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

Vincent Souveton
LMBP, CNRS,
Université Clermont-Auvergne

Arnaud Guillin
LMBP, IUF,
Université Clermont-Auvergne

Jens Jasche
Stockholm University

Guilhem Lavaux
IAP, CNRS, Sorbonne Université

Manon Michel
LMBP, CNRS, Université Clermont-Auvergne

Abstract

Normalizing Flows (NF) are Generative models which transform a simple prior distribution into the desired target. They however require the design of an invertible mapping whose Jacobian determinant has to be computable. Recently introduced, Neural Hamiltonian Flows (NHF) are 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. In this paper, we show that the current NHF architecture may still pose a challenge to interpretability. Inspired by Physics, we introduce a fixed-kinetic energy version of the model. This approach improves interpretability and robustness while requiring fewer parameters than the original model. We illustrate that on a 2D Gaussian mixture and on the MNIST and Fashion-MNIST datasets. Finally, we show how to adapt NHF to the context of Bayesian inference and illustrate the method on an example from cosmology.

1 INTRODUCTION

Generative models are now under growing interest regarding 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 and Bottou, 2017; Berard et al., 2020). More robust techniques like Normalizing Flows (NF) were thus developed (Tabak and Vanden-Eijnden, 2010; Dinh et al., 2014; Rezende and 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. Also, explainability is now a growing concern within the community (Gilpin et al., 2018), in particular regarding applications in natural science, as the transformation learned by NF models is commonly hard to interpret.

First motivated by the mitigation of the Jacobian computation limitation, Neural Hamiltonian Flows (NHF) (Toth et al., 2020) are NF models that use Hamiltonian transformations. Indeed, in classical Newtonian mechanics, the Hamiltonian of a system, composed of a kinetic and a potential energy terms, sets its dynamical evolution, which is reversible and has a Jacobian determinant equal to one. They have exhibited performance similar to RealNVPs in sampling some 2D distributions (Toth et al., 2020). Furthermore, being Physics-driven models, they are expected to enhance interpretability and it is furthermore straightforward to exploit the Hamiltonian properties to include some invariance under symmetrical transformations (Jimenez

Proceedings of the 27$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

Rezende et al., 2019). However, the NHF architecture is made of four neural networks black-boxes that may render difficult the interpretation of the learned dynamics and energies. In particular, the learnt potential energy is not guaranteed to correspond to the corresponding physical potential energy of the data, when writing their probability distribution as a Boltzmann one. Even if it was numerically shown to transfer multimodality from the target distribution to the potential energy in some 2D cases (Toth et al., 2020), this property is not ensured.

We focus here precisely on the transfer of the negative logarithm of the target distribution into the learned potential, which should be the case in any physical system. This leads to propose a fixed-kinetic version of NHF (FK-NHF), where momenta follow a Gaussian distribution, i.e. its classical mechanics formulation. We discuss the impact of the hyperparameters on such transfer and study its overall robustness towards the choice of the numerical integration scheme and of the prior distributions. This work provides the following five main contributions:

- We introduce a FK-NHF which, thanks to the Hamiltonian evolution, enhances the interpretability of the model at a cheaper computational price, compared to the standard multilayer perceptron version (MLPK-NHF).
- We analyze the effect of multiple parameters of the architecture on sampling a 2D multimodal distribution and show that NHF is robust to the hyper-parameters choice, especially FK-NHF.
- We show that the choice of prior has an influence on the learned dynamics. FK-NHF allows better robustness to the choice of prior distribution.
- We evaluate the sampling performance of NHF models on high-dimensional image generation problems and compare them to a classical RealNVP.
- Finally, aside from Generative modeling, flow-based models are relevant for inference (Rezende and 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: Section 2 presents and reviews 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 FK-NHF for enhanced interpretability and reduced complexity. Section 5 discusses the maximization of 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. In Section 6, we discuss the generative performance of NHF models in high-dimensional problems. Finally, in Section 7, we adapt NHF for Bayesian inference and illustrate our method on a standard model from cosmology.

2 RELATED WORKS

Generative models. Various architectures have been presented such as Generative Adversarial Networks (Goodfellow et al., 2014) or diffusion networks (Sohl-Dickstein et al., 2015). Here, we will focus on Normalizing Flows techniques (Tabak and Vanden-Eijnden, 2010; Dinh et al., 2014; Rezende and Mohamed, 2015) as a way of smoothly transforming a simple prior distribution into the target posterior. The FK-NHF we present can however be understood as some ODE counterpart of diffusion models where the transformation is governed by a Langevin SDE. Both transformations agree for one discrete time step and this situation is reminiscent of the one in sampling with HMC (Neal, 2012) and Metropolis-adjusted Langevin algorithm (Rossky et al., 1978). The main limitation of diffusion models is that they usually require many iterations (~ 1000 (Ho et al., 2020)) to produce good samples, while we show that FK-NHF only needs around 10 leapfrog steps.

