(u))\,\frac{du}{|\det f'_\theta(x)|} \\
&\quad= \int_{f_\theta(\Omega)} \delta[u - f_\theta(y)]F(f_\theta^{-1}(u))\,du \\
&\qquad\qquad \text{(since } f_\theta \text{ preserves volume)} \\
&\quad= F(f_\theta^{-1}(f_\theta(y))) \\
&\quad= F(y),
\end{aligned}$$

and so

$$\int_\Omega \delta[x-y]F(x)\,dx \;=\; \int_\Omega \delta[f(x)-f(y)]F(x)\,dx.$$

Hence $\delta[x-y] = \delta[f(x)-f(y)]$. □

Lemma 4.9. *P_H is symmetric:*

$$P_H((\phi,\pi) \mapsto (\phi',\pi')) = P_H((\phi',\pi') \mapsto (\phi,\pi)).$$

Proof. We have

$$\begin{aligned}
&P_H((\phi,\pi) \mapsto (\phi',\pi')) \\
&\quad= \delta[(\phi',\pi') - f_\theta((\phi,\pi))] \\
&\quad= \delta[f_\theta((\phi',\pi')) - f_\theta \circ f_\theta((\phi,\pi))] \\
&\qquad\qquad \text{(by Lemma 4.8)} \\
&\quad= \delta[f_\theta((\phi',\pi')) - (\phi,\pi)] \\
&\qquad\qquad \text{(since } f_\theta \text{ is involutive)} \\
&\quad= \delta[(\phi,\pi) - f_\theta((\phi',\pi'))] \\
&\qquad\qquad \text{(since } \delta[x] = \delta[-x]) \\
&\quad= P_H((\phi',\pi') \mapsto (\phi,\pi)),
\end{aligned}$$

establishing the lemma. □

Lemma 4.10. *P_A satisfies the following simpler form:*

$$P_A((\phi,\pi)\mapsto(\phi',\pi')) = \min\Big(1, \frac{P_S(\phi')P_G(\pi')}{P_S(\phi)P_G(\pi)}\Big).$$

Proof. Apply Lemma 4.9 to the definition of P_A. □

Lemma 4.11. *$P_S P_G P_A$ is symmetric:*

$$P_S(\phi)P_G(\pi)P_A((\phi,\pi)\mapsto(\phi',\pi')) = P_S(\phi')P_G(\pi')P_A((\phi',\pi')\mapsto(\phi,\pi)).$$

Proof. We have

$$\begin{aligned}
&P_S(\phi)P_G(\pi)P_A((\phi,\pi)\mapsto(\phi',\pi')) \\
&\quad= P_S(\phi)P_G(\pi)\ \min\Big(1, \frac{P_S(\phi')P_G(\pi')}{P_S(\phi)P_G(\pi)}\Big) \\
&\quad= \min\big(P_S(\phi)P_G(\pi), P_S(\phi')P_G(\pi')\big) \\
&\quad= P_S(\phi')P_G(\pi')\ \min\Big(\frac{P_S(\phi)P_G(\pi)}{P_S(\phi')P_G(\pi')}, 1\Big) \\
&\quad= P_S(\phi')P_G(\pi')P_A((\phi',\pi')\mapsto(\phi,\pi)),
\end{aligned}$$

as desired. □

Theorem 4.12. *The Markov chain defined by transitions P_M has P_S as a stationary distribution.*

Proof. It suffices to show that P_M satisfies the detailed balance condition

$$P_S(\phi)P_M(\phi\mapsto\phi') = P_S(\phi')P_M(\phi'\mapsto\phi). \qquad (\star)$$

