Fixed-kinetic Neural Hamiltonian Flows for enhanced interpretability and reduced complexity

Vincent Souveton, Arnaud Guillin, Jens Jasche, Guilhem Lavaux, Manon Michel

Fixed-kinetic Neural Hamiltonian Flows for enhanced interpretability and reduced complexity

Vincent Souveton [1] **Arnaud Guillin*** [1 2] **Jens Jasche*** [3 4] **Guilhem Lavaux*** [4] **Manon Michel*** [1]

Abstract

Normalizing Flows (NF) are Generative models which are particularly robust and allow for exact sampling of the learned distribution. They however require the design of an invertible mapping, whose Jacobian determinant has to be computable. Recently introduced, Neural Hamiltonian Flows (NHF) are based on Hamiltonian dynamics-based Flows, which are continuous, volume-preserving and invertible and thus make for natural candidates for robust NF architectures. In particular, their similarity to classical Mechanics could lead to easier interpretability of the learned mapping. However, despite being Physics-inspired architectures, the originally introduced NHF architecture still poses a challenge to interpretability. For this reason, in this work, we introduce a fixed kinetic energy version of the NHF model. Inspired by physics, our approach improves interpretability and requires less parameters than previously proposed architectures. We then study the robustness of the NHF architectures to the choice of hyperparameters. We analyze the impact of the number of leapfrog steps, the integration time and the number of neurons per hidden layer, as well as the choice of prior distribution, on sampling a multimodal 2D mixture. The NHF architecture is robust to these choices, especially the fixed-kinetic energy model. Finally, we adapt NHF to the context of Bayesian inference and illustrate our method on sampling the posterior distribution of two cosmological parameters knowing type Ia supernovae observations.

*Equal contribution [1]Laboratoire de Mathématiques Blaise Pascal UMR 6620, CNRS, Université Clermont-Auvergne, Aubière, France. [2]Institut Universitaire de France. [3]The Oskar Klein Centre, Department of Physics, Stockholm University, AlbaNova University Centre, SE 106 91 Stockholm, Sweden. [4]CNRS, Sorbonne Université, UMR 7095, Institut d'Astrophysique de Paris, 98 bis bd Arago, 75014 Paris, France.. Correspondence to: Vincent Souveton <vincent.souveton@ext.uca.fr>.

1. Introduction

Generative models are widely used for sampling high-dimensional probability distributions with applications from molecular biology (Lopez et al., 2020) to cosmology (Rodriguez et al., 2018) or medical science (Frazer et al., 2021). Traditional architectures like Generative Adversarial Networks (Goodfellow et al., 2014) have shown impressive results in image generation but since their adversarial loss seeks a saddle point rather than a local minimum, GANs are notoriously hard to train and may suffer from mode-collapse (Lin et al., 2018; Arjovsky & Bottou, 2017; Berard et al., 2020). More robust techniques like Normalizing Flows (NF) were developed (Tabak & Vanden-Eijnden, 2010; Dinh et al., 2014; Rezende & Mohamed, 2015). NF consist in training a neural network to map a simple prior distribution onto the desired target through a chain of invertible transformations. They come with interesting characteristics, such as stability and correctness, see for example (Papamakarios et al., 2022). The main limitation comes from the design of an invertible function for the mapping. In particular, computing the Jacobian determinant in the change of variable formula may be costly. Furthermore, while explainability is now under a growing concern within the community (Gilpin et al., 2018), and in particular regarding applications in natural science, the transformation learned by NF models is commonly hard to interpret.

Exploiting the Newtonian evolution in classical Mechanics, Neural Hamiltonian Flows (NHF) (Toth et al., 2020) are NF models that use Hamiltonian transformations. The Hamiltonian of a system is composed of a kinetic and potential energy terms, which sets its dynamical evolution, which is reversible and has a Jacobian determinant equal to one. They come with performance similar to the ones obtained with Real-NVPs in sampling 2D distributions (Toth et al., 2020). Being Physics-driven models, they are expected to enhance interpretability. However, the NHF architecture is made of four neural networks black-boxes that render difficult the interpretation of the learned dynamics. Furthermore, even if they have been numerically shown to transfer multimodality from the target distribution to the potential energy in some cases (Toth et al., 2020), this property is not guaranteed. However, the Hamiltonian formalism could

offer a powerful formalism to design more easily interpreted architectures. Thus, some recent works have proposed to exploit the Hamiltonian properties to design interpretable flows that are invariant under symmetrical transformations (Jimenez Rezende et al., 2019).

In this work, we focus on the transfer of the negative logarithm of the target distribution into the learned potential, which should be the case in any physical systems. This leads us to propose a fixed-kinetic version of NHF where the kinetic-energy term in the Hamiltonian is set to its true form in classical mechanics, where momenta follow a Gaussian distribution. We also discuss the impact of the hyperparameters on such transfer and study the overall robustness of NHF towards the choice of the numerical integration scheme and of the prior distributions. In more details, this work provides the following four main contributions:

- We introduce a fixed-kinetic version of NHF that, thanks to the Hamiltonian evolution, enhances interpretability of the model and does so at a cheaper computational price.
- We analyze the effect of multiple parameters of the architecture on sampling a 2-dimensional multimodal distribution and show that NHF is robust to the choice of hyper-parameters, especially the fixed-kinetic model.
- We show that the choice of prior has an influence on the learned dynamics. The fixed-kinetic model also allows better robustness to the choice of prior distribution.
- Finally, aside from Generative modeling, flow-based models have been shown to be suited as inference algorithms (Rezende & Mohamed, 2015; Winkler et al., 2019). We test a framework for Bayesian inference using NHF and present numerical experiments for inferring cosmological parameters from astronomical observations. The methodology we propose is inspired by Boltzmann generators (Noé et al., 2019).

The manuscript is organized as follows: in Section 2, we present and review the related works. In Section 3, we describe the theoretical framework and the practical implementation of NHF. In Section 4, we discuss the choice of models for the kinetic energy and introduce a fixed-kinetic version for enhanced interpretability and reduced complexity. Section 5 discusses the choices one can make to maximize expressivity given a fixed computational budget, with tests on a 2D Gaussian mixture and analysis of the impact of the Leapfrog-hyperparameters and model complexity. We also show how the choice of base distribution affects the learned energies and thus the interpretability of the model. Finally, in Section 6, we adapt NHF for Bayesian inference and illustrate our method on a toy model from cosmology.

2. Related works

Generative models. These methods make the assumption that data can be represented by an underlying probability distribution. The aim of generative models is to learn such distribution in order to produce original samples similar to the training dataset. Various architectures have been presented such as Generative Adversarial Networks (Goodfellow et al., 2014) or diffusion networks (Sohl-Dickstein et al., 2015). In this paper, we will focus on Normalizing Flows techniques (Tabak & Vanden-Eijnden, 2010; Dinh et al., 2014; Rezende & Mohamed, 2015) as a way of smoothly transforming a simple prior distribution into the target posterior.

Learning Hamiltonians. Learning Hamiltonian potentials, i.e. physical conserved quantities, is a first step towards a better understanding of the physical processes that have governed the data generation. Multiple architectures have been proposed, such as Hamiltonian Neural Networks (Greydanus et al., 2019) or Hamiltonian Generative Networks (Toth et al., 2020). These methods parameterize the Hamiltonian of the system with neural networks and come with useful properties such as exact reversibility and smoothness. They have inspired applications from domain translation (Menier et al., 2022) to fault-detection in industry (Shen et al., 2023). It is worth mentioning that they can be combined with Markov-Chain Monte Carlo (MCMC) methods, for instance as proposals in the Hamiltonian Monte Carlo (HMC) algorithm (Duane et al., 1987; Dhulipala et al., 2022). In this work, we learn artificial Hamiltonians for sampling and our goal is to extract the negative logarithm of the target distribution into the potential.

