Involutive MCMC: a Unifying Framework

Kirill Neklyudov [1 2] **Max Welling** [3 4] **Evgenii Egorov** [5] **Dmitry Vetrov** [1 2]

Abstract

Markov Chain Monte Carlo (MCMC) is a computational approach to fundamental problems such as inference, integration, optimization, and simulation. The field has developed a broad spectrum of algorithms, varying in the way they are motivated, the way they are applied and how efficiently they sample. Despite all the differences, many of them share the same core principle, which we unify as the Involutive MCMC (iMCMC) framework. Building upon this, we describe a wide range of MCMC algorithms in terms of iMCMC, and formulate a number of "tricks" which one can use as design principles for developing new MCMC algorithms. Thus, iMCMC provides a unified view of many known MCMC algorithms, which facilitates the derivation of powerful extensions. We demonstrate the latter with two examples where we transform known reversible MCMC algorithms into more efficient irreversible ones.

1. Introduction

Machine learning algorithms with stochastic latent variables or parameters (a.k.a. Bayesian models) require often intractable posterior inference over these unobserved random variables. The most popular approach these days are variational approximations where possibly complex and/or amortized posterior distributions are optimized and used for inference: e.g. $q(z|x)$ for latent variables or $q(\theta)$ for parameters. However, these distributions are usually biased and may not be easy to optimize.

A completely different class of algorithms is given by MCMC algorithms. Here we design a stochastic process that eventually samples from the correct (i.e. target) distribution. This has the advantage that we are guaranteed to obtain unbiased samples at the cost of possibly slow mixing and long burn-in times. There is a huge literature on MCMC algorithms across many different scientific fields such as statistics, bio-informatics, physics, chemistry, machine learning etc.

Name & Citation	Appendix
Metropolis-Hastings (Hastings, 1970)	B.1
Mixture Proposal (Habib & Barber, 2018)	B.2
Multiple-Try Metropolis (Liu et al., 2000)	B.3
Sample-Adaptive MCMC (Zhu, 2019)	B.4
Reversible-Jump MCMC (Green, 1995)	B.5
Hybrid Monte Carlo (Duane et al., 1987)	B.6
RMHMC (Girolami & Calderhead, 2011)	B.7
NeuTra (Hoffman et al., 2019)	B.8
A-NICE-MC (Song et al., 2017)	B.9
L2HMC (Levy et al., 2017)	B.10
Persistent HMC (Horowitz, 1991)	B.11
Gibbs (Geman & Geman, 1984)	B.12
Look Ahead (Sohl-Dickstein et al., 2014)	B.13
NRJ (Gagnon & Doucet, 2019)	B.14
Lifted MH (Turitsyn et al., 2011)	B.15

Table 1: List of algorithms that we describe by the Involutive MCMC framework. See their descriptions and formulations in terms of iMCMC in corresponding appendices.

More recently, researchers have started to design MCMC kernels by using learnable components, in particular flows which are also often used in variational approaches (Song et al., 2017; Hoffman et al., 2019). We anticipate these hybrid approaches will become an important family of inference methods for approximate inference.

In this paper we provide a unifying framework for MCMC algorithms, including the hybrid approaches mentioned above. We call this Involutive MCMC (iMCMC). We provide an overview of many existing MCMC methods reformulated as iMCMC algorithms. See table 1 for the list. The power of our framework is the ease with which one can now start to combine and improve these algorithms using a number of "tricks" that we discuss extensively. We provide two examples for how this generalization might work in the experiments section. We hope our work might spur the

[1]Samsung AI Center Moscow [2]Samsung-HSE Laboratory, National Research University Higher School of Economics [3]University of Amsterdam [4]Canadian Institute for Advanced Research [5]Skolkovo Institute of Science and Technology. Correspondence to: Kirill Neklyudov <k.necludov@gmail.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

development of new approximate inference methods based on ideas from MCMC inference.

We summarize the main contributions of the paper as follows.

- In Section 2, we introduce the Involutive MCMC formalism to describe a wide range of MCMC algorithms. This formalism provides a simple way to verify invariance of the target distribution, at the same time highlighting the main constraints this invariance put on the design of samplers.
- In Section 3, we summarize the main ideas of different MCMC algorithms in the literature, providing the reader with the set of "tricks". These tricks provide a simple way to incorporate new features into a sampler in order to increase its efficiency, without re-deriving the fixed point equation or validating detailed balance.
- Finally, in Section 4, we demonstrate the potential of iMCMC formalism deriving irreversible counterparts of existent MCMC methods, and demonstrate empirical gains on different target distributions.

2. Involutive MCMC

MCMC algorithms are designed by specifying a transition probability $t(x' \mid x)$ that maps a distribution p_t to a new distribution p_{t+1}. Repeatedly applying this map to an initial distribution p_0 should result in the target distribution p. One can show that this is guaranteed if the map is ergodic (whose average over space is equal to its average over time, which is the number of applications of the map here) and leaves the target distribution invariant:

$$\int dx t(x' \mid x) p(x) = p(x'). \tag{1}$$

Usually, one can not compute the full integral and so we approximate the process of iteratively applying the transition kernel by sampling a single sample from it at every iteration. At convergence, these samples will then be guaranteed to be distributed according to the target distribution. In the rest of the paper, we will refer to equation (1) as *the fixed point equation*.

The transition kernel is usually stochastic, but can also be deterministic, in which case it represents an iterated map. Applying it to a sample from p_0 will thus generate a deterministic trajectory. To be ergodic, this trajectory can not be periodic and is usually chaotic. Deterministic (irreversible) Markov chains can have very high mixing rates, which is the reason why we are interested in them. For a deterministic map we consider a transition kernel of the form $t(x' \mid x) = \delta(x' - f(x))$, where $f(x)$ is a continuous bijection. Invariance then looks like:

$$\int dx \delta(x' - f(x)) p(x) = p(x'). \tag{2}$$

This equation immediately implies the measure-preserving condition

$$p(x) = p(f(x)) \left| \frac{\partial f}{\partial x} \right| = p(f^{-1}(x)) \left| \frac{\partial f^{-1}}{\partial x} \right|, \tag{3}$$

where $\frac{\partial f}{\partial x}$ denotes the Jacobian of $f(x)$.