We have

$$\begin{aligned}
&P_S(\phi)P_M(\phi\mapsto\phi') \\
&\quad= \int \Big(P_S(\phi)P_G(\pi)P_H((\phi,\pi)\mapsto(\phi',\pi')) \\
&\qquad\qquad P_A((\phi,\pi)\mapsto(\phi',\pi'))\Big)d\pi d\pi' \\
&\quad= \int \Big(P_S(\phi')P_G(\pi')P_H((\phi,\pi)\mapsto(\phi',\pi')) \\
&\qquad\qquad P_A((\phi',\pi')\mapsto(\phi,\pi))\Big)d\pi d\pi' \\
&\qquad\qquad\qquad \text{(by Lemma 4.11)} \\
&\quad= \int \Big(P_S(\phi')P_G(\pi')P_H((\phi',\pi')\mapsto(\phi,\pi)) \\
&\qquad\qquad P_A((\phi',\pi')\mapsto(\phi,\pi))\Big)d\pi d\pi' \\
&\qquad\qquad\qquad \text{(by Lemma 4.9)} \\
&\quad= P_S(\phi')P_M(\phi'\mapsto\phi),
\end{aligned}$$

and so ($\star$) holds. □

We have established that for Markov processes generated by our transition, the desired distribution is stationary, by showing that it satisfies detailed balance. This implies, when the chain is ergodic, that the MCMC procedure eventually generates samples which are arbitrarily close in total variation distance to the posterior distribution.

5 Training and sampling algorithm

Having established the generality of our involutive MCMC procedure, we now describe a method for training optimized involutive transition kernels.

As discussed in [21], a useful transition kernel satisfies three criteria: 1) low bias in the limit; 2) fast convergence; and 3) low autocorrelation.

Volume-preserving involutive functions lead to transition kernels with zero bias in the limit (as we saw in Section 4.2, assuming ergodicity), so criterion 1 is satisfied.

Previous work has shown that using a "Markov-GAN" or MGAN objective [21] can satisfy criterion 2, by finding a good tradeoff between proposals being near the posterior and a high proposal acceptance rate. This objective is

$$\begin{aligned}
\min_\theta \max_D \Big\{&\mathbb{E}_{x\sim p_d}[D(x)] - \lambda\mathbb{E}_{\bar{x}\sim\chi_\theta^b}[D(\bar{x})] \\
&- (1-\lambda)\mathbb{E}_{x_d\sim p_d,\,\bar{x}\sim T_\theta^m(\bar{x}|x_d)}[D(\bar{x})]\Big\},
\end{aligned}$$

where θ and D are the parameters of the generator and discriminator, p_d is the true posterior, χ_θ^b is the distribution of samples from the Markov chain after b transitions from X (the given sampling distribution), $T_\theta^m(\bar{x}|x_d)$ is the distribution of samples from the Markov chain after m transitions starting from a sample from the true posterior, and λ is a free parameter.

We train using a new, computationally efficient lower-variance estimator of $\mathbb{E}_{\bar{x}\sim\chi_\theta^b}[D(\bar{x})]$ which enables accurate training through multiple MCMC transitions, even when far from the posterior. This enables us to train using $\lambda = 1$, which is desirable since $\lambda < 1$ requires sampling from the posterior, which is not always tractable. In contrast, the training method of A-NICE-MC ignores inverse transitions, an approximation which is justified only if chains converge quickly to the posterior. We could optimize the true objective by estimating this expectation with individual samples from χ_θ^b, but we instead form a lower variance approximation. We do this by fixing a sampled set of auxiliary variables each training step, and then computing the exact expectation conditioned on the use of those auxiliary variables. This approximation may bias the true MGAN objective; it has been helpful in practice, but more analysis is required to determine