Neural Hamiltonian Flows. NHF is a Normalizing Flow architecture derived as a modification from Hamiltonian Generative Networks. As generative models, they have been used to sample from 2D distributions (Toth et al., 2020). The Hamiltonian formalism is also a natural framework for exploiting symmetries inside data and build interpretable flows (Jimenez Rezende et al., 2019). In this paper, we discuss its robustness with respect to hyperparameters and choice of prior distribution. Also, we propose an alternative version of NHF to enhance interpretability while reducing the complexity of the model.

Inference with NHF. Inferring the posterior distribution of parameters conditioned on data is an important problem in science. Traditional MCMC methods (Robert & Casella, 2004) are very popular because they come with guarantees in terms of convergence and many progress have been made regarding their tuning (Homan & Gelman, 2014; Carpenter et al., 2017). NF architectures have also been proposed in this framework (Rezende & Mohamed, 2015; Winkler et al., 2019). Here, we adapt NHF to sampling Bayesian posterior distributions by transforming the prior distribution into the posterior with no access to samples from the target.

Explainable AI. XAI deals with the problem of understanding the decisions made by an Artificial Intelligence (Samek & Müller, 2019). Indeed, complex architectures made of multiple (deep) neural network are often easier to train than to understand. Some solutions involve surrogate techniques (Ribeiro et al., 2016), local perturbations (Ancona et al., 2022) or meta-explanations (Lapuschkin et al., 2019). Including physical prior knowledge into neural networks may be another solution to understand the model (Raissi et al., 2019; Toth et al., 2020). In this work, we build on that idea and try to make the model as explainable as possible by fixing the kinetic energy of the model and thus enforcing a classical Mechanics knowledge into the architecture.

3. Normalizing Flows with Hamiltonian transformations

3.1. Normalizing Flows

Normalizing flows are generative models that map a complex target distribution π onto a known prior distribution π_0 from which it is easy to sample (Papamakarios et al., 2022). This mapping is a series of smooth invertible transformations. Once the model is trained, one can reverse the learned dynamics to generate samples from the target distribution starting from the prior. If $X = \mathcal{T}(Z)$, where $\mathcal{T}$ is a $\mathcal{C}^1$-diffeomorphism and $Z \sim \pi_0$, then the density followed by X reads $m(x) = \pi_0\left(\mathcal{T}^{-1}(x)\right) \times \left|\det J_{\mathcal{T}^{-1}}(x)\right|$, where $\det J_{f^{-1}}(x)$ is the Jacobian determinant of the inverse transformation of f evaluated in x. For more expressivity, it is possible to chain several invertible transformations $x = \mathcal{T}_L \circ ... \circ \mathcal{T}_1(z)$. In this case, the change of variable formula is now: $m(x) = \pi_0\left(\mathcal{T}_1^{-1} \circ ... \circ \mathcal{T}_L^{-1}(x)\right) \times \prod_{k=1}^{L} \left|\det J_{\mathcal{T}_k^{-1}}(x)\right|$. The model parameters to optimize are denoted Θ. The goal is to minimize the Kullback-Leibler divergence between the target distribution π and the model distribution m with respect to Θ, i.e. minimizing:

$$\begin{aligned}\mathcal{L}(\Theta) &= \mathbf{E}_\pi\left[\log \pi(X) - \log m(X;\Theta)\right] \\ &= -\mathbf{E}_\pi\Bigg[\log \pi\left(\mathcal{T}_1^{-1} \circ ... \circ \mathcal{T}_T^{-1}(X;\Theta)\right) \\ &\qquad + \sum_{k=1}^{T} \left|\det J_{\mathcal{T}_k^{-1}}(X)\right|\Bigg] + C.\end{aligned}$$

As we seek to find the minimum of this quantity, C can be discarded. Then, one can use samples $x_1, ..., x_m$ from the target distribution for estimating the above loss via Monte Carlo. The function to minimize becomes:

$$\begin{aligned}L(\Theta) = -\frac{1}{m}\sum_{k=1}^{m}\Bigg[&\log \pi\left(\mathcal{T}_1^{-1} \circ ... \circ \mathcal{T}_L^{-1}(x_k;\Theta)\right) \\ &+ \sum_{k=1}^{T} \left|\det J_{\mathcal{T}_k^{-1}}(x_k)\right|\Bigg].\end{aligned} \tag{1}$$

At this point, transformations are to some extent arbitrary. Any smooth invertible function would be suited but the main computational cost comes from the Jacobian determinants. The first goal is then to reduce this computational cost, and the second one to enhance interpretability, by a proper choice of the transformation, and of its induced inverse.

3.2. Neural Hamiltonian Flows

To alleviate these issues, Neural Hamiltonian Flows (NHF, Toth et al., 2020) is a NF technique that uses a series of Hamiltonian transformations as normalizing flows. In classical Mechanics, a system is fully described by its coordinates $(\mathbf{q}, \mathbf{p})$ in phase-space. From that description, it is possible to define a scalar quantity called a Hamiltonian (Landau & Lifshitz, 1982). It can be seen as the total energy of the system and, in this paper, we make the assumption that it is written as the sum of a potential energy V, solely depending on the generalized positions $\mathbf{q}$, and a kinetic energy K, solely depending on the momenta $\mathbf{p}$. The system evolves in phase-space following Hamilton's equations that read:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial H}{\partial \mathbf{p}}, \ \frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}. \tag{2}$$

Hamiltonian transformations present at least two main advantages that make them suited for normalizing flows:

- they are invertible by construction and inversion is easy by using a classical numerical integrator, i.e. just reversing the speed;
- their Jacobian determinant is equal to 1, removing the necessity to compute such determinant for each transformation.

Numerically, the continuous solution can be approached by a symplectic, invertible and stable integrator as a Leapfrog:

$$\begin{cases} \mathbf{p}_{n+\frac{1}{2}} &= \mathbf{p}_n - \nabla V(\mathbf{q}_n) \times \frac{dt}{2}, \\ \mathbf{q}_{n+1} &= \mathbf{q}_n + \nabla K(\mathbf{p}_{n+\frac{1}{2}}) \times dt, \\ \mathbf{p}_{n+1} &= \mathbf{p}_{n+\frac{1}{2}} - \nabla V(\mathbf{q}_{n+1}) \times \frac{dt}{2}. \end{cases} \tag{3}$$

NHF is trained on a dataset consisting in realizations from the target distribution. In order to simulate a Hamiltonian dynamics, one must extend the position space in which live the samples into the phase space, by adding artificial momenta: this is the role of the Encoder. The dynamics is integrated in phase-space with the Leapfrog integrator (Toth et al., 2020). More precisely, during training, NHF takes batches of $\mathbf{q}_T$ from the training dataset as inputs. For each $\mathbf{q}_T$, one $\mathbf{p}_T$ is drawn from a Gaussian distribution whose mean $\mu(\mathbf{q}_T)$ and standard deviation $\sigma(\mathbf{q}_T)$ depend on the $\mathbf{q}_T$. The resulting point in phase-space is then evolving through a series of L Leapfrog steps with integration timestep dt. The outputs consist in the final position $\mathbf{q}_0$ and momenta $\mathbf{p}_0$, as well as the initial mean $\mu(\mathbf{q}_T)$, standard deviation