If we find a map $f(x)$ that satisfies equation (3) and that will reach any point in the support of $p(x)$ through repeated application, we obtain a proper sampler (an ergodic chain with stationary distribution $p(x)$). A practical example of such a sampler can be obtained analogously to (Murray & Elliott, 2012; Neal, 2012) using the CDF of the target distribution and its inverse:

$$f(x) = F_p^{-1}\Big((F_p(x) + C) \bmod 1 \Big), \tag{4}$$

where F_p denotes the CDF of the target density $p(x)$ and the constant C can be chosen as an irrational number to guarantee ergodicity of the map. To verify the correctness of this transition kernel one can straightforwardly put formula (4) into equation (3) or treat it as a special case of algorithm by (Murray & Elliott, 2012; Neal, 2012) (see Appendix A.3).

The equation (3) may be too restrictive, making the design of deterministic measure-preserving transformations (solutions of equation (3)) a very difficult task. To the best of our knowledge, only the algorithm proposed in (Murray & Elliott, 2012; Neal, 2012) provides practical examples of such transformations, relying on the knowledge of CDFs and their inverse.

In this paper, we propose a different transition kernel that leaves the target distribution invariant. That is,

$$\begin{aligned} t(x' \mid x) = \delta(x' - f(x)) \underbrace{\min\left\{1, \frac{p(f(x))}{p(x)} \left|\frac{\partial f}{\partial x}\right|\right\}}_{P_{\text{accept}}} + \\ + \delta(x' - x) \underbrace{\left(1 - \min\left\{1, \frac{p(f(x))}{p(x)} \left|\frac{\partial f}{\partial x}\right|\right\}\right)}_{P_{\text{reject}}}, \end{aligned} \tag{5}$$

Assume that p_0 is a delta-peak at some initial location x_0. Then the application of 5 will map this single delta peak to two delta peaks each with it's own weight: one peak at x_0 and the other at $f(x_0)$. At iteration t there are thus t weighted delta peaks, which becomes increasingly expensive to iterate forward. The more practical implementation of this kernel is to accept each new sample $x_{t+1} = f(x_t)$

with probability P_{accept} (see equation 5) or reject and keep the current point x_t with probability $(1 - P_{\text{accept}})$.

Putting this transition kernel into equation (1), we can simplify the fixed point equation to the condition that we formulate in the following proposition (see proof in Appendix A.1).

Proposition 1. *The fixed point equation* (1) *for the transition kernel* (5) *is equivalent to the equation*

$$\min\left\{p(f^{-1}(x))\left|\frac{\partial f^{-1}}{\partial x}\right|, p(x)\right\} = \min\left\{p(x), p(f(x))\left|\frac{\partial f}{\partial x}\right|\right\}. \tag{6}$$

A similar equation can be derived for the Barker's acceptance test (Barker, 1965) (see Appendix A.1).

Firstly, we note that measure-preserving transformations (solutions of (3)) are a special case of solutions of (6), with a zero rejection probability. Thus, these solutions eliminate all the stochasticity from the transition kernel (5), accepting all samples. However, equation (6) accepts a broader family of solutions that can be described by the equation

$$p(f(x))\left|\frac{\partial f}{\partial x}\right| = p(f^{-1}(x))\left|\frac{\partial f^{-1}}{\partial x}\right|. \tag{7}$$

The main difference with (3) is that here we do not restrict f to preserve the target density. Instead, we restrict $f(f(x))$ to preserve the target density:

$$p(x) = p(f(f(x)))\left|\frac{\partial f(f(x))}{\partial x}\right|. \tag{8}$$

The last equation can be obtained from equation (7) by considering the point $x = f(x')$. At first glance, the problem of finding an f such that $f(f(x))$ preserves the target measure is equally difficult to the problem of finding an $f(x)$ that preserves the density. However, the class of functions called involutions solves eq. 7 trivially because they satisfy $f(x) = f^{-1}(x)$. In such a case, $f(f(x)) = x$ indeed preserves the target measure by being an identity mapping. Thus, unlike equation (3), equation (6) is solved by involutive functions f. Unfortunately, it is not silver bullet: by inserting such f into the transition kernel, eq. (5) reduces our transition kernel to jump only between two points: from x to $f(x)$ and then to $f(f(x)) = f^{-1}(f(x)) = x$ again.

To be able to cover the support of the target distribution with involutive f, we introduce an additional source of stochasticity into (5). We do this through an *auxiliary variable*. That is, instead of traversing the target $p(x)$, we traverse the distribution $p(x, v) = p(x)p(v \mid x)$, where $p(v \mid x)$ is an auxiliary distribution that provides another degree of freedom in the design of the kernel. The key ingredients for choosing $p(v \mid x)$ are easy computation of its density and the ability to efficiently sample from it.

For the new target $p(x, v)$, we can apply the transition kernel (5) as well as formulate Proposition 1. This can be done by simply rewriting these equations by substituting the tuple $[x, v]$ for the variable x. Again, for the deterministic function $f(x, v)$, we resort to the family of involutive maps: $f(x, v) = f^{-1}(x, v)$. However, in contrast to the case without the auxiliary variables, now we have an opportunity to reach any point of the target support by resampling $v \mid x$ before applying the deterministic map $f(x, v)$. Interleaving the kernel (5) with the resampling of v one can collect samples from $p(x, v)$, and then obtain samples from the marginal distribution of interest $p(x)$ by simply ignoring v-coordinates of the collected samples. We provide the pseudo-code in Algorithm 1 below. To get an intuition, one can think of the resulting algorithm as of a slightly abstract version of Hybrid Monte Carlo (HMC) (Duane et al., 1987), where the momentum plays the role of the auxiliary variable v, and the Hamiltonian dynamics is a special case of the deterministic map f.

