Asymptotically exact inference in differentiable generative models

Matthew M. Graham
University of Edinburgh

Amos J. Storkey
University of Edinburgh

Abstract

Many generative models can be expressed as a differentiable function of random inputs drawn from some simple probability density. This framework includes both deep generative architectures such as Variational Autoencoders and a large class of procedurally defined simulator models. We present a method for performing efficient MCMC inference in such models when conditioning on observations of the model output. For some models this offers an asymptotically exact inference method where Approximate Bayesian Computation might otherwise be employed. We use the intuition that inference corresponds to integrating a density across the manifold corresponding to the set of inputs consistent with the observed outputs. This motivates the use of a constrained variant of Hamiltonian Monte Carlo which leverages the smooth geometry of the manifold to coherently move between inputs exactly consistent with observations. We validate the method by performing inference tasks in a diverse set of models.

1 Introduction

Developments in generative modelling with deep architectures such as *Variational Auto-Encoders* (VAEs) [15, 30] and *Generative Adversarial Nets* (GANs) [12] have made it possible to learn probabilistic models of increasingly complex, high-dimensional data-sets. The generator of these models can be formulated as a differentiable[1] function $\boldsymbol{G} : \mathcal{U} \rightarrow \mathcal{X}$ which maps random vector inputs $\mathbf{u} \in \mathcal{U} = \mathbb{R}^M$ drawn from a probability distribution with known density $\mathbb{p}_{\mathbf{u}}[\boldsymbol{u}] = \rho(\boldsymbol{u})$ to an implicitly defined distribution over random vector outputs $\mathbf{x} = \boldsymbol{G}(\mathbf{u}) \in \mathcal{X} = \mathbb{R}^N$. We will refer here to a model of this form as a *differentiable generative model* (DGM).

[1] Differentiable here means that the Jacobian $\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}}$ is defined a.e.

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

As well as parametric models learnt from data, the DGM framework also encapsulates a wide class of simulator models where $\boldsymbol{G}$ is defined procedurally, e.g. numerical integration of a system of stochastic differential equations. Here the random inputs $\mathbf{u}$ are the series of draws from a random number generator during the simulator's execution. The operations used in many simulations are differentiable and automatic differentiation provides a computationally efficient framework for calculating the exact derivatives of a differentiable simulator's outputs with respect to its random inputs given just the code used to define the model.

Often we will be interested in using a generative model to make inferences about the modelled variables given observations related to outputs of the model. For example given a generative model of images we may wish to infer plausible in-paintings of an image region given knowledge of the surrounding pixel values. Similarly given a simulator of a physical process and generator of parameters of the process which we believe are reasonable a priori, we may wish to infer our posterior beliefs about the parameters under the model given observations of the physical process.

Concretely, we consider the generated output $\mathbf{x}$ as being partitioned into observed variables $\mathbf{y} \in \mathcal{Y} = \mathbb{R}^{N_y}$ and latent variables $\mathbf{z} \in \mathcal{Z} = \mathbb{R}^{N_z}$ to be inferred, i.e. $\mathbf{x} = [\mathbf{y}; \mathbf{z}]$ and $\mathcal{X} = \mathcal{Y} \times \mathcal{Z}$. Likewise we partition the generator such that $\mathbf{y} = \boldsymbol{G}_y(\mathbf{u})$ and $\mathbf{z} = \boldsymbol{G}_z(\mathbf{u})$. Inference is then the task of computing expectations of $\mathbf{z}$ conditioned on an observed $\mathbf{y}$.

A common special case is where $\mathbf{z}$ is generated from a subset of the random inputs $\mathbf{u}_z$, and $\mathbf{y}$ is then generated as a function of $\mathbf{z}$ and the remaining random inputs $\mathbf{u}_y$. For example when $\mathbf{z}$ is sampled from a *prior* and $\mathbf{y}$ sampled from a *likelihood* given $\mathbf{z}$. We will refer to this specialism as a *directed model* in reference to the fact that it has a natural interpretation as a directed graphical model as illustrated in figure 1.

For many DGMs we cannot explicitly compute the joint density on the generator outputs $\mathbb{p}_{\mathbf{x}}[\boldsymbol{x}] = \mathbb{p}_{\mathbf{y},\mathbf{z}}[\boldsymbol{y}, \boldsymbol{z}]$ and so conditional density $\mathbb{p}_{\mathbf{z}\,|\,\mathbf{y}}[\boldsymbol{z} \mid \boldsymbol{y}]$ we wish to compute expectations with respect to. In directed models this usually corresponds to not being able to evaluate the likelihood $\mathbb{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y} \mid \boldsymbol{z}]$. The lack of a closed form density impedes the direct use of approximate inference methods such as variational inference and *Markov chain Monte Carlo* (MCMC).

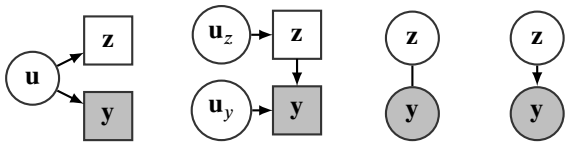


Figure 1: (a) and (b) *Stochastic computation graphs* (SGC) [33] visualising models considered in this paper. Circular nodes represent stochastic variables sampled from a conditional distribution on their parent nodes or marginal for root nodes. Square nodes represent variables that are deterministic functions of their parents. Shaded nodes are observed. (a) SGC of general case in which $\mathbf{y}$ and $\mathbf{z}$ are jointly generated from $\mathbf{u}$. (b) SGC of the directed case in which we first generate $\mathbf{z}$ from $\mathbf{u}_z$ then generate $\mathbf{y}$ from $\mathbf{z}$ and $\mathbf{u}_y$. (c) Undirected graphical model corresponding to (a) when integrating out $\mathbf{u}$. (d) Directed graphical model which is a natural representation of (b) when integrating out $\mathbf{u}_y$ and $\mathbf{u}_z$.

2 Approximate Bayesian Computation

A lack of explicit likelihoods is the motivation for *likelihood-free* inference methods such as *Approximate Bayesian Computation* (ABC) [4, 20]. In ABC the simulated observed outputs $\mathbf{y}$ are decoupled from the observed data $\bar{\mathbf{y}}$ by a noise model or *kernel* $\mathrm{p}_{\bar{\mathbf{y}}\,|\,\mathbf{y}}[\bar{\boldsymbol{y}}\,|\,\boldsymbol{y}] = k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})$ with tolerance parameter ϵ, e.g. $k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y}) \propto \mathbb{I}\left[|\bar{\boldsymbol{y}} - \boldsymbol{y}| < \epsilon\right]/\epsilon^{N_y}$ (uniform ball kernel) or $k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y}) = \mathcal{N}\left(\bar{\boldsymbol{y}}\,|\,\boldsymbol{y},\,\epsilon^2\boldsymbol{I}\right)$ (Gaussian kernel). The *ABC likelihood* is then defined as

$$\mathrm{p}_{\bar{\mathbf{y}}\,|\,\mathbf{z}}[\bar{\boldsymbol{y}}\,|\,\boldsymbol{z};\epsilon] = \int_{\mathcal{Y}} k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}. \tag{1}$$