$\sigma(\mathbf{q}_T)$ and $\mathbf{p}_T$ that are used in the loss computation. Once trained, one can easily define a sampling function that transforms $\mathbf{q}_0$, $\mathbf{p}_0$ into $\mathbf{q}_T$, by changing the sign of integration timestep and moving the system through the learned dynamics. The architecture is illustrated in Figure 1. Following the previous notations, let us call $f(.|\mathbf{q}_T)$ the density of a normal distribution $\mathcal{N}(\mu(\mathbf{q}_T), \sigma(\mathbf{q}_T)^2)$, and $\mathcal{T}^{-1}$ the backward transformation of phase-space performed by NHF i.e. $\mathcal{T}^{-1}(\mathbf{q}_T, \mathbf{p}_T) = (\mathbf{q}_0, \mathbf{p}_0)$. Also, we denote Π_0 the joint distribution of $\mathbf{q}_0$, $\mathbf{p}_0$. By adding artificial momenta $\mathbf{p}_T$ (Toth et al., 2020), the distribution modeled by our NHF is $m(\mathbf{q}_T) = \int M(\mathbf{q}_T, \mathbf{p}_T) d\mathbf{p}_T = \int \Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T, \mathbf{p}_T)) d\mathbf{p}_T$. This integral is intractable so instead one can maximize the following ELBO:

$$\mathcal{L}(\mathbf{q}_T) = \mathbf{E}_f \left[\log \pi(\mathcal{T}^{-1}(\mathbf{q}_T, \mathbf{p}_T)) - \log f(\mathbf{p}_T|\mathbf{q}_T) \right]. \tag{4}$$

This quantity is approximated via Monte Carlo integration.

Once it has learned the transformation, one can reverse the sign of timesteps and use the same potentials to transform the base distribution into the target distribution.

The first part of the architecture consists in adding artificial momenta to simulate a Hamiltonian dynamics. This is done by the Encoder. Here, μ and σ are approximated by two neural networks. As for the Hamiltonian transformations, they are made by chaining Leapfrog steps. To do so, one must design the potential energy V and the kinetic energy K of the system. In previous work (Toth et al., 2020), authors have proposed to parameterize each potential by a neural network. We will discuss this choice in the following section. For now, let us highlight the fact that the choice of neural network for the energies or within the Encoder is without constraint because the properties needed for normalizing flows are ensured by the use of a Leapfrog scheme. This is an illustration of the model flexibility.

4. Designing the kinetic energy for NHF

MLP-kinetic NHF. If the kinetic energy is chosen to be a MLP (Toth et al., 2020), then the model contains two black-boxes that are not easy to interpret *a priori*, namely the kinetic and potential energies K and V. In particular, when sampling a multimodal distribution from a unimodal prior, one may have some troubles understanding where the transfer of multimodality occurs in the process.

Fixed-kinetic NHF. By fixing the kinetic energy inside NHF, we expect to gain interpretability on the learned flow by forcing the latter to obey some Physics principles. Ideally, one would like to find a way of enforcing these energies to be classical from a classical Physics perspective, i.e. making learned kinetic to be a quadratic form and learned potential to be the negative logarithm of the target distribution (or an approximation). This is an important aspect because we would like to keep track of the transformation dynamics. This way, we can play with the familiar classical mechanics framework and interpret the model more easily. By doing so, both interpretability and maniability (less parameters) are gained through the Fixed-kinetic NHF. In this model, K is no longer a MLP but a fixed function. In practice, one can impose the kinetic energy to be written as:

$$K(\mathbf{p}) = \frac{1}{2} \mathbf{p}^T \mathcal{M} \mathbf{p}, \tag{5}$$

with $\mathcal{M}$ a mass matrix. We impose $\mathcal{M}$ to be a positive symmetric definite matrix, as it is the case in classical mechanics. For instance, in 2D, it comes down to optimizing three real parameters, rather than a complete network. Starting from $(\mathbf{q}_0, \mathbf{p}_0)$ drawn from a unimodal prior distribution and imposing a quadratic kinetic energy significantly reduces the number of possibilities for the potential energy if one wants to recover a multimodal $\mathbf{q}_T$. Let us examine what happens for a single Leapfrog transformation. We have $\nabla K(\mathbf{p}) = \mathcal{M}\mathbf{p}$. Using the first step of the Leapfrog algorithm (3), the final expression for $(\mathbf{q}_1, \mathbf{p}_1)$ reads:

$$\begin{aligned}
\mathbf{q_1} &= \mathbf{q_0} + \mathcal{M} \left(\mathbf{p}_0 - \nabla V(\mathbf{q}_0) \frac{dt}{2} \right) dt \\
\mathbf{p}_1 &= \mathbf{p_0} - \nabla V(\mathbf{q_0}) \frac{dt}{2} \\
&\quad - \nabla V \left(\mathbf{q_0} + \mathcal{M} \left(\mathbf{p}_0 - \nabla V(\mathbf{q}_0) \frac{dt}{2} \right) dt \right) \frac{dt}{2}.
\end{aligned}$$

So, if V is quadratic, then the overall transformation is linear and multimodality cannot be obtained from the original Gaussian measure. Thus, multimodality can be achieved only through the potential V in order to sample from the correct target.

It is possible to create many different versions of NHF by fixing its kinetic energy, which thus leads to analysis of the learned dynamics in light of a different Physics framework. One could, for instance, use a relativistic kinetic energy instead of a classical one. We now numerically show how the classical choice for kinetic energy yields an interpretable potential V and such in a robust manner.

5. Robustness and optimization of NHF

We present the results of numerical experiments for analyzing the robustness of NHF. The experiments consist in sampling a 2D Gaussian mixture with 9 modes (see figure 2). Such toy example, similarly studied in (Toth et al., 2020), is interesting in the sense that it will allow us to discuss various aspects, from memory usage to interpretability. Also, traditional generative models like GAN may suffer from mode-collapse problems even in simple multimodal 2D settings (Eghbal-zadeh et al., 2019). It should be noted that we

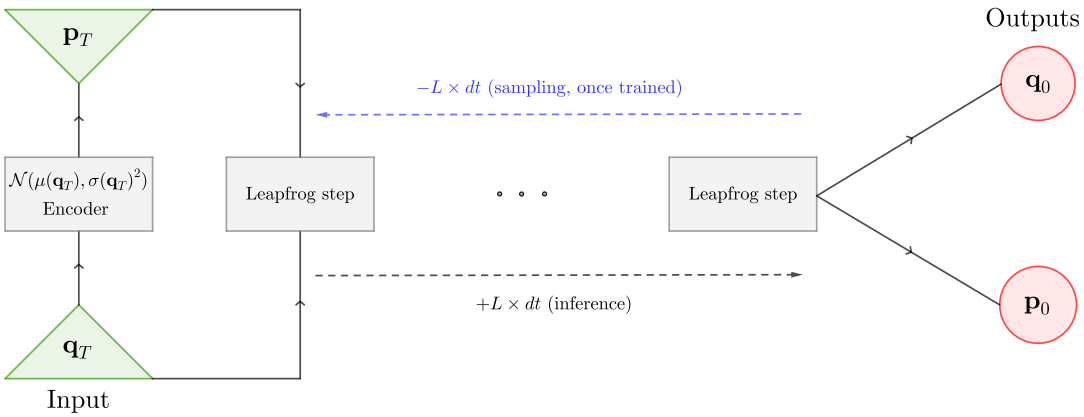

Figure 1. NHF architecture

never observed such issue with NHF in all the experiments that have been conducted.

In each model, μ and σ are MLPs with size (D, H, H, D). The energies are MLPs with size $(D, H, H, 1)$, D being the dimension of data, except for the fixed-kinetic version of NHF where K is a fixed quadratic form. Models are trained on a 5,000 points dataset with minibatches of size 512. Weights and biases are optimized with Adam (Kingma & Ba, 2015), setting the learning rate to 5×10^{-4}.