By construction, Algorithm 1 keeps the joint density $p(x, v)$ invariant (satisfies the fixed point equation), but does not provide any guarantees for ergodicity. In practice, the ergodicity is usually achieved by choosing proper involution and auxiliary distribution, such that the whole kernel is irreducible (Roberts et al., 2004).

Algorithm 1 Involutive MCMC

input target density $p(x)$
input density $p(v \mid x)$ and a sampler from $p(v \mid x)$
input involutive $f(x, v)$, i.e. $f(x, v) = f^{-1}(x, v)$
  initialize x
  for $i = 0 \dots n$ **do**
    sample $v \sim p(v \mid x)$
    propose $(x', v') = f(x, v)$
    $P = \min\{1, \frac{p(x', v')}{p(x, v)} \left|\frac{\partial f(x, v)}{\partial [x, v]}\right|\}$
    $x_i = \begin{cases} x', & \text{with probability } P \\ x, & \text{with probability } (1 - P) \end{cases}$
    $x \leftarrow x_i$
  end for
output samples $\{x_0, \dots, x_n\}$

Among the kernels that satisfy the fixed point equation there is a family of kernels called *reversible* which satisfy the detailed balance condition

$$t(x' \mid x)p(x) = t(x \mid x')p(x'). \tag{9}$$

Such kernels are known to mix slower compared to the kernels that satisfy the fixed point equation but are *irreversible* (do not satisfy the detailed balance condition) (Ichiki &

Ohzeki, 2013). In the following proposition, we demonstrate that the chain from Algorithm 1 is reversible on both the support of $p(x, v)$ and $p(x)$ (proof in Appendix A.2).

Proposition 2. *Transition kernel $t(x', v' \mid x, v)$ from Algorithm 1 satisfies detailed balance*

$$t(x', v' \mid x, v)p(x, v) = t(x, v \mid x', v')p(x', v'). \tag{10}$$

Moreover, the marginalized kernel on x

$$\widehat{t}(x' \mid x) = \int dv dv' t(x', v' \mid x, v)p(v \mid x) \tag{11}$$

also satisfies detailed balance

$$\widehat{t}(x' \mid x)p(x) = \widehat{t}(x \mid x')p(x'). \tag{12}$$

The reversibility of the chain $t(x', v' \mid x, v)$ is a direct consequence of the involutive property of the map $f(x, v)$ so it seems hard to avoid. However, it is still possible to construct an irreversible chain by composing several reversible kernels. We discuss this further in Section 3.3.

3. Tricks

The only two degrees of freedom possible to design in Involutive MCMC are the auxiliary distribution $p(v \mid x)$ and the involution $f(x, v)$. However, we will show that many existent MCMC algorithms from the literature can be formulated as Involutive MCMC by choosing suitable $f(x, v)$ and $p(v \mid x)$. As such, iMCMC represents a unifying framework for understanding existing and designing new MCMC algorithms.

We start by considering a simple involution $f(x, v) = [v, x]$ that is a swap of x and v. Choosing $q(v \mid x)$ such that the chain can reach any point in the support of $p(x)$, we end up with the Metropolis-Hastings algorithm (MH) with proposal $q(v \mid x)$. Indeed, the acceptance probability in the Algorithm 1 then equals

$$P = \min\left\{1, \frac{p(f(x, v))}{p(x)q(v \mid x)}\right\} = \min\left\{1, \frac{p(v)q(x \mid v)}{p(x)q(v \mid x)}\right\}.$$

While for MH the involution is very simple (a swap), we can also design MCMC algorithms by proposing sophisticated involutions. In the following subsections we explore this spectrum by demonstrating that a variety of MCMC algorithms can be formulated as Involutive MCMC methods. To avoid the large amounts of technical details of all the considered algorithms, we formulate the most important ideas as *tricks*. Besides being the main ideas of the algorithms, these tricks can serve as useful tools to design efficient novel samplers.

3.1. Smart auxiliary spaces

In this subsection we consider algorithms that focus on the development of advanced auxiliary distributions.

We start with the trick that allows one to circumvent the evaluation of intractable integrals in the target distribution or in the auxiliary distribution when we use Algorithm 1.

Trick 1 (Mixture distributions)**.** *Consider the joint distribution $p(x, v) = p(x)p(v \mid x)$, whose density is given as a mixture:*

$$p(x) = \int p(x \mid z)p(z)dz, \tag{13}$$

$$p(v \mid x) = \int q(v \mid a)q(a \mid x)da, \tag{14}$$

the evaluation of the integrals can be costly or even intractable. One can bypass the integration by sampling from the joint distribution $p(x, v, z, a) = p(x \mid z)p(z)q(v \mid a)q(a \mid x)$ using the Algorithm 1 with some involution $f(x, v, z, a)$. Note that to sample v we usually sample a at each step; hence, we may leave this intermediate variable a untouched by the subsequent involution, i.e. $f(x, v, z, a) = [f'(x, v, z), a]$. All arguments hold for discrete a and z as well. Moreover, conditioning the distribution $q(v \mid a)$ by the current state x: $q(v \mid a, x)$, allows one to obtain a so-called state-dependent mixture as a proposal:

$$p(v \mid x) = \int q(v \mid a, x)q(a \mid x)da. \tag{15}$$

In the discrete case

$$p(v \mid x) = \sum_j q(v \mid j, x)q(j \mid x). \tag{16}$$

This trick immediately implies the algorithm proposed in (Habib & Barber, 2018) (see Appendix B.2 for the proof), where the authors consider the proposal $q(x' \mid x)$ for the Metropolis-Hastings algorithm as a mixture $q(x' \mid x) = \int q(x' \mid a)q(a \mid x)da$. Another application of this trick can be found in the Multiple-Try Metropolis scheme (Liu et al., 2000) and Reversible-Jump MCMC (Green, 1995). We will return to these algorithms shortly.