Algorithm 1 Involutive Neural MCMC Training

Let X and Y be sampling distributions for states and auxiliary variables respectively.
Let b be the number of steps of MCMC we consider during training.
Let `training_steps` be the desired number of training steps.
Let w be the permitted magnitude of weights for WGAN training.
Initialize a neural net D (the discriminator) and an involutive neural net G (the generator).
for `step` in `training_steps` **do**
  Sample a true state $\widehat{\phi}$ from the posterior Φ.
  Sample an initial state $\phi_0 \sim X$.
  for i in $\{0, \ldots, b-1\}$ **do**
    Sample an auxiliary variable $\pi \sim Y$.
    $(\phi_{i+1}, \pi') \leftarrow G(\phi_i, \pi)$
    $A_{i \mapsto i+1} \leftarrow \min\left(1, \frac{f_\Phi(\phi_{i+1}) f_Y(\pi')}{f_\Phi(\phi_i) f_Y(\pi)}\right)$
  end for
  for t in $\{0, \ldots, b\}$ **do**
    $\chi_{0,t} \leftarrow (1 - A_{0 \mapsto 1})^t$
  end for
  for j in $\{1, \ldots, b-1\}$ **do**
    $\chi_{j,j-1} \leftarrow 0$
    for t in $j, \ldots, b$ **do**
      $\chi_{j,t} \leftarrow \chi_{j-1,t-1} A_{j-1 \mapsto j} + \chi_{j,t-1}(1 - A_{j \mapsto j+1})$
    end for
  end for
  $P(i) \coloneqq \chi_{i,b}$
  if `step` $\%\, 2 == 0$ **then**
    $D_{\text{loss}} \leftarrow D(\widehat{\phi}) - \mathbb{E}_{i \sim P}[D(\phi_i)]$
    Update D by objective D_{loss} via RMSProp.
    Clamp weights of D in range $[-w, w]$.
  else
    $G_{\text{loss}} \leftarrow \mathbb{E}_{i \sim P}[D(\phi_i)]$
    Update G by objective G_{loss} via RMSProp.
  end if
end for
return G

when this approximation is justified. Specifically, we optimize

$$\min_\theta \max_D \left\{ \mathbb{E}_{x \sim p_d}[D(x)] - \mathbb{E}_{\bar{x} \sim \chi_\theta^b}[D(\bar{x})] \right\}$$

where χ_θ^b is the distribution of states after b transitions of a random Markov chain with states ϕ_i and transition probabilities $A_{i \mapsto j}$ defined inductively over i by

$$\begin{aligned} \pi_i &\sim Y \\ \phi_0 &\sim X \\ \phi_{i+1} \frown \pi_i' &= G_\theta(S_i, \pi_i) \\ A_{i \mapsto j} &= \delta_{j,i+1} \min\left(1, \frac{f_X(\phi_j) f_Y(\pi_i')}{f_X(\phi_i) f_Y(\pi_i)}\right), \end{aligned}$$

where X is a sampling distribution of initial states, Y is a sampling distribution for auxiliary variables, G_θ is an involutive neural network parameterized by θ, Φ is the true posterior, δ is the Kronecker delta, and f_X and f_Y are the densities of X and Y respectively.

Algorithm 2 Involutive Neural MCMC Sampling

Let G be the generator network obtained during training.
Let X and Y be the sampling distributions for states and auxiliary variables respectively used during training.
Let b be the number of steps of MCMC to use, which may be different than that used in training.
Sample initial state $\phi \sim X$.
for i in $\{1, \ldots, b\}$ **do**
  Sample auxiliary variable $\pi \sim Y$.
  $(\phi', \pi') \leftarrow G(\phi, \pi)$
  $p \leftarrow \frac{f_\Phi(\phi) f_Y(\pi')}{f_\Phi(\phi) f_Y(\pi)}$
  With probability p, update $\phi \leftarrow \phi'$.
end for
return ϕ

Previous work [21] has also shown that training using a pairwise discriminator can reduce autocorrelation. One trains the discriminator to distinguish pairs of samples from the same chain from pairs of samples from the posterior. When it is possible to generate true independent pairs of samples from the posterior, this technique can be used to reduce autocorrelation and satisfy criterion 3.

Algorithms 1 and 2 provide pseudocode describing our Wasserstein-GAN-style training procedure [2] and our sampling procedure. In practice, both of these algorithms should be batched over chains, and all probability calculations should be done in log space.

If one does not have access to a generative model for true posterior samples, one can instead bootstrap [21].

6 Experimental Results

We train a deep volume-preserving involutive neural network to serve as a neural proposal for sampling from a mixture of six Gaussians (mog6 from [21]) and compare its convergence rate to that of A-NICE-MC.