5.1. Impact of Leapfrog-hyperparameters and model complexity

We first discuss the effect of Leapfrog-hyperparameters L (number of Leapfrog steps) and $T = L \times dt$ (integration time) on the optimization, but also the impact of the model complexity. The latter is governed by the total number of neurons in the model, this number being an increasing function of H, the number of neurons per hidden layer in each MLP of the model. If the model is complex enough, we expect it to learn how to adjust to the number of Leapfrog steps and choice of integration time. If not, we expect the model to perform better by increasing the number of computations, i.e. increasing L. As L increases, we do not expect to see any difference as the number of computations becomes sufficient to solve the problem.

We tested both fixed-kinetic and MLP-kinetic NHF with various choices of L, T and H. We use a soft-uniform prior $\propto s(x+3)s(-x+3)$, where s is the sigmoid function. The corresponding loss decays are illustrated in Figure 3 and additional details can be found in Appendix A.

Firstly, the fixed-kinetic NHF model is shown to be more robust than the MLP-kinetic one to the choices of L and T, at fixed H, as discrepancy in the loss decay more clearly appears only with $H = 8$. Then, regarding the tuning

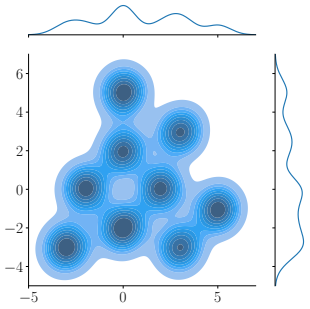

Figure 2. Density estimation of the 2D multimodal target distribution that is used in Section 5 with its marginals. It consists in 9 equally-weighted Gaussians with same covariance matrix $0.5^2 I_2$.

of the Leapfrog scheme, at fixed-integration time, models with $L = 1$ always reach higher final value of the loss, this effect being less visible with the fixed-kinetic model. Increasing the number of leapfrog steps leads to better final performance of the model even if this effect disappears once the number of Leapfrog steps gets sufficient and no further expressivity can be achieved. Finally, as for the effect of integration time T, it barely appears for the fixed-kinetic model, showing that the latter efficiently adjusts to this parameter. As for the MLP-kinetic model, the effect of the integration time is clearer, but mostly at $H = 8$, where performance improves for $T = 1, 10$ compared to $T = 0.1$. Overall, as the number of parameters in the model is increasing ($H = 32$ or 128), the impact of the choice of the integration time is very limited.

Thus, for a given training dataset, there are basically four hyperparameters that require tuning: three are usual in learning (minibatch size, learning rate and number of neurons per hidden layer, i.e. number of learning parameters of the model) and only one is specific to NHF (minimum number of Leapfrog steps), whose tuning is especially less sensitive

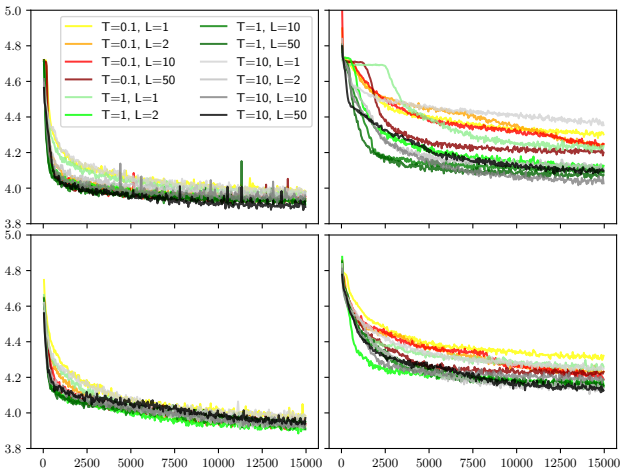

Figure 3. Training loss as a function of epochs for models with different H (number of neurons per hidden layer in each neural network of the model), L (number of Leapfrog steps) and T (integration time). First row: MLP-kinetic NHF; second row: fixed-kinetic model. From left to right: $H = 128,\ 8$. Models were trained for $15,000$ epochs on a 5,000-point dataset with mini-batches of size 512.

when using the fixed-kinetic variant.

5.2. Impact of the prior distribution on the learned dynamics

Now we illustrate the impact of the prior choice on the transfer of characteristics of the target distribution on the potential V, especially regarding the multimodality nature. All models were trained for 15,000 epochs using $H = 128$, $T = 1$ and $L = 10$, with a 5,000 points training dataset.

In terms of sampling, all considered schemes recover the nine correct modes from the target distribution, as illustrated in Figure4. We now consider the learned potential V, as illustrated in Figure5. As the Hamiltonian evolution only involves its derivative, we also include a shifted version in Figure6. When choosing a relatively flat soft-uniform prior distribution that covers the target region, multimodality transfers to the potential energy for both fixed-kinetic and MLP-kinetic NHF. The potential exhibits indeed local extrema centered at the modes of the target, which can either be minima or maxima for the MLP-kinetic NHF but are minima for the fixed-kinetic one. Indeed, with a MLP-kinetic model, the orientation of the learned energies may change from one numerical experiment to another, as we do not enforce the positiveness of the output of V and K making this variant of NHF indifferent to the sign of the learned energies. Finally, similar results were obtained using a Gaussian base $\mathcal{N}(0, 2.5^2 I_2)$ with variance large enough to cover the support of the target distribution, which stresses the impact of the spatial expansion rather than the nature of the prior distribution.

On the other hand, when choosing a "peaked" prior distribution $\mathcal{N}(0, I_2)$, we remark that, for the MLP-kinetic NHF, the momenta $\mathbf{p}_T$ generated by the Encoder inherit from the multimodality of the target distribution, with the same number of modes. As a consequence, learned energies are different from the classical Physics ones. Learned potential energy is different from one model to another but in the case of fixed-kinetic NHF, multimodality is always transferred to the potential energy, showing the robustness of the model to the choice of prior distribution.

Thus, using a fixed-kinetic model allows to more robustly transfer important properties of the target distribution into the learned potential. For both models though, choosing a wide prior that covers the support of the target distribution is improving such transfer. Finally, when the learned potential is not multimodal, it is an indication that multimodality has been transferred instead to the artificial momenta $\mathbf{p}_T$ generated by the Encoder.

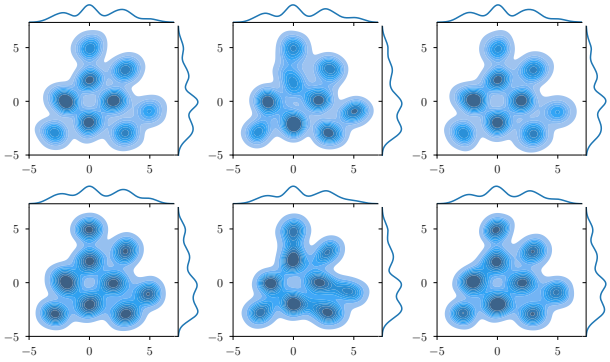

Figure 4. Density estimation of the samples produced by the six models previously defined. First row: MLP-kinetic NHF; second row: Fixed-kinetic model. From left to right: wide soft-uniform prior; peaked Gaussian prior; wide Gaussian prior. Models were trained for $15,000$ epochs on a 5,000-point dataset with mini-batches of size 512, $H = 128$, $L = 10$ and $T = 1$.