Note, however, that avoiding by this trick the analytical integration, one only shifts the integration burden to the algorithm reducing its efficiency. Indeed, extending the target distribution with additional variables requires the sampling in higher dimensions, which may result in a slower convergence and a higher variance of the estimate.

The mixture of auxiliary variables in Trick 1 can be considered as an adaptive change of the family of proposal distributions depending on the current state of the chain. Another way to enrich the set of proposed points is to choose a suitable involution based on the current state. We describe this idea in the following trick.

Trick 2 (Mixture of Involutions)**.** *Consider the joint distribution $p(x, v) = p(x)p(v \mid x)$ and a parametric family of involutions $f_a(x, v) = f_a^{-1}(x, v)$, i.e., functions that define a proper involution in the space of tuples $[x, v]$ for a given a. It can be useful to apply different involutive maps depending on the current state $[x, v]$. For that purpose, one may introduce an auxiliary random variable a and define the joint distribution $p(x, v, a) = p(x, v)p(a \mid x, v)$. Then the involution in the new space is $f'(x, v, a) = [f_a(x, v), a]$, and the acceptance probability is*

$$P = \min\left\{1, \frac{p(f_a(x,v))p(a \mid f_a(x,v))}{p(x,v)p(a \mid x,v)}\left|\frac{\partial f_a(x,v)}{\partial [x,v]}\right|\right\}.$$

We thus observe that by first sampling $v \sim p(v \mid x)$ and $a \sim p(a \mid x, v)$, and then applying $f_a(x, v)$ we can have different involutions depending on x, v.

The crucial part of this trick is leaving the auxiliary variable a invariant by the involution in order to satisfy the fixed point equation. The correctness of such a kernel can be obtained by application of Proposition 1 for the tuple $[x, v, a]$ and the target distribution $p(x, v, a)$. Moreover, we immediately obtain the reversibility of this kernel from Proposition 2. For more details and formal derivations, we refer the reader to Appendix A.4.

Together with Trick 1, this trick provides an iMCMC formulation of the Multiple-Try Metropolis scheme (Liu et al., 2000). Speaking informally, we generate several proposals v, indexed by the variable a, using the mixture of distributions from Trick 1, and then stochastically decide which swap we use to propose the next state (see Appendix B.3 for the proof).

Surprisingly, using this trick we obtain the iMCMC formulation of Sample-Adaptive MCMC (Zhu, 2019) (see Appendix B.4), which does not have the MH acceptance test at all. Furthermore, Sample-Adaptive MCMC greatly relies on the aggregation functions that do not depend on the order of their arguments, i.e.

$$g(x_1, \dots, x_n) = g(\pi(x_1, \dots, x_n)), \tag{17}$$

where $\pi(\cdot)$ is an arbitrary permutation. Using the iMCMC formalism we can easily remove this restriction and obtain a more general scheme (see Appendix B.4.1).

Further, we will use Trick 2 for the reformulation of several algorithms: Reversible-Jump MCMC (Green, 1995), Non-Reversible Jump scheme (Gagnon & Doucet, 2019), and Look Ahead HMC (Sohl-Dickstein et al., 2014).

3.2. Smart deterministic maps

In this subsection we consider algorithms that introduce sophisticated involutive maps to obtain an efficient sampler. Historically, the first example is the Hybrid Monte Carlo (HMC) algorithm (Duane et al., 1987). Its core part is the Leap-Frog integrator of the corresponding Hamiltonian dynamics. We denote a single application of Leap-Frog as L and its iterative application as L^k, where k is the number of applications (steps of the dynamics). Then we can formulate HMC in terms of Involutive MCMC as follows. Consider the joint distribution $p(x, v) = p(x)p(v)$, where $p(v)$ usually equals to the standard normal and v represents the momentum variable. The involutive map can be constructed as the composition FL^k, where F denotes the momentum flip operator ($F(x, v) = [x, -v]$), and is applied after k iterative applications of L. According to the iMCMC formalism, the acceptance probability in Algorithm 1 is

$$P = \min\left\{1, \frac{p(FL^k(x,v))}{p(x,v)}\right\}. \tag{18}$$

Here we use the fact that both L and F preserves volume, hence, their Jacobians equal to 1. For the formal proof see Appendix B.6.

Contrary to the MH algorithm, for HMC we see that all the "knowledge about the target" of a sampler is concentrated in the involutive map. In contrast, the distribution $q(v|x)$ is very simple (a standard normal independent of x). This fact motivates the number of MCMC algorithms that try to build expressive deterministic maps using neural networks. We describe the main ideas of these algorithms in the following tricks.

Trick 3 (Auxiliary direction)**.** *Consider the joint distribution $p(x, v) = p(x)p(v \mid x)$, which we denote as $p(y) = p(x, v)$ with $y = [x, v]$. To obtain an expressive sampler, one can construct the required involution f using some non-involutive bijection $T(y)$ in the following way.*

Consider the joint distribution $p(y, d) = p(y)p(d \mid y)$, where the binary auxiliary variable $d = \{-1, +1\}$ encodes the direction in which we move from the current state. The involution f is then constructed as $f(y, d = +1) = [T(y), -1]$, $f(y, d = -1) = [T^{-1}(y), +1]$. The acceptance probability is

$$P = \min\left\{1, \frac{p(T_d(y))p(-d \mid T_d(y))}{p(y)p(d \mid y)}\left|\frac{\partial T_d}{\partial y}\right|\right\}, \tag{19}$$

where $T_{d=+1} = T$, and $T_{d=-1} = T^{-1}$.

More generally, we can choose d to lie in a vector space, parameterizing the family of bijections $T_d(y)$. To construct an involution we require that for any d there exists a unique d' such that $T_{d'} = T_d^{-1}$ and a smooth map $g(d) = d'$. Note that by requiring $T_{d'} = T_d^{-1}$ we immediately obtain $T_{d'}^{-1} = T_d$, hence $g(d') = d$ meaning that g is an involution. The final involution is then $f(y, d) = [T_d(y), g(d)]$, and the

acceptance probability is

$$P = \min\left\{1, \frac{p(T_d(y), g(d))}{p(y,d)}\left|\frac{\partial T_d}{\partial y}\right|\left|\frac{\partial g}{\partial d}\right|\right\}, \tag{20}$$

For instance, one can consider a Lie group on $\mathbb{R}^n$ with the group operation $T(y,d) = yd : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$. Then the involution f can be constructed as $f(y,d) = [yd, d^{-1}]$.