The *ABC posterior* is likewise defined as

$$\mathrm{p}_{\mathbf{z}\,|\,\bar{\mathbf{y}}}[\boldsymbol{z}\,|\,\bar{\boldsymbol{y}};\,\epsilon] = \frac{\int_{\mathcal{Y}} k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}}{\int_{\mathcal{Y}} k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}}[\boldsymbol{y}]\,\mathrm{d}\boldsymbol{y}} \tag{2}$$

and we can express expectations of functions of the latent variables $\mathbf{z}$ with respect to this approximate posterior

$$\begin{aligned}\mathbb{E}\left[f(\mathbf{z})\,|\,\bar{\mathbf{y}} = \bar{\boldsymbol{y}};\,\epsilon\right] &= \int_{\mathcal{Z}} f(\boldsymbol{z})\,\mathrm{p}_{\mathbf{z}\,|\,\bar{\mathbf{y}}}[\boldsymbol{z}\,|\,\bar{\boldsymbol{y}};\,\epsilon]\,\mathrm{d}\boldsymbol{z} \qquad (3)\\ &= \frac{\int_{\mathcal{Z}}\int_{\mathcal{Y}} f(\boldsymbol{z})\,k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\boldsymbol{z}}{\int_{\mathcal{Z}}\int_{\mathcal{Y}} k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\boldsymbol{z}}.\end{aligned}$$

Various Monte Carlo inference schemes can be used to estimate this expectation. The simplest is to generate a set of independent samples $\left\{\boldsymbol{y}^{(s)},\,\boldsymbol{z}^{(s)}\right\}_{s=1}^{S}$ from $\mathrm{p}_{\mathbf{y},\mathbf{z}}[\boldsymbol{y},\,\boldsymbol{z}]$ [2] and use an importance sampling estimator

$$\mathbb{E}\left[f(\mathbf{z})\,|\,\bar{\mathbf{y}} = \bar{\boldsymbol{y}};\,\epsilon\right] \approx \frac{\sum_{s=1}^{S}\left\{f\left(\boldsymbol{z}^{(s)}\right)\,k_\epsilon\left(\bar{\boldsymbol{y}};\,\boldsymbol{y}^{(s)}\right)\right\}}{\sum_{s=1}^{S}\left\{k_\epsilon\left(\bar{\boldsymbol{y}};\,\boldsymbol{y}^{(s)}\right)\right\}}. \tag{4}$$

[2] ABC is usually applied to directed models hence this is usually considered as generating $\mathbf{z}$ from a prior then simulating $\mathbf{y}$ given $\mathbf{z}$ however more generally we can just sample from the joint.

In the case of a uniform ball kernel this corresponds to the standard ABC reject algorithm, with expectations being estimated as averages over the latent variable samples for which the corresponding simulated outputs are within a (Euclidean) distance of ϵ from the observations.

Alternatively a MCMC scheme can be used to estimate the ABC posterior expectation in (3) [21]. A Markov chain is constructed with stationary distribution

$$\mathrm{p}_{\mathbf{y},\mathbf{z}\,|\,\bar{\mathbf{y}}}[\boldsymbol{y},\,\boldsymbol{z}\,|\,\bar{\boldsymbol{y}}] \propto k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}] \tag{5}$$

by proposing a new state $(\boldsymbol{y}',\,\boldsymbol{z}')$ given the current state $(\boldsymbol{y},\,\boldsymbol{z})$ by sampling $\boldsymbol{z}' \sim q(\cdot\,|\,\boldsymbol{z})$ from some perturbative proposal distribution q on $\mathcal{Z}$ and then generating a new $\boldsymbol{y}' \sim \mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\cdot\,|\,\boldsymbol{z}]$. This is then accepted with probability

$$a(\boldsymbol{y}',\boldsymbol{z}'\,|\,\boldsymbol{y},\boldsymbol{z}) = \min\left[1,\,\frac{k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y}')\,q(\boldsymbol{z}\,|\,\boldsymbol{z}')\,\mathrm{p}_{\mathbf{z}}\left[\boldsymbol{z}'\right]}{k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})\,q(\boldsymbol{z}'\,|\,\boldsymbol{z})\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]}\right], \tag{6}$$

the overall transition operator leaving (5) stationary. The samples of the chain state can then be used to compute consistent estimators of (3). This scheme relies on being able to evaluate $\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]$ and so is only applicable to directed models where the generator $\boldsymbol{G}_z$ is of a tractable form such that $\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]$ is known.

In the limit of $\epsilon \to 0$ valid ABC kernels $k_\epsilon(\bar{\boldsymbol{y}};\,\boldsymbol{y})$ collapse to a Dirac delta distribution $\delta\left(\bar{\boldsymbol{y}} - \boldsymbol{y}\right)$ and so the ABC posterior in (2) converges to the posterior $\mathrm{p}_{\mathbf{z}\,|\,\mathbf{y}}[\boldsymbol{z}\,|\,\bar{\boldsymbol{y}}]$. Similarly for the ABC posterior expectation in (3) we have

$$\begin{aligned}&\lim_{\epsilon\to 0}\left\{\mathbb{E}\left[f(\mathbf{z})\,|\,\bar{\mathbf{y}} = \bar{\boldsymbol{y}};\,\epsilon\right]\right\}\\ &= \frac{\int_{\mathcal{Z}}\int_{\mathcal{Y}} f(\boldsymbol{z})\,\delta\left(\bar{\boldsymbol{y}} - \boldsymbol{y}\right)\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\boldsymbol{z}}{\int_{\mathcal{Z}}\int_{\mathcal{Y}} \delta\left(\bar{\boldsymbol{y}} - \boldsymbol{y}\right)\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{y}\,\mathrm{d}\boldsymbol{z}} \qquad (7)\\ &= \frac{\int_{\mathcal{Z}} f(\boldsymbol{z})\,\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\bar{\boldsymbol{y}}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{z}}{\int_{\mathcal{Z}} \mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\bar{\boldsymbol{y}}\,|\,\boldsymbol{z}]\,\mathrm{p}_{\mathbf{z}}[\boldsymbol{z}]\,\mathrm{d}\boldsymbol{z}} = \mathbb{E}\left[f(\mathbf{z})\,|\,\mathbf{y} = \bar{\boldsymbol{y}}\right].\end{aligned}$$

For both the ABC reject and ABC MCMC schemes just described, simulated observations $\boldsymbol{y}$ are independently generated from the conditional $\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]$ given the current latent variables $\boldsymbol{z}$. The observed values $\bar{\boldsymbol{y}}$ are a zero-measure set in $\mathcal{Y}$ under non-degenerate $\mathrm{p}_{\mathbf{y}\,|\,\mathbf{z}}[\boldsymbol{y}\,|\,\boldsymbol{z}]$ and so as $\epsilon \to 0$ the probability of accepting a sample / proposed move becomes zero. Applying ABC with a non-zero ϵ therefore can be seen as a practically motivated relaxation of the constraint that true and simulated data exactly match, and hence the 'approximate' in *Approximate Bayesian Computation*.