6. Adapting NHF for Bayesian inference

6.1. Methodology, derivation of the new loss function

We present how NHF can be used to perform Bayesian inference. It consists in using Hamiltonian flows to transform the prior distribution, in the sense of Bayes' theorem, π_0 of some vector of parameters $\mathbf{q}$ into the target posterior distribution $\pi(\mathbf{q}|\boldsymbol{d})$ of these parameters, knowing some data $\boldsymbol{d}$ and likelihood ℓ. The main difference with the above described NHF lies in the loss inspired from the KL phase in Boltzmann Generators (Noé et al., 2019), as well as in the learning procedure. During training, this NHF takes batches of $\mathbf{q}_0$ from the prior distribution as inputs. For each $\mathbf{q}_0$, one

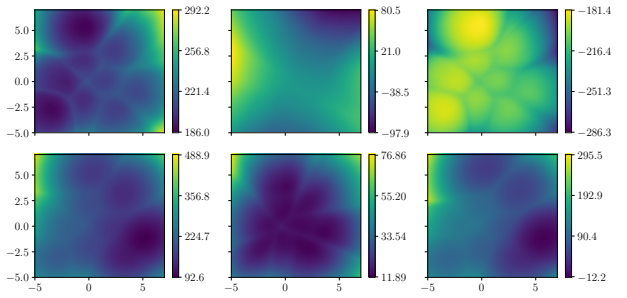

Figure 5. Potential energies learned by the six models previously defined. First row: MLP-kinetic NHF; second row: Fixed-kinetic model. From left to right: wide soft-uniform prior; peaked Gaussian prior; wide Gaussian prior. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512, $H = 128$, $L = 10$ and $T = 1$.

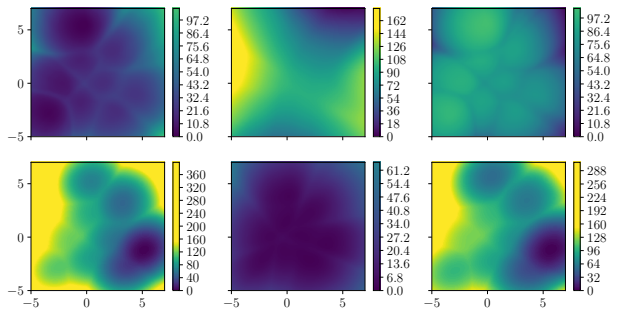

Figure 6. Shifted potential energies learned by the six models previously defined. First row: MLP-kinetic NHF; second row: Fixed-kinetic model. From left to right: wide soft-uniform prior; peaked Gaussian prior; wide Gaussian prior. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512, $H = 128$, $L = 10$ and $T = 1$.

$\mathbf{p}_0$ is drawn from a Gaussian distribution whose mean and standard deviation depend on the $\mathbf{q}_0$. The resulting point in phase-space is then evolved through a series of L Leapfrog steps with integration time dt. The outputs consist in the final positions $\mathbf{q}_T$ and momenta $\mathbf{p}_T$, as well as the initial mean $\mu(\mathbf{q}_0)$, standard deviation $\sigma(\mathbf{q}_0)$ and $\mathbf{p}_0$. All these outputs, as well as the data $\boldsymbol{d}$, are used in the loss computation. Once trained, it is able to transform the prior into the desired posterior distribution of the parameters. Thus, both training and sampling are now made following the forward-direction flow from the prior to the posterior.

In order to compute the loss, one needs to have access to the likelihood distribution ℓ of the model, which encapsulates the covariance matrix of the data as well as the underlying physical mapping between vectors of parameters and the corresponding data. Then, in the framework of Hamiltonian dynamics, the full system is made of both positions (the parameters of interest) and artificial momenta. Let us call $\mathbf{q}_0, \mathbf{p}_0$ the initial position and

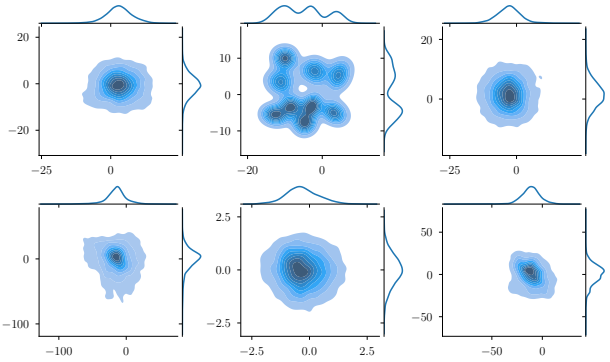

Figure 7. Density estimation of the artificial momenta drawn by the Encoder for the six models previously defined. First row: MLP-kinetic NHF; second row: Fixed-kinetic model. From left to right: wide soft-uniform prior; peaked Gaussian prior; wide Gaussian prior. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512, $H = 128$, $L = 10$ and $T = 1$.

momentum, respectively, and $\mathbf{q}_T, \mathbf{p}_T$ the corresponding final position and momentum, respectively, obtained after L Leapfrog transformations $\mathcal{T}_1^{dt}, ..., \mathcal{T}_L^{dt}$ with timestep dt, i.e: $(\mathbf{q}_T, \mathbf{p}_T) = \mathcal{T}_L^{dt} \circ ... \circ \mathcal{T}_1^{dt}(\mathbf{q}_0, \mathbf{p}_0) := \mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)$. Also, we introduce the notations for the projections along the final positions and momenta, i.e. $\mathbf{q}_T := \mathcal{T}_q(\mathbf{q}_0, \mathbf{p}_0)$ and $\mathbf{p}_T := \mathcal{T}_p(\mathbf{q}_0, \mathbf{p}_0)$. By the change of variables formula and the fact that the Jacobian determinant of each Hamiltonian transformation is one, the model joint distribution M is written as:

$$\begin{aligned} M(\mathbf{q}_T, \mathbf{p}_T) &= 1 \times \Pi_0(\mathcal{T}_1^{-dt} \circ ... \circ \mathcal{T}_L^{-dt}(\mathbf{q}_T, \mathbf{p}_T)) \\ &= \Pi_0(\mathbf{q}_0, \mathbf{p}_0) = \pi_0(\mathbf{q}_0) \times f(\mathbf{p}_0|\mathbf{q}_0), \end{aligned}$$

where Π_0 is the joint prior distribution, π_0 the prior distribution of the parameters of interest and f the Gaussian distribution of the Encoder. We fix the target density of the final momenta $g(\mathbf{p})$ (e.g. Gaussian). We now seek to minimize the KL-divergence between the model joint distribution and the desired target joint distribution conditioned on data $\Pi(\mathbf{q}, \mathbf{p}|\boldsymbol{d}) = \pi(\mathbf{q}|\boldsymbol{d})g(\mathbf{p})$. We write the latter as the product of a density depending on $\mathbf{q}$ and an other density depending on $\mathbf{p}$ so that the two are independent. Using Bayes' theorem, we have (see Appendix B for more details):

$$\begin{aligned} &D_{KL}(M(\mathbf{q}_T, \mathbf{p}_T) \,||\, \pi(\mathbf{q}_T|\boldsymbol{d})g(\mathbf{p}_T)) = \\ &\int \pi_0(\mathbf{q}_0, \mathbf{p}_0)\big[\log \pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0) - \log \pi_0(\mathcal{T}_q(\mathbf{q}_0, \mathbf{p}_0)) \\ &\quad - \log \ell(\boldsymbol{d}|\mathcal{T}_q(\mathbf{q}_0, \mathbf{p}_0)) - \log g(\mathcal{T}_p(\mathbf{q}_0, \mathbf{p}_0))\big] d\mathbf{q}_0 d\mathbf{p}_0 + \text{cst}. \end{aligned} \tag{6}$$

6.2. Application to cosmology

We apply the herein-above architecture to cosmological analysis. One typical case is the determination of the cosmic expansion, and more generally of the cosmological param-

eters, from the observation of brightness and recession velocity of Type Ia supernovæ (e.g. Riess et al., 1998; Betoule et al., 2014). We use it as a toy example, however, while the model used so far has been simple, it may be expanded in very complicated direction for which sampling from the probability distribution becomes very complex. New observatory are presently being built which is expected to deliver tens of thousands of new supernovæ Ia over the next decade (LSST Science Collaboration, 2009). We now present the essential feature of the model that we need before applying our method to the toy example.