The described trick is a generalization of A-NICE-MC algorithm (Song et al., 2017) (see Appendix B.9), and L2HMC algorithm (Levy et al., 2017) (see Appendix B.10). Indeed, considering the uniform distribution over the binary auxiliary variable $p(d \mid x, v) = p(d) = \text{Uniform}\{-1, +1\}$, and taking the bijection $T(x, v)$ as the corresponding model, we immediately obtain both algorithms. Furthermore, taking the same distribution for the auxiliary direction $p(d) = \text{Uniform}\{-1, +1\}$ and combining Tricks 1, 2, 3 we formulate Reversible-Jump MCMC algorithm (Green, 1995) in terms of iMCMC (see Appendix B.5).

Note that also vanilla HMC can easily be expressed using this trick by using $T(y) = L(y)$ and $T^{-1}(y) = L^{-1}(y) = FLF(y)$ and where $d \sim \text{Uniform}\{-1, +1\}$. Indeed, the flips $F(y = [x, v]) = [x, -v]$ in the Leap-Frog procedure do not influence the chain when $p(x, v) = p(x, -v)$. We will return to this formulation of HMC during the discussion of irreversible chains.

In the following trick, we demonstrate another way to use a bijection T to construct an expressive involution that generalizes the NeuTra algorithm of (Hoffman et al., 2019) (see Appendix B.8).

Trick 4 (Embedded involution)**.** *Consider the iMCMC sampler with the joint distribution $p(x, v) = p(x)p(v \mid x)$ and the involution $f(x, v) = f^{-1}(x, v)$. Assume that for some reason the involution f is not expressive enough to yield an efficient sampler for the target distribution $p(x)$. We can enrich the sampler by choosing a suitable bijection T, and introducing the new involution $f_T = T^{-1} \circ f \circ T$, where $\circ$ is the composition operation.*

Moreover, consider the embedded random variable $[X_T, V_T] = T(X, V)$, $[X, V] \sim p(x, v)$ with the density

$$p_T(x_T, v_T) = p(T^{-1}(x_T, v_T))\left|\frac{\partial T^{-1}}{\partial [x_T, v_T]}\right|. \tag{21}$$

Then the Algorithm 1 with the joint distribution $p(x, v)$ and the involution f_T is equivalent to the following procedure. Given the sample $[x, v] \sim p(x, v)$, map this sample as $T(x, v)$. Starting from $T(x, v)$, collect new samples $\{(x_T, v_T)_i\}$ from $p_T(x_T, v_T)$ using the Algorithm 1 with the joint distribution $p_T(x_T, v_T)$ and the involution $f(x_T, v_T)$. Then map all the collected samples as $T^{-1}(x_T, v_T)$.

The map T can be viewed as a 'flow' model to a simpler 'disentangled' or more symmetric latent space x_T, v_T where algorithms such as HMC are easier to run. The map T could be learned using unsupervised learning on already generated samples.

3.3. Smart compositions

In this subsection, we apply the formalism of involutive MCMC to describe irreversible chains, i.e. chains that do not satisfy detailed balance. Recall that in Section 2 we have shown that any iMCMC chain must be reversible. However, a composition of reversible chains is not necessarily reversible. Thus, in the following tricks, we use reversible iMCMC chains as building blocks to construct a composition that is irreversible. A representative example of such a composition is Gibbs sampling (Geman & Geman, 1984). Indeed, the update of a single coordinate in the Gibbs algorithm is a reversible kernel easily described by the iMCMC framework, while the composition of these kernels yields Gibbs sampling which is irreversible (see Appendix B.12).

Using a composition of kernels, we can make an irreversible analogue of Trick 3. The main difference is that in this Trick we do not resample the auxiliary (directional) variable d at every iteration. This would reverse the direction after each accepted proposal. However, by composing this with a kernel that simply flips d again we get a persistent (irreversible) kernel that only flips directions when a sample is rejected.

Trick 5 (Persistent direction)**.** *Given the target distribution $p(y) = p(x, v)$, we consider the joint distribution $p(y, d) = p(y)p(d)$, where $p(d) = \text{Uniform}\{-1, +1\}$, and the variable $d = \{-1, +1\}$ encodes the direction in which we move from the current state. Following Trick 3, we consider some non-involutive bijection $T(y)$ and the corresponding involution $f_1(y, d = +1) = [T(y), -1]$, $f_1(y, d = -1) = [T^{-1}(y), +1]$. Thus, we obtain the iMCMC kernel $t_1(y', d' \mid y, d)$ that accepts the proposal point $[T_d(y), -d]$ with the probability*

$$P_1 = \min\left\{1, \frac{p(T_d(y))p(-d)}{p(y)p(d)}\left|\frac{\partial T_d}{\partial y}\right|\right\} = \tag{22}$$

$$= \min\left\{1, \frac{p(T_d(y))}{p(y)}\left|\frac{\partial T_d}{\partial y}\right|\right\}, \tag{23}$$

where $T_{d=+1} = T$, and $T_{d=-1} = T^{-1}$. Then we compose the kernel t_1 with the kernel t_2 that just flips the directional variable. In terms of iMCMC, the target distribution is $p(y, d)$ and the involution is $f_2(y, d) = [y, -d]$. Note that this proposal will be always accepted since $p(y, -d) = p(y, d)$. Then the composition of t_1 and t_2 works as follows.

$$\textit{current state} = [y, d] \tag{24}$$

$$\textit{next state} = \begin{cases} [T_d(y), d], \textit{ with probability } P_1 \\ [y, -d], \textit{ with probability } (1 - P_1) \end{cases} \tag{25}$$

The same logic can be applied to the variable v. Since $T_d(y) = T_d(x, v)$ may significantly depend on the variable v, instead of resampling it at each step, one can use another kernel to update it conditioned on its previous value.