Alternatively the kernel k_ϵ can be given a modelling interpretation as representing uncertainty introduced by noise in the observations or mismatch between the unknown generative process by which the observed data was produced and the generative model [29, 36]. In practice however the kernel and choice of ϵ seems more often to be motivated on computational efficiency grounds [31].

3 Inference in the input space

Using the *Law of the Unconscious Statistician* we can reparametrise (3) in terms of the random inputs $\mathbf{u}$ to the generative model

$$\mathbb{E}\left[f(\mathbf{z}) \,|\, \bar{\mathbf{y}} = \bar{\mathbf{y}};\, \epsilon\right] \propto \int_{\mathcal{U}} f \circ \boldsymbol{G}_z(\boldsymbol{u})\, k_\epsilon\left(\bar{\mathbf{y}};\, \boldsymbol{G}_y(\boldsymbol{u})\right) \rho(\boldsymbol{u})\, \mathrm{d}\boldsymbol{u}.$$

This immediately indicates we can estimate ABC expectations by applying any MCMC method of choice in the input space to construct a chain with stationary density

$$\pi_\epsilon(\boldsymbol{u}) \propto k_\epsilon\left(\bar{\mathbf{y}};\, \boldsymbol{G}_y(\boldsymbol{u})\right) \rho(\boldsymbol{u}). \tag{8}$$

We can also however again consider the limit of $\epsilon \to 0$ as derived in the output space in (7). We have that the kernel term $k_\epsilon\left(\bar{\mathbf{y}};\, \boldsymbol{G}_y(\boldsymbol{u})\right) \to \delta\left(\bar{\mathbf{y}} - \boldsymbol{G}_y(\boldsymbol{u})\right)$ and so

$$\begin{aligned}\mathbb{E}\left[f(\mathbf{z}) \,|\, \mathbf{y} = \bar{\mathbf{y}}\right] &= \lim_{\epsilon \to 0} \left\{\mathbb{E}\left[f(\mathbf{z}) \,|\, \bar{\mathbf{y}} = \bar{\mathbf{y}};\, \epsilon\right]\right\} \\ &\propto \int_{\mathcal{U}} f \circ \boldsymbol{G}_z(\boldsymbol{u})\, \delta\left(\bar{\mathbf{y}} - \boldsymbol{G}_y(\boldsymbol{u})\right)\, \rho(\boldsymbol{u})\, \mathrm{d}\boldsymbol{u}.\end{aligned} \tag{9}$$

The Dirac delta term restricts the integral across the input space $\mathcal{U}$ to an embedded, $M - N_y$ dimensional, implicitly-defined manifold $C = \left\{\boldsymbol{u} \in \mathcal{U} : \boldsymbol{G}_y(\boldsymbol{u}) - \bar{\mathbf{y}} = \mathbf{0}\right\}$. By applying the the *Co-Area Formula* [11, §3.2.12] the integral with respect to the Lebesgue measure across $\mathcal{U}$ in (9) can be rewritten as integral across the embedded manifold C with respect the Hausdorff measure for the manifold

$$\begin{aligned}&\mathbb{E}\left[f(\mathbf{z}) \,|\, \mathbf{y} = \bar{\mathbf{y}}\right] \propto \\ &\int_C f \circ \boldsymbol{G}_z(\boldsymbol{u}) \left|\frac{\partial \boldsymbol{G}_y}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{G}_y}{\partial \boldsymbol{u}}^{\mathrm{T}}\right|^{-\frac{1}{2}} \rho(\boldsymbol{u})\, \mathcal{H}^{M-N_y}\left\{\mathrm{d}\boldsymbol{u}\right\}.\end{aligned} \tag{10}$$

Therefore if we generate a set of input MCMC samples $\{\boldsymbol{u}^{(s)}\}_{s=1}^S$ restricted to C and with invariant density

$$\pi(\boldsymbol{u}) \propto \left|\frac{\partial \boldsymbol{G}_y}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{G}_y}{\partial \boldsymbol{u}}^{\mathrm{T}}\right|^{-\frac{1}{2}} \rho(\boldsymbol{u}) \tag{11}$$

with respect to the Hausdorff measure for the manifold, then

$$\mathbb{E}\left[f(\mathbf{z}) \,|\, \mathbf{y} = \mathbf{y}\right] = \lim_{S \to \infty} \frac{1}{S} \sum_{s=1}^{S} \left[f \circ \boldsymbol{G}_z\left(\boldsymbol{u}^{(s)}\right)\right]. \tag{12}$$

Intuitively the determinant term in (11) adjusts for the change in the infinitesimal 'thickness' (extent in directions orthogonal to the tangent space) of the manifold when mapping through the generator function $\boldsymbol{G}_y$. The result (10) is given in a slightly different form in [9]. A derivation is provided in Appendix A of the Supplementary Material.

A general framework for performing asymptotically exact inference in differentiable generative models is therefore to define MCMC updates which explore the target density (11) on the constraint manifold C. We propose here to use a method which simulates the dynamics of a constrained mechanical system.

4 Constrained Hamiltonian Monte Carlo

In *Hamiltonian Monte Carlo* (HMC) [10, 26] the vector variable of interest $\mathbf{u}$ is augmented with a momentum variable $\mathbf{p} \in \mathbb{R}^M$. The momentum is taken to be independently Gaussian distributed with zero mean and covariance $\boldsymbol{M}$, often called the mass matrix. The new joint target density is then proportional to $\exp\left[-H(\boldsymbol{u}, \boldsymbol{p})\right]$ where

$$H(\boldsymbol{u}, \boldsymbol{p}) = -\log \pi(\boldsymbol{u}) + \frac{1}{2} \boldsymbol{p}^{\mathrm{T}} \boldsymbol{M}^{-1} \boldsymbol{p} \tag{13}$$

is termed the *Hamiltonian*.

The canonical Hamiltonian dynamic is described by the system of ordinary differential equations

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \frac{\partial H}{\partial \boldsymbol{p}} = \boldsymbol{M}^{-1} \boldsymbol{p}, \quad \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{u}} = \frac{\partial \log \pi}{\partial \boldsymbol{u}}. \tag{14}$$

This dynamic is time-reversible, measure-preserving and exactly conserves the Hamiltonian. Symplectic integrators allow approximate integration of the Hamiltonian flow while maintaining the time-reversibility and measure-preservation properties. Subject to stability bounds on the time-step, such integrators will exactly conserve some 'nearby' Hamiltonian, and so the change in the Hamiltonian will tend to remain small even over long simulated trajectories [17].

These properties make simulated Hamiltonian dynamics an ideal proposal mechanism for a Metropolis MCMC method. The Metropolis accept ratio for a proposal $(\boldsymbol{u}_\mathrm{p},\, \boldsymbol{p}_\mathrm{p})$ generated by simulating the dynamic N_s time steps forward from $(\boldsymbol{u},\, \boldsymbol{p})$ with a symplectic integrator and then negating the momentum, is simply $\exp\left\{H(\boldsymbol{u},\, \boldsymbol{p}) - H(\boldsymbol{u}_\mathrm{p},\, \boldsymbol{p}_\mathrm{p})\right\}$. Typically the change in the Hamiltonian will be small and so the probability of acceptance high. To ensure ergodicity, dynamic moves can be interspersed with updates independently sampling a new momentum from $\mathcal{N}(\mathbf{0},\, \boldsymbol{M})$.