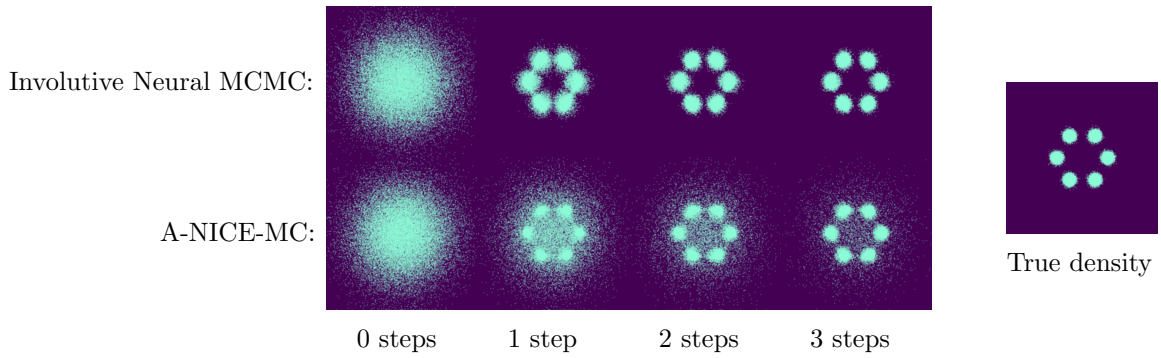

Figure 4: Density plots of samples from A-NICE-MC and Involutive Neural MCMC. Note the outliers remaining in A-NICE-MC, which are samples for which a forward transition has never been proposed. Almost every sample from Involutive Neural MCMC is near the posterior after only one step.

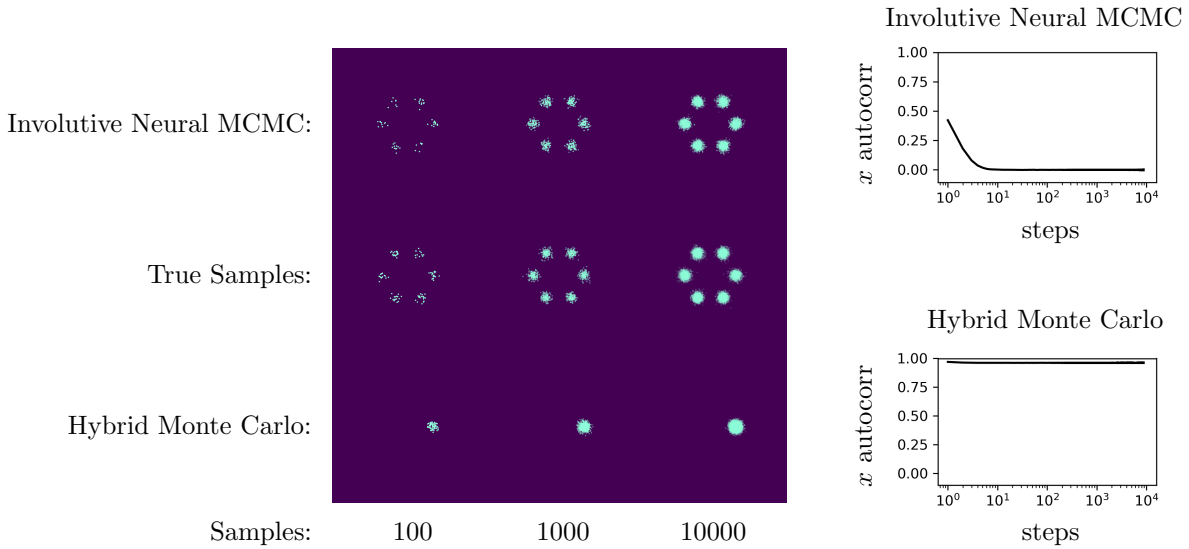

Figure 5: Density plots of samples from a single long chain of Involutive Neural MCMC and HMC. Note that Involutive Neural MCMC mixes completely within 10 steps, whereas HMC does not mix even after 10000 steps.