According to the Λ-CDM model, the relation between the distance and the brightness of Type Ia supernovae is of great interest because it depends on two cosmological parameters: the matter density parameter Ω_m and the adimensional Hubble parameter h. To be more specific, database of type Ia supernovae report the distance modulus μ. This quantity is defined as the difference between the apparent and the absolute magnitude of an astronomical object, and is directly related to luminosity distance (Weinberg, 1972) and thus a function of the redshift z, Ω_m and h:

$$\mu(z, \Omega_m, h) = 5 \log_{10} \left(\frac{D_L^*(z, \Omega_m)}{h 10\text{pc}} \right)$$

where $D_L^*(z, \Omega_m) = \frac{c(1+z)}{H_0} \int_0^z \frac{ds}{\sqrt{1-\Omega_m+\Omega_m(1+s)^3}}$, and $H_0 = 100$ km s^{-1} Mpc^{-1}, c being the speed of light. In practice, we avoid computing the integral in D_L^* by using an approximation (Pen, 1999) which is only valid for a flat Universe:

$$D_L^*(z, \Omega_m) = \frac{c(1+z)}{H_0} \left[\eta(1, \Omega_m) - \eta\left(\frac{1}{1+z}, \Omega_m \right) \right],$$

with

$$\eta(a, \Omega_m) = 2\sqrt{1+s^3} \left(\frac{1}{a} - 0.1540 \frac{s}{a^3} + 0.4304 \frac{s^2}{a^2} + 0.19097 \frac{s^3}{a} + 0.066941 s^4 \right).$$

The formal definition of these quantities imposes constraints on the possible values of the parameters, that can only be comprised between zero and one. We avoid the problem by outputting a sigmoid of $\mathbf{q}_T$.

We aim to sample from the posterior distribution $\pi(\Omega_m, h|\text{data})$ quantifying the probability that we are living in a universe whose mean density and expansion is equal to Ω_m and h given D observations data $= \{z_i, \mu_i\}_{1 \leq i \leq D}$ of type Ia supernovae, and the covariance matrix C of the observed distance moduli. The likelihood of the problem is supposed to be Gaussian, i.e. the observed data and the simulated output from parameters are different up to a Gaussian noise. The final momenta distribution g is set to a Normal distribution. The cumulative plots in figure 8 compare the performance of a fixed- and MLP-kinetic NHF with a HMC. They are both biased which is expected since they minimize the KL-divergence between the model distribution and the target, but less than with an ELBO. We leave for future work a possible correction using importance sampling methods at the end of training.

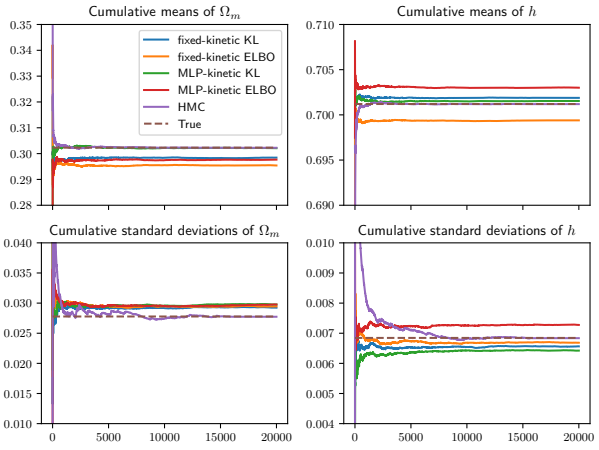

Figure 8. Cumulative plots of means and standard deviations of both Ω_m and h produced by trained NHF models and a HMC on a 20,000-sample dataset, compared to the ground truth. Soft-uniform prior, 30,000 training epochs, g chosen to be a $\mathcal{N}(0, I_2)$

7. Conclusion

In this work, we analyzed how well Hamiltonian Normalizing Flows are suited to solving some issues in Generative modeling. The main advantage of these methods is twofold. First, the volume-preservation in phase-space avoids the costly computation of Jacobian determinants. Then they allow for flexibility in the type of neural networks that is used since reversibility is ensured by the symplectic integrator. It is this flexibility that allows us to propose a NHF variant based on classical kinetic energy. By exploring 2D sampling problem, we showed that both variants are robust to the choice of hyperparameters and prior distribution. In particular, we illustrate how the explicit classical design of the kinetic energy is a way to increase such robustness and make the model more easily interpretable. Finally, we explained how to adapt NHF to the context of Bayesian inference to obtain a sampler of the posterior distribution. Further work could then address how the offsets could be corrected by importance sampling techniques on a trained model. More generally, a next step would be testing such architectures on more complex problems such as image generation to attest their behavior in a high-dimensional setting.

Acknowledgments

All the authors are grateful to the support from CNRS through the MITI-Prime 80 project "CosmoBayes". All the authors thank the Mésocentre Clermont Auvergne University (https://mesocentre.uca.fr/) and the HPC cluster at Fysikum (it.fysik.su.se/hpc/) for providing help, computing and storage resources. This work was supported by the French ANR under the grant ANR-20-CE46-0007 (*SuSa* project) and by the Simons Collaboration on "Learning the Universe". JJ acknowledges support by the Swedish Research Council (VR) under the project 2020-05143 – "Deciphering the Dynamics of Cosmic Structure".

We acknowledge the use of the following packages in this analysis: Numpy (Harris et al., 2020), IPython (Pérez & Granger, 2007), Matplotlib (Hunter, 2007), Pytorch (Paszke et al., 2019), SciPy (Jones et al., 2001–).

References

Abell, P. A., Allison, J., Anderson, S. F., Andrew, J. R., Angel, J. R. P., Armus, L., Arnett, D., Asztalos, S. J., et al. LSST Science Book, Version 2.0. arXiv e-prints, art. arXiv:0912.0201, December 2009.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Gradient-Based Attribution Methods, pp. 169–191. Springer-Verlag, Berlin, Heidelberg, 2022. ISBN 978-3-030-28953-9. URL https://doi.org/10.1007/978-3-030-28954-6_9.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. In International Conference on Learning Representations, 2017. URL https://openreview.net/forum?id=Hk4_qw5xe.

Berard, H., Gidel, G., Almahairi, A., Vincent, P., and Lacoste-Julien, S. A closer look at the optimization landscapes of generative adversarial networks. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=HJeVnCEKwH.

Betoule, M., Kessler, R., Guy, J., Mosher, J., Hardin, D., Biswas, R., Astier, P., El-Hage, P., Konig, M., Kuhlmann, S., Marriner, J., et al. Improved cosmological constraints from a joint analysis of the SDSS-II and SNLS supernova samples. Astronomy & Astrophysics, 568:A22, aug 2014. doi: 10.1051/0004-6361/201423413. URL https://doi.org/10.1051%2F0004-6361%2F201423413.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. Stan: A probabilistic programming language. Journal of statistical software, 76(1), 2017.

Dhulipala, S. L. N., Che, Y., and Shields, M. D. Bayesian inference with latent hamiltonian neural networks, 2022. URL https://arxiv.org/abs/2208.06120.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. 10 2014.

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. Hybrid monte carlo. Physics Letters B, 195(2):216–222, 1987. ISSN 0370-2693. doi: https://doi.org/10.1016/0370-2693(87)91197-X. URL https://www.sciencedirect.com/science/article/pii/037026938791197X.

Eghbal-zadeh, H., Zellinger, W., and Widmer, G. Mixture density generative adversarial networks. Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 06 2019. doi: 10.1109/CVPR.2019.00597.