In our case the system is subject to a constraint of the form $\boldsymbol{C}(\boldsymbol{u}) = \boldsymbol{G}_y(\boldsymbol{u}) - \bar{\mathbf{y}} = \mathbf{0}$. By introducing Lagrangian multipliers λ_i for each of the constraints, the Hamiltonian for a constrained system can be written as

$$H(\boldsymbol{u}, \boldsymbol{p}) = -\log \pi(\boldsymbol{u}) + \frac{1}{2} \boldsymbol{p}^{\mathrm{T}} \boldsymbol{M}^{-1} \boldsymbol{p} + \boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{C}(\boldsymbol{u}), \tag{15}$$

and a corresponding constrained Hamiltonian dynamic

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{M}^{-1} \boldsymbol{p}, \quad \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = \frac{\partial \log \pi}{\partial \boldsymbol{u}} - \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}^{\mathrm{T}} \lambda, \tag{16}$$

$$\text{subject to } \boldsymbol{C}(\boldsymbol{u}) = \mathbf{0}, \quad \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}} \boldsymbol{M}^{-1} \boldsymbol{p} = \mathbf{0}. \tag{17}$$

A popular numerical integrator for simulating constrained Hamiltonian dynamics is RATTLE [2] (and the algebraically equivalent SHAKE [32] scheme). This a natural generalisation of the Störmer-Verlet (leapfrog) integrator typically used in standard HMC with additional projection steps in

which the Lagrange multipliers λ are solved for to satisfy the conditions (17). RATTLE and SHAKE maintain the properties of being time-reversible, measure-preserving and symplectic [18].

The use of constrained dynamics in HMC has been proposed several times. In the molecular dynamics literature, both [14] and [19] suggest using a simulated constrained dynamic within a HMC framework to estimate free-energy profiles.

Most relevantly here [6] proposes using a constrained HMC variant to perform inference in distributions defined on implicitly defined embedded non-linear manifolds. This gives sufficient conditions on H, C and $\boldsymbol{C}$ for the scheme to satisfy detailed balance and be ergodic: that H is C^2 continuous, and C is a connected smooth and differentiable manifold and $\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}$ has full row-rank everywhere.

5 Method

Our constrained HMC implementation is shown in algorithm 1. We use a generalisation of the RATTLE scheme to simulate the dynamic. The inner updates of the state to solve for the geodesic motion on the constraint manifold are split into multiple smaller steps, which can be considered a special case of the scheme described in [16]. This allows more flexibility in choosing an appropriately small step-size to ensure convergence of the iterative solution of the equations projecting on to the constraint manifold while still allowing a more efficient larger step size for updates to the momentum due to the negative log density gradient. We have assumed $\boldsymbol{M} = \boldsymbol{I}$ here; other mass matrix choices can be equivalently implemented by adding an initial linear transformation stage in the generator.

Each inner geodesic time-step involves making an unconstrained update $\boldsymbol{u} \to \tilde{\boldsymbol{u}}$ and then projecting $\tilde{\boldsymbol{u}}$ back on to C by solving for λ which satisfy $\boldsymbol{C}(\tilde{\boldsymbol{u}} - \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}^{\mathrm{T}} \lambda) = \boldsymbol{0}$. This is performed in the function PROJECTPOSITION in algorithm 1. Here we use a quasi-Newton method for solving the system of equations in the projection step. The true Newton update would be

$$\boldsymbol{u}^{(t)} \leftarrow \boldsymbol{u}^{(t)} - \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}\Big|_{\boldsymbol{u}^{(t-1)}}^{\mathrm{T}} \left\{ \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}\Big|_{\boldsymbol{u}^{(t)}} \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}\Big|_{\boldsymbol{u}^{(t-1)}}^{\mathrm{T}} \right\}^{-1} \boldsymbol{C}(\boldsymbol{u}^{(t)}).$$

This requires recalculating the Jacobian and solving a dense linear system within the optimisation loop. Instead we use a symmetric quasi-Newton update as proposed in [3], the Jacobian Gram matrix $\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}^{\mathrm{T}}$ evaluated at the previous state used to condition the moves. This matrix is positive-definite and a Cholesky decomposition can be calculated outside the optimisation loop allowing cheaper quadratic cost solves within the loop. In the rare cases where the quasi-Newton iteration fails we fall back to a MINPACK [24] implementation of the robust Powell's Hybrid method [28].

For larger systems, the Cholesky decomposition of the constraint Jacobian Gram matrix (line 36) will become a dominant cost, generally scaling cubically with N_y. The elementwise or autoregressive noise structures of many models however allows a significantly reduced quadratically scaling computational cost as explained in the supplementary material in Appendix C.

The momentum updates in the SIMULATEDYNAMIC routine require evaluating the gradient of the logarithm of the target density (11). This can by achieved by using automatic differentiation to calculate the gradient from the expression given in (11), however both the log-density and gradient can be more efficiently calculated by reusing the Cholesky decomposition of the constraint Jacobian Gram matrix computed in line 36. Details are given in Appendix B.

In the PROJECTPOSITION routine convergence is signalled when the elementwise maximum absolute value of the constraint function is below some tolerance ϵ. This acts analogously to the ϵ parameter in ABC methods, however here we typically set this parameter to some multiple of machine precision and so the approximation introduced is comparable to that otherwise incurred for using non-exact arithmetic.

A final implementation detail is the requirement to find some initial $\boldsymbol{u}$ satisfying $\boldsymbol{C}(\boldsymbol{u}) = \boldsymbol{0}$. In some cases structure in the generator function $\boldsymbol{G}_y$ such as that described in Appendix C can be leveraged to come up with an efficient non-iterative scheme for finding a constraint satisfying $\boldsymbol{u}$. For general generators, we can choose a subset of the inputs (or linear projections of the inputs) of dimensionality N_y and plug the resulting system of equations into a black-box solver.

6 Related work

Closely related is the *Constrained HMC* method of [6]. This demonstrates the validity of the constrained HMC framework theoretically and experimentally, and we do not claim any original contribution in this respect. The focus in [6] is on performing inference in distributions inherently defined on a fixed non-Euclidean manifold such as the unit sphere or space of orthogonal matrices.

Our work builds on [6] by highlighting that conditioning on the output of a generator imposes a constraint on its inputs and so defines a density in input space restricted to some manifold. Unlike the cases considered in [6] our constraints are therefore data-driven and the target density on the manifold implicitly defined by a generator function and base density.

Geodesic Monte Carlo [7] also considers applying a HMC scheme to sample from non-linear manifolds embedded in a Euclidean space. Similarly to [6] however the motivation is performing inference with respect to distributions explicitly defined on a manifold such as directional statistics.