Architecture: In this experiment, we use states $\phi \in \mathbb{R}^2$ and auxiliary variables $\pi \sim N(0^{30}, \mathrm{Id}_{30})$. The higher dimension of π helps increase the width, and thus capacity, of our deep involutive network.

Our discriminator is a neural network consisting of a single hidden layer of width 64 and ReLU activation.

Our generator is an involutive neural network consisting of a symmetric composition $I_F^{n,g} \circ I_P^{n,\sigma} \circ I_F^{n,h} \circ I_P^{n,\sigma} \circ I_F^{n,g}$ where σ is a uniform random involutive permutation (chosen once at network initialization), and g and h are invertible neural networks. Each of g and h consists of three composed invertible blocks [1]. The first and third invertible blocks each use as their nonlinear functions two densely connected neural networks with a single hidden layer of width 8 times its input dimension and ReLU activation. The second invertible block is a uniformly random permutation.

Results: Accepted transitions from both A-NICE-MC and Involutive Neural MCMC converge very quickly. However, initial proposals from A-NICE-MC have acceptance probabilities of about 50%, whereas we observe nearly 100% acceptance for proposals from Involutive Neural MCMC. See Figs. 4, 5, and 6 for density plots and a comparison of convergence rates.

7 Discussion

This paper has introduced new deep learning building blocks for constructing involutive neural networks, deep involutive generative models, and methods to

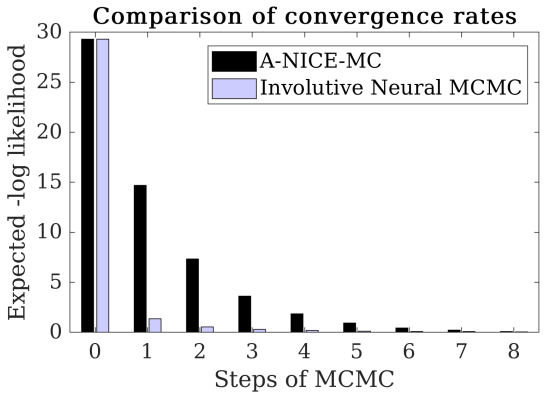

Figure 6: Expected negative log likelihood of samples from A-NICE-MC and Involutive Neural MCMC relative to samples from the true posterior.

constrain these to preserve volume; it has also proved that these generative models are universal approximators for probability kernels, and it has shown how to train and use deep volume-preserving involutive generative models for fast neural MCMC. This paper has also demonstrated that Involutive Neural MCMC can converge more rapidly than A-NICE-MC, a recently introduced neural MCMC technique, and that it is possible for Involutive Neural MCMC to switch modes more effectively than Hybrid Monte Carlo.

Much more work is needed to empirically study the performance of Involutive Neural MCMC on a broader class of problems and involutive architectures. The relationship between training time, network capacity, and convergence rate are not yet clear, even on simple examples. We note that because deep involutive generative models are self-inverting, it may be feasible to use recently introduced auxiliary variable techniques to assess the convergence rate of Involutive Neural MCMC to the true posterior, in terms of KL divergence [6].

There is a widespread need for techniques that construct fast, accurate MCMC proposals for broad classes of Bayesian inference problems. Involutive Neural MCMC offers a way to learn MCMC proposals using neural networks without requiring the ability to analytically calculate output probability densities of those networks. A broad class of GAN-based techniques thus become available to MCMC algorithm designers. Also, because volume-preserving involutive generative models are universal approximators, they can in principle learn arbitrarily good proposals given enough network capacity and training compute time. We hope the flexibility afforded by Involutive Neural MCMC leads to the development of many fast neural MCMC schemes for challenging inference problems.

Acknowledgements

The authors would like to thank Ian Hunter, Sarah Bricault, Marco Cusumano-Towner, and Jayson Lynch for helpful discussions, and Sam Power for identifying an error in an earlier version of this paper. This research was supported by Fonterra, IBM (via the MIT-IBM Watson AI Lab), and gifts from the Siegel Family Foundation and the Aphorism Foundation.