The intuition of this composition is as follows. In the case of an accept we now try to move further by applying the same T_d instead of the inverse map T_{-d}, whereas, in the case of a reject, we flip the variable d and move in the opposite direction. We depict this intuition in Fig. 1.

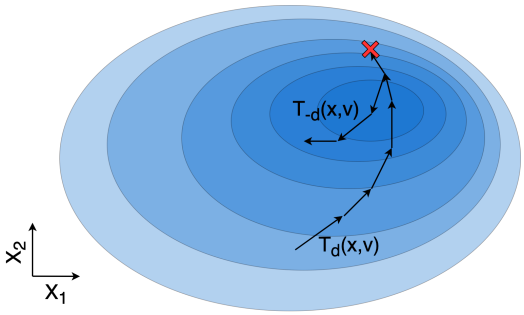

Figure 1: Schematic representation of Trick 5. The deterministic map $T_d(y) = T_d(x, v)$ is iteratively applied with resampling of the variable v, moving in the regions of a high density. When the chain tries to move to a lower density region, the proposal may be rejected (red cross), which triggers the chain to change direction and apply $T_{-d}(x, v)$.

Trick 5 describes the main idea behind persistent chains leading to irreversibility. Its generalizations can be derived similar to what we explained in Trick 3, by considering a conditional direction $p(d \,|\, y)$ or a vector valued direction variable d. If we use a distribution $p(d \,|\, y)$, we must also take care to change the second kernel t_2 to preserve the target $p(y, d)$.

The analogue of the direction flip in Trick 5 may be found in the HMC algorithm with persistent momentum (Horowitz, 1991) (see Appendix B.11) and the Look Ahead HMC algorithm (Sohl-Dickstein et al., 2014) (see Appendix B.13). These algorithms use post-acceptance negation of the momentum variable v relying on the symmetry of the auxiliary distribution: $p(-v) = p(v)$. However, using Trick 5 we can easily generalize these algorithms to the case of an asymmetric auxiliary distribution $p(-v) \neq p(v)$ by considering the forward map as a Leap-Frog operator $T_{d=+1}(y) = L(y)$ and its inverse as $T_{d=-1}(y) = L^{-1}(y)$, where L^{-1} is the Leap-Frog backward in time. More details are provided in Appendix B.11.

In light of Tricks 2 and 5, we can obtain yet another generalization of Look Ahead HMC (Sohl-Dickstein et al., 2014). In this paper, the authors propose the mixture of involutions $f_k(x, v) = FL^k(x, v)$, where we choose k stochastically based on the current state. Using Trick 5, we can look ahead of any function we want, by considering the family of involutions

$$f_k(x, v, d = +1) = [T^k(x, v), -1], \tag{26}$$

$$f_k(x, v, d = -1) = [T^{-k}(x, v), +1], \tag{27}$$

where T^k means k iterative applications of the map T, and T^{-k} means the same but for the map T^{-1}. Note that by considering $T = L$ and $T^{-1} = FLF$, and the symmetric auxiliary distribution $p(v) = p(-v)$ we obtain Look Ahead HMC.

The combination of Tricks 2, 5 provides a neat iMCMC formulation of Gibbs sampling (Geman & Geman, 1984) and Non-Reversible Jump MCMC (Gagnon & Doucet, 2019). Details can be found in appendices B.12 and B.14 respectively.

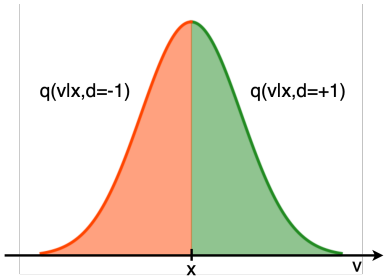

Figure 2: Schematic representation of auxiliary distribution in Trick 6. Red mass represents $q(v \,|\, x, d = -1)$, green mass represents $q(v \,|\, x, d = +1)$. These proposals do not intersect only for illustrative purposes to highlight that we cannot move to the left if $d = +1$.

Trick 5 tells us how to design an irreversible chain using the deterministic part of the iMCMC framework. The following trick makes it possible using the stochastic auxiliary variables.

Trick 6 (Persistent proposal)**.** *Consider the joint target distribution $p(x, d)$, where $p(d) = Uniform\{-1, +1\}$ is the directional variable. Choose the auxiliary distribution $q(v \,|\, x, d)$ that proposes new points depending on the current direction d. For instance, this can be done by splitting a random walk proposal $q(v \,|\, x)$ as depicted in Fig. 2. Then the constructed iMCMC kernel $t_1(x', d' \,|\, x, d)$ with involution $f(x, v, d) = [v, x, -d]$ has the probability of acceptance:*

$$P_1 = \min\left\{1, \frac{p(v)q(x \,|\, v, -d)}{p(x)q(v \,|\, x, d)}\right\}. \tag{28}$$

Note that having proposals as depicted in Fig. 2 we must change the directional variable when proposing the next state, otherwise we obtain zero probability of acceptance.

As well as in Trick 5, we then compose the kernel t_1 with the kernel t_2 that just flips the directional variable. In terms

Table 2: Performance of the algorithms as measured by the batch-means estimator of Effective Sample Size (ESS) averaged across 100 independent chains. Higher values of ESS and ESS per second are better (for detailed formulation see Appendix C.2). For computational efforts we provide ESS per second taking into account the sampling time for 20000 samples. See description of the compared methods in the text.