The method presented in [7] uses an exact solution for the

Algorithm 1 Constrained Hamiltonian Monte Carlo algorithm

Input: $(\boldsymbol{u}, \boldsymbol{p})$: current state pair with $\boldsymbol{C}(\boldsymbol{u}) = \boldsymbol{0}$, $\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}\boldsymbol{p} = \boldsymbol{0}$; $(\boldsymbol{J}, \boldsymbol{L})$: constraint Jacobian and Gram matrix Cholesky factor at current $\boldsymbol{u}$; ϵ : convergence tolerance; δt : time step; N_s : number of time steps to simulate; N_g : number of geodesic updates per time step.

Output: $(\boldsymbol{u}', \boldsymbol{p}')$: new state pair with $\boldsymbol{C}(\boldsymbol{u}') = \boldsymbol{0}$, $\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}'}\boldsymbol{p}' = \boldsymbol{0}$; $(\boldsymbol{J}', \boldsymbol{L}')$: constraint Jacobian and Gram matrix Cholesky factor at new $\boldsymbol{u}'$.

$\boldsymbol{u}_\mathrm{p}, \boldsymbol{p}_\mathrm{p}, \boldsymbol{J}_\mathrm{p}, \boldsymbol{L}_\mathrm{p} \leftarrow$ SIMULATEDYNAMIC$(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
$r \leftarrow \mathrm{Uniform}(0, 1)$
if $r < \exp\left\{H(\boldsymbol{u}, \boldsymbol{p}) - H(\boldsymbol{u}_\mathrm{p}, \boldsymbol{p}_\mathrm{p})\right\}$ **then**
    $\boldsymbol{u}', \boldsymbol{p}', \boldsymbol{J}', \boldsymbol{L}' \leftarrow \boldsymbol{u}_\mathrm{p}, \boldsymbol{p}_\mathrm{p}, \boldsymbol{J}_\mathrm{p}, \boldsymbol{L}_\mathrm{p}$
else
    $\boldsymbol{u}', \boldsymbol{p}', \boldsymbol{J}', \boldsymbol{L}' \leftarrow \boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L}$
$\boldsymbol{n} \leftarrow \mathrm{Normal}(\boldsymbol{0}, \boldsymbol{I})$
$\boldsymbol{p}' \leftarrow$ PROJECTMOMENTUM$(\boldsymbol{n}, \boldsymbol{J}', \boldsymbol{L}')$

function SIMULATEDYNAMIC$(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
    $\tilde{\boldsymbol{p}} \leftarrow \boldsymbol{p} + \frac{\delta t}{2} \left.\frac{\partial \log \pi}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}}$
    $\boldsymbol{p} \leftarrow$ PROJECTMOMENTUM$(\tilde{\boldsymbol{p}}, \boldsymbol{J}, \boldsymbol{L})$
    $\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L} \leftarrow$ SIMULATEGEODESIC$(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
    for $s \in \left\{2 \dots N_s\right\}$ **do**
        $\tilde{\boldsymbol{p}} \leftarrow \boldsymbol{p} + \delta t \left.\frac{\partial \log \pi}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}}$
        $\boldsymbol{p} \leftarrow$ PROJECTMOMENTUM$(\tilde{\boldsymbol{p}}, \boldsymbol{J}, \boldsymbol{L})$
        $\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L} \leftarrow$ SIMULATEGEODESIC$(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
    $\tilde{\boldsymbol{p}} \leftarrow \boldsymbol{p} + \frac{\delta t}{2} \left.\frac{\partial \log \pi}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}}$
    $\boldsymbol{p} \leftarrow$ PROJECTMOMENTUM$(\tilde{\boldsymbol{p}}, \boldsymbol{J}, \boldsymbol{L})$
    return $\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L}$

function PROJECTPOSITION$(\boldsymbol{u}, \boldsymbol{J}, \boldsymbol{L})$
    $\boldsymbol{c} \leftarrow \boldsymbol{C}(\boldsymbol{u})$
    while $\|\boldsymbol{c}\|_\infty > \epsilon$ **do**
        $\boldsymbol{u} \leftarrow \boldsymbol{u} - \boldsymbol{J}^\mathrm{T}\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{c}$
        $\boldsymbol{c} \leftarrow \boldsymbol{C}(\boldsymbol{u})$
    return $\boldsymbol{u}$

function PROJECTMOMENTUM$(\boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
    return $\boldsymbol{p} - \boldsymbol{J}^\mathrm{T}\boldsymbol{L}^{-\mathrm{T}}\boldsymbol{L}^{-1}\boldsymbol{J}\boldsymbol{p}$

function SIMULATEGEODESIC$(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L})$
    for $i \in \left\{1 \dots N_g\right\}$ **do**
        $\tilde{\boldsymbol{u}} \leftarrow \boldsymbol{u} + \frac{\delta t}{N_g}\boldsymbol{p}$
        $\boldsymbol{u}' \leftarrow$ PROJECTPOSITION$(\tilde{\boldsymbol{u}}, \boldsymbol{J}, \boldsymbol{L})$
        $\boldsymbol{J} \leftarrow \left.\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}'}$
        $\boldsymbol{L} \leftarrow \mathrm{chol}\left(\boldsymbol{J}\boldsymbol{J}^\mathrm{T}\right)$
        $\tilde{\boldsymbol{p}} \leftarrow \frac{N_g}{\delta t}(\boldsymbol{u}' - \boldsymbol{u})$
        $\boldsymbol{p} \leftarrow$ PROJECTMOMENTUM$(\tilde{\boldsymbol{p}}, \boldsymbol{J}, \boldsymbol{L})$
        $\boldsymbol{u} \leftarrow \boldsymbol{u}'$
    return $\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{J}, \boldsymbol{L}$

geodesic flow on the manifold. Our use of constrained Hamiltonian dynamics, and in particular the geodesic integration scheme of [16], can be considered an extension for cases when an exact geodesic solution is not available. Instead the geodesic flow is approximately simulated while still maintaining the required measure-preservation and reversibility properties for validity of the overall HMC scheme.

Hamiltonian ABC [22], also proposes applying HMC to perform inference in simulator models as considered here. An ABC set-up is used with a Gaussian synthetic-likelihood formed by estimating moments from simulated data.

Rather than using automatic differentiation to exactly calculate gradients of the generator function, *Hamiltonian ABC* uses a stochastic gradient estimator. This is based on previous work considering methods for using a stochastic gradients within HMC [8, 35]. It has been suggested however that the use of stochastic gradients can destroy the favourable properties of Hamiltonian dynamics which enable coherent exploration of high dimensional state spaces [5].

In *Hamiltonian ABC* it is also observed that representing the generative model as a deterministic function by fixing the random inputs to the generator is a useful method for improving exploration of the state space. This is achieved by including the seed of the pseudo-random number generator in the chain state rather than the set of random inputs.

Also related is *Optimisation Monte Carlo* [23]. The authors propose using an optimiser to find parameters of a simulator model consistent with observed data (to within some tolerance ϵ) given fixed random inputs sampled independently. The optimisation is not measure-preserving and so the Jacobian of the map is approximated with finite differences to weight the samples. Our method also uses an optimiser to find inputs consistent with the observations, however by using a measure-preserving dynamic we avoid having to reweight samples which can scale poorly with dimensionality.