References

[1] L. Ardizzone, J. Kruse, S. Wirkert, D. Rahner, E. W. Pellegrini, R. S. Klessen, L. Maier-Hein, C. Rother, and U. Köthe. Analyzing inverse problems with invertible neural networks. In *Int. Conf. Learning Representations (ICLR)*, 2019.

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein Generative Adversarial Networks. In *Proc. 34th Int. Conf. Machine Learning (ICML)*, 2017.

[3] J. Arndt. *Generating random permutations*. PhD thesis, Australian National University, 2010.

[4] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, editors. *Handbook of Markov Chain Monte Carlo*. CRC Press, 2011.

[5] M. Cusumano-Towner. Detailed balance for Gen's `general_mh` operator, 2018. URL https://github.com/probcomp/Gen.jl/blob/b9d72b/docs/mcmc.tex.

[6] M. Cusumano-Towner and V. K. Mansinghka. AIDE: An algorithm for measuring the accuracy of probabilistic inference algorithms. In *Adv. Neural. Inf. Process. Syst. (NeurIPS) 30*, pages 3000–3010, 2017.

[7] L. Dinh, D. Krueger, and Y. Bengio. NICE: Nonlinear Independent Components Estimation. In *Int. Conf. Learning Representations (ICLR)*, 2015.

[8] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. In *Int. Conf. Learning Representations (ICLR)*, 2017.

[9] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Phys. Lett. B*, 195(2):216–222, 1987.

[10] J. Dunkley, M. Bucher, P. G. Ferreira, K. Moodley, and C. Skordis. Fast and reliable Markov chain Monte Carlo technique for cosmological parameter estimation. *Mon. Notices Royal Astron. Soc.*, 356 (3):925–936, 2005.

[11] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4 (2):251–257, 1991.

[12] J.-H. Jacobsen, A. W. M. Smeulders, and E. Oyallon. i-RevNet: Deep Invertible Networks. In *Int. Conf. Learning Representations (ICLR)*, 2018.

[13] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Int. Conf. Learning Representations (ICLR)*, 2014.

[14] T. D. Kulkarni, P. Kohli, J. B. Tenenbaum, and V. Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognition (CVPR)*, pages 4390–4399, 2015.

[15] J. Levine and H. M. Nahikian. On the construction of involutory matrices. *Amer. Math. Monthly*, 69 (4):267–272, 1962.

[16] D. Levy, M. D. Hoffman, and J. Sohl-Dickstein. Generalizing Hamiltonian Monte Carlo with neural networks. In *Int. Conf. Learning Representations (ICLR)*, 2018.

[17] A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *Proc. 31st Int. Conf. Machine Learning (ICML)*, pages 1791–1799, 2014.

[18] J. M. Pak, C. K. Ahn, Y. S. Shmaliy, and M. T. Lim. Improving reliability of particle filter-based localization in wireless sensor networks via hybrid particle/FIR filtering. *IEEE Trans. Ind. Informat.*, 11(5):1089–1098, 2015.

[19] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proc. 31st Int. Conf. Machine Learning (ICML)*, pages 1278–1286, 2014.

[20] F. Ronquist, M. Teslenko, P. van der Mark, D. L. Ayres, A. Darling, S. Höhna, B. Larget, L. Liu, M. A. Suchard, and J. P. Huelsenbeck. MrBayes 3.2: Efficient Bayesian phylogenetic inference and model choice across a large model space. *Syst. Biol.*, 61(3):539–542, 2012.

[21] J. Song, S. Zhao, and S. Ermon. A-NICE-MC: Adversarial training for MCMC. In *Adv. Neural. Inf. Process. Syst. (NeurIPS) 30*, pages 5140–5150, 2017.

[22] T. Wang, Y. Wu, D. A. Moore, and S. J. Russell. Meta-learning MCMC proposals. In *Adv. Neural. Inf. Process. Syst. (NeurIPS) 31*, pages 4146–4156. 2018.