Frazer, J., Notin, P., Dias, M., Gomez, A., Min, J. K., Brock, K., Gal, Y., and Marks, D. S. Disease variant prediction with deep generative models of evolutionary data. Nature, 599(7883):91–95, 2021.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. Explaining explanations: An overview of interpretability of machine learning. In 2018 IEEE 5th International Conference on data science and advanced analytics (DSAA), pp. 80–89. IEEE, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), Advances in Neural Information Processing Systems, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian Neural Networks. Curran Associates Inc., Red Hook, NY, USA, 2019.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'ıo, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy.

Nature, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Homan, M. D. and Gelman, A. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. J. Mach. Learn. Res., 15(1):1593–1623, jan 2014. ISSN 1532-4435.

Hunter, J. D. Matplotlib: A 2d graphics environment. Computing In Science & Engineering, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

Jimenez Rezende, D., Racanière, S., Higgins, I., and Toth, P. Equivariant Hamiltonian Flows. arXiv e-prints, art. arXiv:1909.13739, September 2019.

Jones, E., Oliphant, T., Peterson, P., et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL http://arxiv.org/abs/1412.6980.

Landau, L. and Lifshitz, E. Mechanics: Volume 1. Number vol. 1. Elsevier Science, 1982. ISBN 9780080503479. URL https://books.google.fr/books?id=bE-9tUH2J2wC.

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K.-R. Unmasking clever hans predictors and assessing what machines really learn. Nature communications, 10(1):1096–1096, 2019. ISSN 2041-1723.

Lin, Z., Khetan, A., Fanti, G., and Oh, S. Pacgan: The power of two samples in generative adversarial networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), Advances in Neural Information Processing Systems, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf.

Lopez, R., Gayoso, A., and Yosef, N. Enhancing scientific discoveries in molecular biology with deep generative models. Molecular Systems Biology, 16(9):e9198, 2020.

Menier, E., Bucci, M. A., Yagoubi, M., Mathelin, L., and Schoenauer, M. Continuous Methods : Hamiltonian Domain Translation. arXiv e-prints, art. arXiv:2207.03843, July 2022.

Noé, F., Olsson, S., Köhler, J., and Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. Science, 365(6457):eaaw1147, 2019. doi: 10.1126/science.aaw1147. URL https://www.science.org/doi/abs/10.1126/science.aaw1147.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. J. Mach. Learn. Res., 22(1), jul 2022. ISSN 1532-4435.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pen, U.-L. Analytical fit to the luminosity distance for flat cosmologies with a cosmological constant. The Astrophysical Journal Supplement Series, 120(1):49–50, jan 1999. doi: 10.1086/313167. URL https://doi.org/10.1086%2F313167.

Pérez, F. and Granger, B. E. IPython: a system for interactive scientific computing. Computing in Science and Engineering, 9(3):21–29, May 2007. ISSN 1521-9615. doi: 10.1109/MCSE.2007.53. URL https://ipython.org.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational physics, 378:686–707, 2019.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In Bach, F. and Blei, D. (eds.), Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/rezende15.html.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?": Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939778. URL

https://doi.org/10.1145/2939672.2939778.

Riess, A. G., Filippenko, A. V., Challis, P., Clocchiatti, A., Diercks, A., Garnavich, P. M., Gilliland, R. L., Hogan, C. J., Jha, S., Kirshner, R. P., Leibundgut, B., Phillips, M. M., Reiss, D., Schmidt, B. P., Schommer, R. A., Smith, R. C., Spyromilio, J., Stubbs, C., Suntzeff, N. B., and Tonry, J. Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. The Astronomical Journal, 116(3):1009–1038, September 1998. doi: 10.1086/30049910.48550/arXiv.astro-ph/9805201.

Robert, C. P. and Casella, G. Monte Carlo Statistical Methods. Springer, 2nd edition, 2004. ISBN 0387212396.

Rodriguez, A. C., Kacprzak, T., Lucchi, A., Amara, A., Sgier, R., Fluri, J., Hofmann, T., and Réfrégier, A. Fast cosmic web simulations with generative adversarial networks. Computational Astrophysics and Cosmology, 5 (1):1–11, 2018.

Samek, W. and Müller, K.-R. Towards explainable artificial intelligence. In Explainable AI: interpreting, explaining and visualizing deep learning, pp. 5–22. Springer, 2019.

Shen, J., Chowdhury, J., Banerjee, S., and Terejanu, G. Machine fault classification using hamiltonian neural networks, 2023. URL https://arxiv.org/abs/2301.02243.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In Bach, F. and Blei, D. (eds.), Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Tabak, E. and Vanden-Eijnden, E. Density estimation by dual ascent of the log-likelihood. Communications in Mathematical Sciences, 8(1):217–233, 2010. ISSN 1539-6746. doi: 10.4310/CMS.2010.v8.n1.a11.

Toth, P., Rezende, D. J., Jaegle, A., Racanière, S., Botev, A., and Higgins, I. Hamiltonian generative networks. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=HJenn6VFvB.

Weinberg, S. Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity. 1972.

Winkler, C., Worrall, D., Hoogeboom, E., and Welling, M. Learning likelihoods with conditional normalizing flows, 2019. URL https://arxiv.org/abs/1912.00042.

A. Additional plots about robustness and optimization of NHF

We present the results of additional experiments with $H = 32$ in figure 9. Removing more than 90% of parameters (passage from $H = 128$ to $H = 32$), the final values are always higher by less than 4%, for both models. The different final values of the loss function can be represented on scatter plots, see Figure 10. The latter clearly illustrate the robustness of the fixed-kinetic model, see Figure 9. It also shows that models with $L = 1$ perform poorer than with $L = 2,\ 10,\ 50$.