Our method also differs in treating all inputs to a generator equivalently; while the *Optimisation Monte Carlo* authors similarly identify the simulator models as deterministic functions they distinguish between parameters and random inputs, optimising the first and independently sampling the latter. This can lead to random inputs being sampled for which no parameters can be found consistent with the observations (even with a 'soft' within ϵ constraint). Although optimisation failure is also potentially an issue for our method, as we jointly optimise all inputs and are approximating some exact continuous time constraint-satisfying dynamic we found this occurred rarely in practice if an appropriate step size is chosen. Our method can also be applied in cases were the parameter dimension is greater than the dimension of the observations unlike *Optimization Monte Carlo*.

7 Experiments

To illustrate the general applicability of our method we performed inference tasks in three diverse settings: parameter inference in a stochastic Lotka-Volterra predator-prey model simulation, 3D human pose and camera parameter inference given 2D joint position information and finally in-painting of missing regions of digit images using a generative model trained on MNIST. In all three experiments Theano [34] was used to specify the generator function and calculate the required derivatives. All experiments were run on an Intel Core i5-2400 quad-core CPU. Python code for the experiments is available at `https://git.io/dgm`.

7.1 Lotka–Volterra parameter inference

As a first demonstration we considered a stochastic continuous state variant of the Lotka–Volterra model, a common example problem for ABC methods e.g. [23]. In particular we consider parameter inference given a simulated solution of the following stochastic differential equations

$$\mathrm{d}y_1 = \left(z_1 y_1 - z_2 y_1 y_2\right)\mathrm{d}t + \mathrm{d}n_1, \tag{18}$$

$$\mathrm{d}y_2 = \left(-z_3 y_2 + z_4 y_1 y_2\right)\mathrm{d}t + \mathrm{d}n_2, \tag{19}$$

where y_1 represents the prey population, y_2 the predator population, $\left\{z_i\right\}_{i=1}^4$ the system parameters and n_1 and n_2 zero-mean, unit variance white noise processes.

The observed data was generated with an Euler-Maruyama discretisation, time-step 1, initial condition $y_1^{(0)} = y_2^{(0)} = 100$ and $z_1 = 0.4$, $z_2 = 0.005$, $z_3 = 0.05$, $z_4 = 0.001$ (chosen to give stable dynamics). The simulation was run for 50 time-steps with the observed outputs defined as the concatenated vector $\boldsymbol{y} = \left[y_1^{(1)}\ y_2^{(1)}\ \ldots\ y_1^{(50)}\ y_2^{(50)}\right]$. Log-normal priors $z_i \sim \log\mathcal{N}(-2, 1)$ were place on the system parameters.

We compared our method to various ABC approaches (§2) using a uniform ball kernel with radius ϵ. ABC rejection failed catastrophically, with no acceptances in 10^6 samples even with a large $\epsilon = 1000$. ABC MCMC with a Gaussian proposal distribution q also performed very poorly with the dynamic having zero acceptances over multiple runs of 10^5 updates for $\epsilon = 100$ and getting stuck at points in parameter space over many updates for larger $\epsilon = 1000$, even with small proposal steps. Based on a method proposed in the pseudo-marginal literature [25], we tried using alternating elliptical slice sampling updates of the random inputs used to generate the parameters $\mathbf{u}_z$ and remaining random inputs $\mathbf{u}_y$ used to generate the simulated observations given parameters. The slice sampling updates locally adapt the size of steps made to ensure a move can always be made. Using this method we were able to obtain reasonable convergence over long runs for both $\epsilon = 100$ and $\epsilon = 10$.

The results are summarised in Figure 2. Figure 2a shows the simulated data used as observations and ABC sample trajectories for $\epsilon = 10$ and $\epsilon = 100$. Though both samples follow the general trends of the observed data there are large discrepancies particularly for $\epsilon = 100$. Our method in contrast samples parameters generating trajectories *exactly* matching the observations at all points. Figure 2b shows the marginal histograms for the parameters. The inferred posterior on the parameters are significantly more tightly distributed about the true values used to generate the observations for our approach and the $\epsilon = 10$ case compared to the results for $\epsilon = 100$; even for the $\epsilon = 10$ case however it can be seen that there are spurious appearing peaks in the distributions for z_3 and z_4.

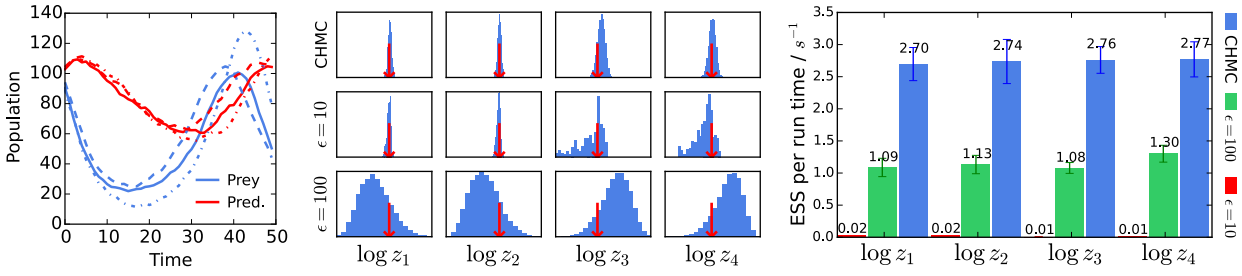
Figure 2c shows the relative sampling efficiency of our approach against the ABC methods, as measured by the *effective sample sizes* (ESS) (computed with R-CODA [27]) normalised by run time averaged across 10 sampling runs for each method. Despite the significantly higher per-update cost in our method, the coherent movement about the state space afforded by the Hamiltonian dynamic gave significantly better performance even over the very approximate $\epsilon = 100$ case.

7.2 Human pose and camera model inference

For our next experiment we considered inferring a 3D human pose and camera model from 2D projection(s) of joint positions. We used a 19 joint skeleton model, with a prior density over poses parametrised by 47 local joint angles $\mathbf{z}_a$ learnt from the *PosePrior* motion capture data-set [1] with a VAE with a 30 dimensional hidden representation $\mathbf{h}$. For the bone lengths $\mathbf{z}_b$, a log-normal model was fitted to data from the *ANSUR* anthropometric data-set [13], due to symmetry 13 independent lengths being specified. A simple pin-hole projective camera model with 3 position parameters $\mathbf{z}_c$ and fixed focal-length was used[3]. A log-normal prior was placed on the depth co-ordinate $\mathrm{z}_{c,3}$ to enforce positivity with normal priors on the other two co-ordinates $\mathrm{z}_{c,1}$ and $\mathrm{z}_{c,2}$. A small amount of Gaussian noise was added to the projected positions to give the observed 2D joint positions $\mathbf{y}$. This ensured the constraint Jacobian was full row-rank everywhere and gave a known hierarchical joint density on $\left\{\mathbf{y}, \mathbf{h}, \mathbf{z}_a, \mathbf{z}_b, \mathbf{z}_c\right\}$ allowing comparison with HMC as a baseline.