	ESS				**ESS per second**			
Algorithm	MoG2	Australian	German	Heart	Mog2	Australian	German	Heart
MALA	0.007 ± 0.002	$\mathbf{0.043 \pm 0.001}$	$\mathbf{0.025 \pm 0.005}$	$\mathbf{0.081 \pm 0.012}$	2	5	3	9
Irr-MALA	$\mathbf{0.027 \pm 0.008}$	0.006 ± 0.001	0.004 ± 0.001	0.012 ± 0.001	4	1	1	1
A-NICE-MC	0.852 ± 0.239	0.137 ± 0.026	0.032 ± 0.004	0.253 ± 0.033	1700	94	17	241
Irr-NICE-MC	$\mathbf{1.643 \pm 0.626}$	$\mathbf{0.177 \pm 0.030}$	0.032 ± 0.004	$\mathbf{0.341 \pm 0.051}$	**3280**	**121**	17	**324**

of iMCMC that is, the target distribution is $p(x, d)$ and the involution is $f_2(x, d) = [x, -d]$. Note that this proposal will be always accepted since $p(x, -d) = p(x, d)$. Then the composition of t_1 and t_2 works as follows.

$$\textit{current state} = [x, d] \tag{29}$$

$$\textit{proposal} = v \sim q(v \mid x, d) \tag{30}$$

$$\textit{next state} = \begin{cases} [v, d], \ \textit{with probability } P_1 \\ [x, -d], \ \textit{with probability } (1 - P_1) \end{cases} \tag{31}$$

Once again, a more general version of this trick can be obtained as in Trick 3, by considering a conditional direction $p(d \mid x)$ or a direction vector-valued d. If we change the distribution to $p(d \mid x)$, we must also change the second kernel t_2 to preserve the target $p(x, d)$.

Implicitly this trick is used in the Lifted Metropolis-Hastings algorithm (Turitsyn et al., 2011), which gives a rise to many irreversible algorithms. The only difference with Trick 6 is that Lifted MH design the proposal distribution $q(v \mid x, d)$ as the transition kernel of the conventional MH algorithm (see Appendix B.15).

Note that taking the kernels t_+ and t_- that already satisfy the generalized detailed balance $t_+(x' \mid x)p(x) = t_-(x \mid x')p(x')$ as positive and negative parts of $q(x \mid v, d)$, we obtain the irreversible chain that is equivalent to the application either of t_+ or t_-.

4. Examples

We now proceed with illustrating that the proposed framework provides an easy paradigm to extend and combine existing methods and potentially improve them. Below, we propose simple extensions, to make MALA and A-NICE-MC irreversible. We empirically validate these examples on a set of tasks, which includes a mixture of two 2d-Gaussians (MoG2) and the posterior distribution of Bayesian logistic regression on several datasets (Australian, German, Heart) (see Appendix C.1 for details). For performance evaluation, we use the effective sample size (ESS), which measures how many independent samples the chain actually contains. To be more precise, we use batch-means estimator of ESS, which is shown to be more robust (Thompson, 2010) (see Appendix C.2 for details).

We start with the Metropolis-Adjusted Langevin algorithm (MALA) (Roberts et al., 1996), which generates proposals by following the gradient of the target log-probability. We modify the original algorithm with a directional variable $p(d) = \text{Uniform}\{-1, +1\}$ as follows. The joint distribution is now:

$$p(x, v, d) = p(x)\mathcal{N}(v \mid x + d\varepsilon\nabla_x \log p(x), 2\varepsilon)p(d),$$

and the involutive map is

$$f(x, v, d) = [v, x, -d \cdot \text{sign}(\nabla_x \log p(x)^T \nabla_v \log p(v))].$$

Thus, our modification (Irr-MALA) ensures that the gradient in the proposed point will be directed towards the initial point. The irreversible chain can be obtained by the application of the described kernel followed by the negation of d (see Appendix C.3 for pseudo-code). However, allowing the chain to traverse along the gradient of decreasing probability reduces its acceptance rate, which leads to a poor performance on the unimodal posteriors of Bayesian logistic regression. However, if we need to traverse low probability regions between two modes of a distribution this idea becomes beneficial and leads to improved performance. We can see this when we sample from the bimodal MoG2 distribution. (see Table 2).

We now turn to a more complex model, and design the irreversible version of A-NICE-MC (Song et al., 2017), which learns the NICE model (Dinh et al., 2014) to obtain an expressive proposal. Our modification (Irr-NICE-MC) is a straightforward application of Trick 5 to the original algorithm (see Appendix C.4 for pseudo-code). The only difference with Trick 5 is that we add one more kernel that conditionally updates the auxiliary variable v as

$$v' = v\sqrt{1 - \alpha^2} + \alpha \cdot \eta, \ \ \eta \sim \mathcal{N}(0, 1). \tag{32}$$

For all targets we choose $\alpha = 0.8$. To provide a robust comparison, we do not change the training process of the original algorithm. Moreover, we compare our modification against the original method, using the same–already learned–model as the proposal distribution. In Table 2, we see that simply introducing irreversibility into the kernel may result in significant performance gains while having a negligible computational overhead.

Code for reproducing the experiments is available at `https://github.com/necludov/iMCMC`. We also refer the reader to the Gen probabilistic language (Cusumano-Towner et al., 2019), where the Involutive MCMC is proposed as a software design pattern.

5. Related Work

Several approaches unifying MCMC exist in the literature. They focus on the kernels with multiple proposals and describe them, extending the state space through the auxiliary variables (Tjelmeland, 2004; Storvik, 2011). The most general unifying framework is given in (Finke, 2015), which considers all Monte Carlo algorithms as the importance sampling on differently extended spaces.

The key difference of the proposed framework is the explicit usage of the involutive deterministic map inside of the generic kernel. Although this deterministic part appears to be trivial in some cases (for instance, the swap in the MH algorithm), it may serve as a design principle for many MCMC algorithms. The main benefit of this principle comes when we consider hybrid algorithms that incorporate expressive deterministic maps into MCMC kernels. Such hybrid algorithms demonstrate promising results in modern physics (Kanwar et al., 2020), and the iMCMC framework may give a hint on how to design these algorithms as well as how to combine features of different MCMC kernels.