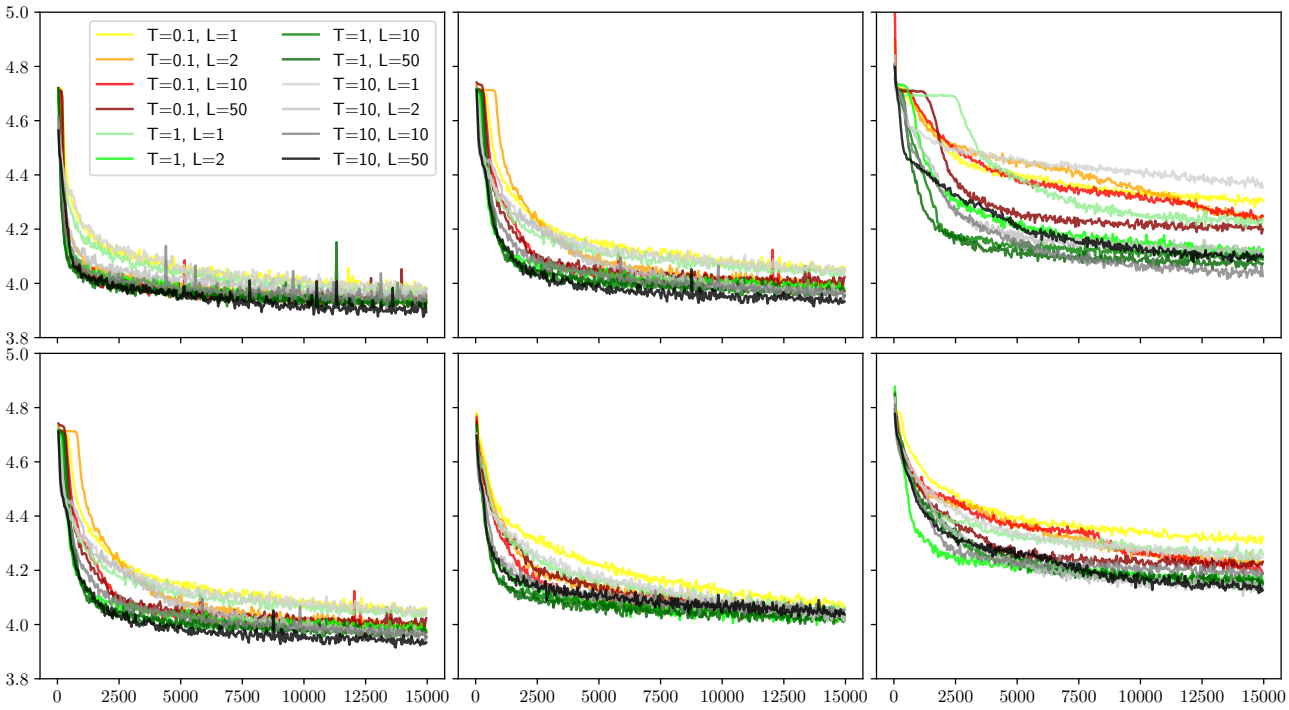

Figure 9. Training loss as a function of epochs for models with different H (number of neurons per hidden layer in each neural network of the model), L (number of Leapfrog steps) and T (integration time). First row: MLP-kinetic NHF; second row: fixed-kinetic model. From left to right: $H = 128,\ 32,\ 8$. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512.

B. Derivation of the KL-divergence and ELBO for the inference problem

The KL-divergence suited to the inference problem is derived as follows:

$$
\begin{aligned}
&D_{KL}(M(\mathbf{q}_T,\mathbf{p}_T)\,||\,\pi(\mathbf{q}_T|\boldsymbol{d})g(\mathbf{p}_T)) = \int M(\mathbf{q}_T,\mathbf{p}_T)\log M(\mathbf{q}_T,\mathbf{p}_T)d\mathbf{q}_T d\mathbf{p}_T - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi(\mathbf{q}_T|\boldsymbol{d}) + \log g(\mathbf{p}_T)\right] d\mathbf{q}_T d\mathbf{p}_T \\
&= \int \Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T,\mathbf{p}_T))\log\Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T,\mathbf{p}_T))d\mathbf{q}_T d\mathbf{p}_T - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi_0(\mathbf{q}_T) + \log\ell(\boldsymbol{d}|\mathbf{q}_T) - \log p(\boldsymbol{d}) + \log g(\mathbf{p}_T)\right] d\mathbf{q}_T d\mathbf{p}_T \\
&= \int \Pi_0(\mathbf{q}_0,\mathbf{p}_0)\left[\log\pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0)\right] d\mathbf{q}_0 d\mathbf{p}_0 - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi_0(\mathbf{q}_T) + \log\ell(\boldsymbol{d}|\mathbf{q}_T) + \log g(\mathbf{p})\right] d\mathbf{q}_T d\mathbf{p}_T + \text{cst} \\
&= \int \Pi_0(\mathbf{q}_0,\mathbf{p}_0)\left[\log\pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0) - \log\pi_0(\mathcal{T}_q(\mathbf{q}_0,\mathbf{p}_0)) - \log\ell(\boldsymbol{d}|\mathcal{T}_q(\mathbf{q}_0,\mathbf{p}_0)) - \log g(\mathcal{T}_p(\mathbf{q}_0,\mathbf{p}_0)\right] d\mathbf{q}_0 d\mathbf{p}_0 + \text{cst}.
\end{aligned}
\tag{7}
$$

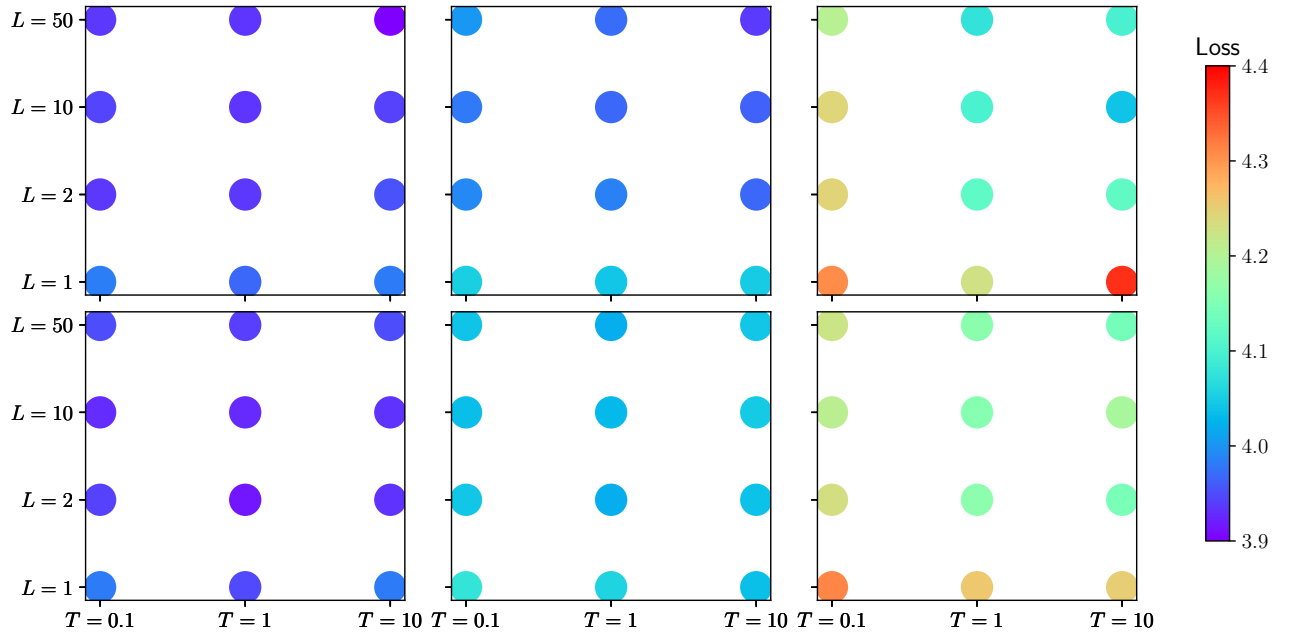

Figure 10. Scatter plots L vs. T of final values of the loss averaged on last 500 epochs. First row: MLP-kinetic NHF; second row: fixed-kinetic model. From left to right: $H = 128,\ 32,\ 8$. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512.

We can also adapt the ELBO from (Toth et al., 2020) to our inference framework:

$$\begin{aligned}
\ln \pi_0(\mathbf{q}_0) &= \ln \int \Pi_0(\mathbf{q}_0, \mathbf{p}_0) d\mathbf{p}_0 \\
&= \ln \int \frac{\Pi_0(\mathbf{q}_0, \mathbf{p}_0)}{f(\mathbf{p}_0|\mathbf{q}_0)} f(\mathbf{p}_0|\mathbf{q}_0) d\mathbf{p}_0 \\
&= \ln E_f \left[\frac{\Pi_0(\mathbf{q}_0, \mathbf{p}_0)}{f(\mathbf{p}_0|\mathbf{q}_0)} \right] \\
&\geq E_f \left[\ln \Pi_0(\mathbf{q}_0, \mathbf{p}_0) - \ln f(\mathbf{p}_0|\mathbf{q}_0) \right] \\
&= E_f \left[\ln M(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)) - \ln f(\mathbf{p}_0|\mathbf{q}_0) \right]
\end{aligned}$$

Then, expliciting $M(q, p) = \pi_0(q)\ell(d|q)g(p)$:

$$\text{ELBO}(\mathbf{q}_0) = E_f \left[\ln \left[\pi_0(T_q(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)))\ell(d|T_q(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)))g(T_p(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0))) \right] - \ln f(\mathbf{p}_0|\mathbf{q}_0) \right] \tag{8}$$