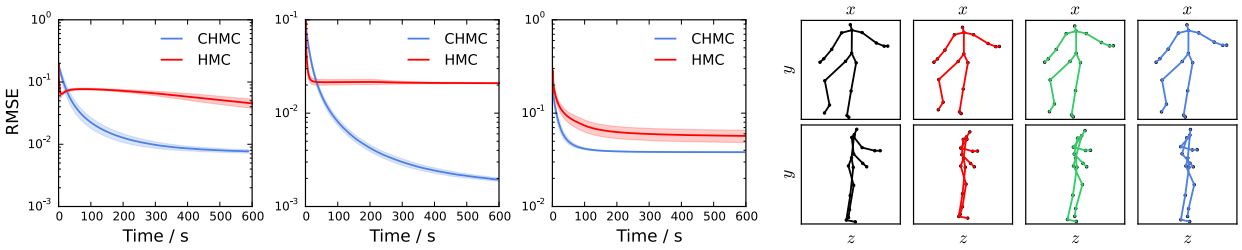
We first considered binocular pose estimation, with the 3D scene information inferred given 2D projections from two cameras with a known offset between them. In this stereo vision case, the disparity between the projections gives information about the depth direction and so we would expect the posterior distribution on the 3D pose to be tightly distributed around the true values used to generate the observations. We compared our constrained HMC method to running standard HMC on the conditional density of $\left\{\mathbf{h}, \mathbf{z}_a, \mathbf{z}_b, \mathbf{z}_c\right\}$ given $\mathbf{y}$.

[3]The camera orientation was assumed fixed to avoid replicating the degrees of freedom specified by the angular orientation of the root joint of the skeleton: only the relative camera–skeleton orientation is important.



(a) (b) (c)

Figure 2: **Lotka–Volterra** (a) Observed predator-prey populations (solid) and ABC sample trajectories with $\epsilon = 10$ (dashed) and $\epsilon = 100$ (dot-dashed). (b) Marginal empirical histograms for the (logarithm of the) four parameters (columns) from constrained HMC samples (top) and ABC samples with $\epsilon = 10$ (middle) and $\epsilon = 100$ (bottom). Horizontal axes shared across columns. Red arrows indicate true parameter values. (c) Mean ESS normalised by compute time for each of four parameters for ABC with $\epsilon = 10$ (red), $\epsilon = 100$ (green) and our method (blue). Error bars show ± 3 standard errors of mean.



(a) (b)

Figure 3: **Human pose** (a) RMSEs of 3D pose posterior mean estimates given binocular projections, using samples from our method (blue) versus running HMC in hierarchical model (red) for three different scenes sampled from the prior. Horizontal axes show computation time to produce number of samples in estimate. Solid curves are average RMSE over 10 runs with different seeds and shaded regions show ± 3 standard errors of mean. (b) Orthographic projections (top: front view, bottom: side view) of 3D poses consistent with monocular projections. Left most pair (black) shows pose used to generate observations, right three show constrained HMC samples.

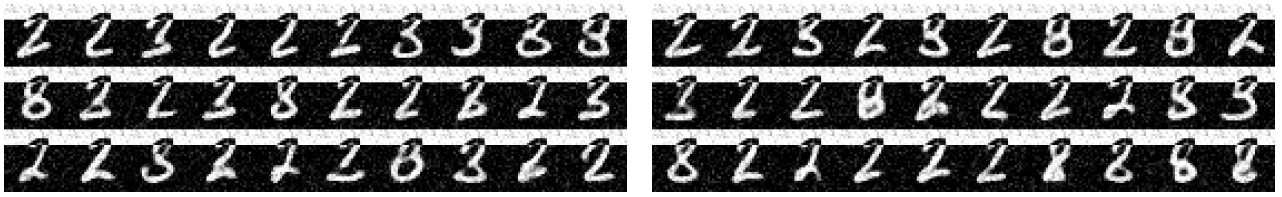
Figure 3a shows the *root mean squared error* (RMSE) between the posterior mean estimate of the 3D joint positions and the true positions used to generate the observations as the number of samples included in the estimate increases for three test scenes. For both methods the horizontal axis has been scaled by run time.

The constrained HMC method (blue curves) tends to give significantly more accurate estimates particularly over longer periods. Visually inspecting the sampled poses and individual run traces (not shown) it seems that the HMC runs tended to often get stuck in local modes corresponding to a subset of joints being 'incorrectly' positioned while still broadly matching the (noisy) projections. The complex dependencies of the joint positions on the angle parameters mean the dynamic struggles to find an update which brings the 'incorrect' joints closer to their true positions without moving other joints out of line. The constrained HMC method seemed to be less susceptible to this issue.

We also considered inferring 3D scene information from a single 2D projection. Monocular projection is inherently information destroying with significant uncertainty to the true pose and camera parameters which generated the observations. Figure 3b shows pairs of orthographic projections of 3D poses: the left most column is the pose used to generate the projection conditioned on and the right three columns are poses sampled using constrained HMC consistent with the observations. The top row shows front x–y views, corresponding to the camera view though with a orthographic rather than perspective projection, the bottom row shows side z–y views with the z axis the depth from the camera. The dynamic is able to move between a range of plausible poses consistent with the observations while reflecting the inherent depth ambiguity from the monocular projection.

7.3 MNIST in-painting

As a final task we considered inferring in-paintings for a missing region of a digit image $\mathbf{z}$ given knowledge of the rest of the pixel values $\mathbf{y}$. A Gaussian VAE trained on MNIST was used as the generative model, with a 50-dimensional hidden code $\mathbf{h}$. We compared our method to running HMC in the known conditional density on $\mathbf{h}$ given $\mathbf{y}$ ($\mathbf{z}$ can then be directly sampled from its Gaussian conditional given $\mathbf{h}$).



(a) Constrained HMC samples

(b) HMC in hierarchical model samples

Figure 4: **MNIST** In-painting samples. The top black-on-white quarter of each image is the fixed observed region and the remaining white-on-black region the proposed in-painting. In left-right scan order the images in (a) are consecutive samples from a run; in (b) the images are every 40^{th} sample to account for the quicker updates.

Example samples are shown in Figure 4. In this case the constrained and standard HMC approaches appear to be performing similarly, with both able to find a range of plausible in-paintings given the observed pixels. Without cost adjustment the standard HMC samples show greater correlation between subsequent updates, however for a fairer comparison the samples shown were thinned to account for the approximately $40\times$ larger run-time per constrained HMC sample. Although the constrained dynamic does not improve efficiency here neither does it seem to hurt it.

8 Discussion

We have presented a generally applicable framework for performing inference in differentiable generative models. Though simulating the constrained Hamiltonian dynamic is computationally costly, the resulting coherent exploration of the state space can lead to significantly improved sampling efficiency over alternative methods.

Further our approach allows asymptotically exact inference in differentiable generative models where ABC methods might otherwise be used. We suggest an approach for dealing with two of the key issues in ABC methods — enabling inference in continuous spaces as ϵ collapses to zero and allowing efficient inference when conditioning on high-dimensional observations without the need for dimensionality reduction with summary statistics (and the resulting task of choosing appropriate summary statistics). As well as being of practical importance itself, this approach should be useful in providing 'ground truth' inferences in more complex models to assess the affect of the approximations used in ABC methods on the quality of the inferences.