Besides providing theoretical insights, Involutive MCMC may serve as a design principle in silico (see Gen language (Cusumano-Towner et al., 2019)). The authors of Gen thoroughly discuss this opportunity (Cusumano-Towner et al., 2020) by developing the algorithm that automates iMCMC and exploits sparsity to increase efficiency.

6. Conclusion

In this paper, we have proposed a unifying view of a large class of MCMC algorithms. This was achieved by reformulating MCMC as the composition of sampling from an auxiliary distribution and an involutive deterministic map, followed by a MH accept-reject step. This was shown to represent a very large family of reversible MCMC algorithms. We then extend this class further with the use of auxiliary variables into irreversible MCMC algorithms. Through a number of "Tricks" we facilitate the process of extending existing algorithms, which we illustrate through some simple examples.

We believe our unifying view of MCMC algorithms will lead to a number of generalizations in the future. For instance, some of the versions look very similar to flow-based models used for unsupervised learning and some of our proposed kernels can indeed be used for such a purpose. We also believe there are interesting connections between deterministic samplers as in (Murray & Elliott, 2012; Neal, 2012) and the theory of (chaotic) iterated maps and nonlinear dynamical systems.

7. Acknowledgments

The authors are thankful to the reviewers who provided detailed feedback and pointed out essential related works. Kirill Neklyudov and Dmitry Vetrov have been supported by the Russian Science Foundation grant no. 19-71-30020.

References

Barker, A. A. Monte carlo calculations of the radial distribution functions for a proton? electron plasma. *Australian Journal of Physics*, 18(2):119–134, 1965.

Besag, J. Comments on "representations of knowledge in complex systems" by u. grenander and mi miller. *J. Roy. Statist. Soc. Ser. B*, 56:591–592, 1994.

Bierkens, J., Roberts, G., et al. A piecewise deterministic scaling limit of lifted metropolis–hastings in the curie–weiss model. *The Annals of Applied Probability*, 27(2): 846–882, 2017.

Cusumano-Towner, M., Lew, A. K., and Mansinghka, V. K. Automating involutive mcmc using probabilistic and differentiable programming. *arXiv preprint arXiv:2007.09871*, 2020.

Cusumano-Towner, M. F., Saad, F. A., Lew, A. K., and Mansinghka, V. K. Gen: a general-purpose probabilistic programming system with programmable inference. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 221–236, 2019.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

Finke, A. *On extended state-space constructions for Monte Carlo methods*. PhD thesis, University of Warwick, 2015.

Gagnon, P. and Doucet, A. Non-reversible jump algorithms for bayesian nested model selection. *arXiv preprint arXiv:1911.01340*, 2019.

Geman, S. and Geman, D. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741, 1984.

Geyer, C. J. Practical markov chain monte carlo. *Statistical science*, pp. 473–483, 1992.

Geyer, C. J. The metropolis-hastings-green algorithm, 2003.

Girolami, M. and Calderhead, B. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011.

Green, P. J. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Green, P. J. and Hastie, D. I. Reversible jump mcmc. *Genetics*, 155(3):1391–1403, 2009.

Habib, R. and Barber, D. Auxiliary variational mcmc. 2018.

Hastings, W. K. Monte carlo sampling methods using markov chains and their applications. 1970.

Hoffman, M., Sountsov, P., Dillon, J. V., Langmore, I., Tran, D., and Vasudevan, S. Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

Horowitz, A. M. A generalized guided monte carlo algorithm. *Physics Letters B*, 268(2):247–252, 1991.

Ichiki, A. and Ohzeki, M. Violation of detailed balance accelerates relaxation. *Physical Review E*, 88(2):020101, 2013.

Kanwar, G., Albergo, M. S., Boyda, D., Cranmer, K., Hackett, D. C., Racanière, S., Rezende, D. J., and Shanahan, P. E. Equivariant flow-based sampling for lattice gauge theory. *arXiv preprint arXiv:2003.06413*, 2020.

Levy, D., Hoffman, M. D., and Sohl-Dickstein, J. Generalizing hamiltonian monte carlo with neural networks. *arXiv preprint arXiv:1711.09268*, 2017.

Liu, J. S., Liang, F., and Wong, W. H. The multiple-try method and local optimization in metropolis sampling. *Journal of the American Statistical Association*, 95(449): 121–134, 2000.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Murray, I. and Elliott, L. T. Driving markov chain monte carlo with a dependent random stream. *arXiv preprint arXiv:1204.3187*, 2012.

Neal, R. M. How to view an mcmc simulation as a permutation, with applications to parallel simulation and improved importance sampling. *arXiv preprint arXiv:1205.0070*, 2012.

Roberts, G. O. and Rosenthal, J. S. Optimal scaling of discrete approximations to langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268, 1998.

Roberts, G. O., Tweedie, R. L., et al. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

Roberts, G. O., Rosenthal, J. S., et al. General state space markov chains and mcmc algorithms. *Probability surveys*, 1:20–71, 2004.

Sohl-Dickstein, J., Mudigonda, M., and DeWeese, M. R. Hamiltonian monte carlo without detailed balance. *arXiv preprint arXiv:1409.5191*, 2014.

Song, J., Zhao, S., and Ermon, S. A-nice-mc: Adversarial training for mcmc. In *Advances in Neural Information Processing Systems*, pp. 5140–5150, 2017.

Storvik, G. On the flexibility of metropolis–hastings acceptance probabilities in auxiliary variable proposal generation. *Scandinavian Journal of Statistics*, 38(2):342–358, 2011.

Thompson, M. B. A comparison of methods for computing autocorrelation time. *arXiv preprint arXiv:1011.0175*, 2010.

Tjelmeland, H. Using all metropolis–hastings proposals to estimate mean values. Technical report, 2004.

Turitsyn, K. S., Chertkov, M., and Vucelja, M. Irreversible monte carlo algorithms for efficient sampling. *Physica D: Nonlinear Phenomena*, 240(4-5):410–414, 2011.

Zhu, M. Sample adaptive mcmc. In *Advances in Neural Information Processing Systems*, pp. 9063–9074, 2019.