Learning Hamiltonians. Learning Hamiltonians, i.e. physical conserved quantities, is a first step towards a better understanding of the physical processes that govern the data generation. Multiple architectures are proposed, such as Hamiltonian Neural Networks (Greydanus et al., 2019) or Hamiltonian Generative Networks (Toth et al., 2020). These methods parameterize the Hamiltonian 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). Notably, 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). We learn here artificial Hamiltonians for sampling and our goal is to extract the negative logarithm of the target distribution into the potential.

Neural ODE Flows. Compared to the transformation in Neural ODE flow (Chen et al., 2018), the Hamiltonian ODE in NHF are volume-preserving, making for

a cheaper log-likelihood computation, and can be integrated via symplectic integrators. We discuss here its robustness with respect to hyperparameters and choice of prior distribution. Also, we propose the alternative FK version of NHF to enhance interpretability while reducing the complexity of the model.

Inference with NHF. Traditional MCMC methods (Robert and Casella, 2004) are very popular because they come with guarantees in terms of convergence and much progress has been made regarding their tuning (Homan and Gelman, 2014; Carpenter et al., 2017). NF architectures have also been proposed in this framework (Rezende and 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 and Müller, 2019). Indeed, complex architectures made of multiple (deep) neural networks 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 its kinetic energy and thus enforcing classical Mechanics knowledge into the architecture.

3 NORMALIZING FLOWS WITH HAMILTONIAN TRANSFORMATIONS

3.1 Normalizing Flows

Normalizing flows are generative models mapping a complex target distribution π onto a known prior distribution π_0 which is easy to sample (Papamakarios et al., 2022). This mapping is a series of smooth invertible transformations $\mathcal{T}_1, ..., \mathcal{T}_L$. Once the model is trained, one can reverse the learned dynamics to generate samples from the target distribution starting from the prior. If $X = \mathcal{T}_L \circ ... \circ \mathcal{T}_1(Z)$, where $Z \sim \pi_0$, the density followed by X reads $m(x) = \pi_0\left(\mathcal{T}_1^{-1} \circ ... \circ \mathcal{T}_L^{-1}(x)\right) \times \prod_{k=1}^{L} \left|\det J_{\mathcal{T}_k^{-1}}(x)\right|$. 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:

$$\begin{aligned}\mathcal{L}(\Theta) &= \mathbf{E}_\pi\left[\log \pi(X) - \log m(X;\Theta)\right] \\ &= -\mathbf{E}_\pi\Bigg[\log \pi_0\left(\mathcal{T}_1^{-1} \circ ... \circ \mathcal{T}_L^{-1}(X;\Theta)\right) \\ &\quad + \sum_{k=1}^{L} \left|\det J_{\mathcal{T}_k^{-1}}(X;\Theta)\right|\Bigg] + C.\end{aligned}$$

One can use samples from the target distribution in order to get a Monte Carlo estimation of the above loss and minimize it with gradient descent.

At this point, transformations are to some extent arbitrary. Any smooth invertible transformation is suited but the main computational cost comes from the Jacobian determinants. The first goal then is 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

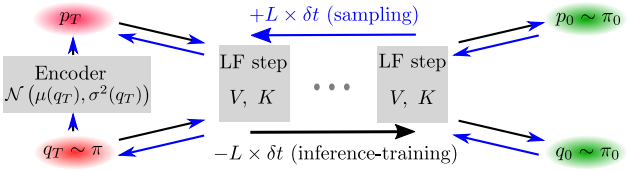

Figure 1: Schematic representation of the NHF architecture. In training, dataset samples are identified as generalized positions and the Encoder generates artificial generalized momenta. The system then evolves in the phase space following a discretized Hamiltonian flow. The resulting output must follow the prior distribution. The target data distribution can be sampled by inverting the learned dynamics.

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 and 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:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial H}{\partial \mathbf{p}}, \quad \frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}. \tag{1}$$

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:

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

An illustration of a NHF model can be found in Figure 1. NHF is trained on a dataset consisting of realizations from the target distribution. To simulate 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 are integrated in phase-space with the Leapfrog integrator. 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 deviation $\sigma(\mathbf{q}_T)$ depend on the $\mathbf{q}_T$. The resulting point in phase-space then evolves through a series of L Leapfrog steps with integration timestep $-\delta t$. The outputs consist in the final position $\mathbf{q}_0$ and momenta $\mathbf{p}_0$, as well as the initial mean $\mu(\mathbf{q}_T)$, deviation $\sigma(\mathbf{q}_T)$ and $\mathbf{p}_T$ 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. An illustration of the architecture can be found in Figure 1.

Now regarding the training, following the previous notations, let $f(.|\mathbf{q}_T)$ be 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. $\mathcal{T}^{-1}(\mathbf{q}_T, \mathbf{p}_T) = (\mathbf{q}_0, \mathbf{p}_0)$. 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 the 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 being intractable, one maximizes the following ELBO:

$$\mathcal{L}(\mathbf{q}_T) = \mathbf{E}_f \left[\log \Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T, \mathbf{p}_T)) - \log f(\mathbf{p}_T|\mathbf{q}_T) \right]. \tag{3}$$

This quantity is approximated via Monte Carlo integration. Having learned the transformation, one can reverse the sign of timesteps and use the same potentials to transform the prior distribution into the target distribution.