In molecular simulations, constrained dynamics are often used to improve efficiency. Intra-molecular motion is removed by fixing bond lengths. This allows a larger time-step to be used due to the removal of high-frequency bond oscillations [16]. An analogous effect is present when performing inference in an ABC setting with a ϵ kernel 'soft-constraint' to enforce consistency between the inputs and observed outputs. As $\epsilon \to 0$ the scales over which the inputs density changes value in directions orthogonal to the constraint manifold and along directions tangential to the manifold increasingly differ. To stay within the soft constraint a very small step-size needs to be used. Using a constrained dynamic decouples the motion on the constraint manifold from the steps to project on to it, allowing more efficient larger steps to be used for moving on the manifold.

A limitation of our method is the requirement of differentiability of the generator. This prevents applying our approach to generative models which use discontinuous operations or discrete random inputs. In some cases conditioned on fixed values of discrete random inputs the generator may still be differentiable and so the proposed method can be used to update the continuous random inputs given values of the discrete inputs. This would need to be alternated with updates to the discrete inputs, which would require devising methods for updating the discrete inputs to the generator while constraining its output to exactly match observations.

A common technique in ABC methods is to define a kernel or distance measure in terms of summary statistics of the observed data [20]. This is necessary in standard ABC approaches to cope with the 'curse of dimensionality' with the probability of accepting samples / moves for a fixed tolerance ϵ exponentially decreasing as the dimensionality of the observations conditioned on increases. Although as already noted the proposed method is better able to cope with high observation dimensions, if appropriate informative statistics are available (e.g. based on expert knowledge) and these are differentiable functions of the generator outputs, they can be easily integrated in to the proposed method by absorbing the statistic computation in to the definition of $\boldsymbol{G}_y$.

Acknowledgements

Thanks to Iain Murray for several useful discussions and to Ben Leimkuhler for pointing out the potential of using a geodesic integration scheme. This work was supported in part by grants EP/F500385/1 and BB/F529254/1 for the University of Edinburgh School of Informatics Doctoral Training Centre in Neuroinformatics and Computational Neuroscience (www.anc.ac.uk/dtc) from the UK Engineering and Physical Sciences Research Council (EPSRC), UK Biotechnology and Biological Sciences Research Council (BBSRC), and the UK Medical Research Council (MRC).

References

[1] I. Akhter and M. J. Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[2] H. C. Andersen. RATTLE: A velocity version of the SHAKE algorithm for molecular dynamics calculations. *Journal of Computational Physics*, 1983.

[3] E. Barth, K. Kuczera, B. Leimkuhler, and R. D. Skeel. Algorithms for constrained molecular dynamics. *Journal of computational chemistry*, 1995.

[4] M. A. Beaumont, W. Zhang, and D. J. Balding. Approximate Bayesian computation in population genetics. *Genetics*, 2002.

[5] M. Betancourt. The fundamental incompatibility of scalable Hamiltonian Monte Carlo and naive data subsampling. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[6] M. A. Brubaker, M. Salzmann, and R. Urtasun. A family of MCMC methods on implicitly defined manifolds. In *International Conference on Artificial Intelligence and Statistics*, 2012.

[7] S. Byrne and M. Girolami. Geodesic Monte Carlo on embedded manifolds. *Scandinavian Journal of Statistics*, 2013.

[8] T. Chen, E. Fox, and C. Guestrin. Stochastic gradient Hamiltonian Monte Carlo. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[9] P. Diaconis, S. Holmes, and M. Shahshahani. Sampling from a manifold. In *Advances in Modern Statistical Theory and Applications*, pages 102–125. Institute of Mathematical Statistics, 2013.

[10] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 1987.

[11] H. Federer. *Geometric measure theory*. Springer, 2014.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[13] C. C. Gordon, T. Churchill, C. E. Clauser, B. Bradtmiller, J. T. McConville, I. Tebbets, and R. A. Walker. Anthropometric survey of US army personell: Final report. Technical report, United States Army, 1988.

[14] C. Hartmann and C. Schutte. A constrained hybrid Monte-Carlo algorithm and the problem of calculating the free energy in several variables. *ZAMM-Zeitschrift fur Angewandte Mathematik und Mechanik*, 2005.

[15] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[16] B. Leimkuhler and C. Matthews. Efficient molecular dynamics using geodesic integration and solvent–solute splitting. In *Proc. R. Soc. A*. The Royal Society, 2016.

[17] B. Leimkuhler and S. Reich. *Simulating Hamiltonian dynamics*. Cambridge University Press, 2004.

[18] B. J. Leimkuhler and R. D. Skeel. Symplectic numerical integrators in constrained Hamiltonian systems. *Journal of Computational Physics*, 1994.

[19] T. Lelièvre, M. Rousset, and G. Stoltz. Langevin dynamics with constraints and computation of free energy differences. *Mathematics of computation*, 2012.

[20] J.-M. Marin, P. Pudlo, C. P. Robert, and R. J. Ryder. Approximate Bayesian computational methods. *Statistics and Computing*, 2012.

[21] P. Marjoram, J. Molitor, V. Plagnol, and S. Tavaré. Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 2003.

[22] E. Meeds, R. Leenders, and M. Welling. Hamiltonian ABC. In *Proceedings of 31st Conference of Uncertainty in Artificial Intelligence*, 2015.

[23] T. Meeds and M. Welling. Optimization Monte Carlo: Efficient and embarrassingly parallel likelihood-free inference. In *Advances in Neural Information Processing Systems*, 2015.

[24] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. *User Guide for MINPACK-1*. ANL-80-74, Argonne National Laboratory, 1980.

[25] I. Murray and M. M. Graham. Pseudo-marginal slice sampling. In *International Conference on Artificial Intelligence and Statistics*, 2016.

[26] R. M. Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2011.

[27] M. Plummer, N. Best, K. Cowles, and K. Vines. CODA: Convergence diagnosis and output analysis for MCMC. *R News*, 2006.

[28] M. J. D. Powell. *Numerical Methods for Nonlinear Algebraic Equations*, chapter A Hybrid Method for Nonlinear Equations. Gordon and Breach, 1970.

[29] O. Ratmann, C. Andrieu, C. Wiuf, and S. Richardson. Model criticism based on likelihood-free inference, with an application to protein network evolution. *Proceedings of the National Academy of Sciences*, 2009.

[30] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of The 31st International Conference on Machine Learning*, 2014.

[31] C. P. Robert, K. Mengersen, and C. Chen. Model choice versus model criticism. *Proceedings of the National Academy of Sciences of the United States of America*, 2010.

[32] J.-P. Ryckaert, G. Ciccotti, and H. J. Berendsen. Numerical integration of the Cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes. *Journal of Computational Physics*, 1977.

[33] J. Schulman, N. Heess, T. Weber, and P. Abbeel. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, 2015.

[34] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, 2016.

[35] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.

[36] R. D. Wilkinson. Approximate Bayesian computation (ABC) gives exact results under the assumption of model error. *Statistical applications in genetics and molecular biology*, 2013.