To summarize, the first part of the architecture consists of adding artificial momenta, as done by the Encoder, to simulate Hamiltonian dynamics. 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 Toth et al. (2020), each energy term is parameterized by a neural network. We will discuss this choice in the following section. For now, let us highlight that integrating Hamilton's equations with a symplectic numerical scheme provides flexibility. Most NF architectures rely on a careful architecture design rendering the computation of the Jacobian determinant easy (Dinh et al., 2017). This is not the case with NHF since invertibility and volume-conservation are ensured by the use of a Leapfrog integrator and do not depend on the neural networks that are used to parameterize μ, σ, V and K.

4 DESIGNING THE KINETIC ENERGY FOR NHF

MLPK-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 kinetic and potential energies K and V. In particular, when sampling a multimodal distribution from an unimodal prior, the learnt potential V may not reflect the multimodal distribution.

FK-NHF. By fixing the kinetic energy inside NHF, we gain interpretability on the learned flow by forcing the latter to obey some Physics principles. In this model, K is no longer a MLP but a quadratic function $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^T\mathcal{M}^{-1}\mathbf{p}$, with $\mathcal{M}$ a symmetric positive matrix. Starting from $(\mathbf{q}_0, \mathbf{p}_0)$ drawn from an unimodal prior distribution and imposing quadratic kinetic energy significantly reduces the possibilities for the potential energy to recover a multimodal $\mathbf{q}_T$. We indeed aim at enforcing these energies to be classical from a Physics perspective, i.e. making the learned kinetic energy to be of a quadratic form and the learned potential to be the negative logarithm of the target distribution $-\log \pi$ (or an approximation), as it is the case for diffusion models. It is noteworthy that, if the learnt potential is $-\log \pi$ and K of a classical form, then any initial distribution of $\mathbf{q}_0$ can be mapped to the distribution π of the $\mathbf{q}_T$ given the distribution of $\mathbf{p}_0$ is rich enough (e.g. a Normal law). Indeed, Hamiltonian dynamics with momenta refreshment yields an ergodic exploration of phase-space that leaves the canonical distribution ($\propto \exp(-\log \pi - K)$) invariant. Canonical distribution invariance uses volume-preservation in phase-space, i.e. Liouville's theorem, and ergodicity

comes from momenta refreshment.

This is also at the basis of the HMC method (Duane et al., 1987; Neal, 2012). Therefore the FK variant can learn any distribution of $\mathbf{q}_T$.

Keeping track of the transformation dynamics, and the analogy with the familiar classical mechanics framework, makes possible the interpretability of the learned potential. Also, learning the energy landscape associated with the target distributions offers guarantees in terms of control of the discretization scheme, avoiding chaotic behaviour and making the model less sensitive to the choice of Leapfrog hyperparameters as we show in Section 5. By doing so, both interpretability, sparsity and robustness are gained through FK-NHF. It is possible to create different versions of NHF by fixing its kinetic energy. 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 INTERPRETABILITY AND ROBUSTNESS

We present the results of numerical experiments for sampling a 2D Gaussian mixture (9 equally-weighted Gaussians with same covariance matrix $0.5^2 I_2$) (see Figure 2). Such an example, similarly studied in Toth et al. (2020), enables us to understand important aspects of NHF, like sensitivity to the choice of hyperparameters and, more importantly, interpretability. Also, traditional generative models like GANs may suffer from mode-collapse problems even in simple multimodal 2D settings (Eghbal-zadeh et al., 2019), mode-collapses which were never observed with NHF experiments. Additional details about the models and hyperparameters choice can be found in Appendix B.1. A public version of the code is available on a Git repository [1].

5.1 Impact of Leapfrog-hyperparameters and model complexity

Let us discuss the effect of Leapfrog-hyperparameters L (number of Leapfrog steps) and $T = L \times \delta t$ (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 N, the number of neurons per hidden layer in each MLP of the model. If the model is complex enough, we expect to learn how to adjust to the number of Leapfrog steps and choice of integration time. If not, the model may have better performance by increasing the number of steps, i.e. increasing L.

[1] https://plmlab.math.cnrs.fr/stoch-algo-phys/generative-models/fixed-kinetic-NHF/

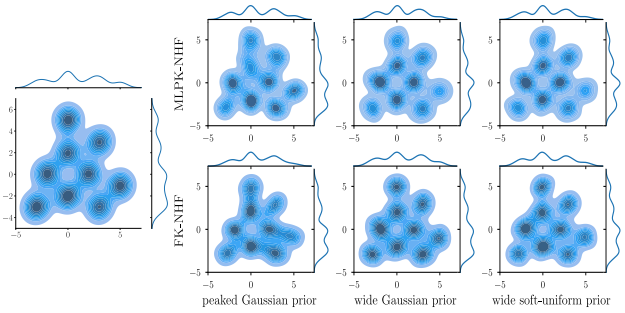

Figure 2: Density estimation with its marginals of the target 2D Gaussian mixture (Left) and of the samples produced by MLPK-NHF (Right, Top) and FK-NHF (Right, Bottom), with, from left to right, peaked Gaussian, wide Gaussian and wide soft-uniform prior.

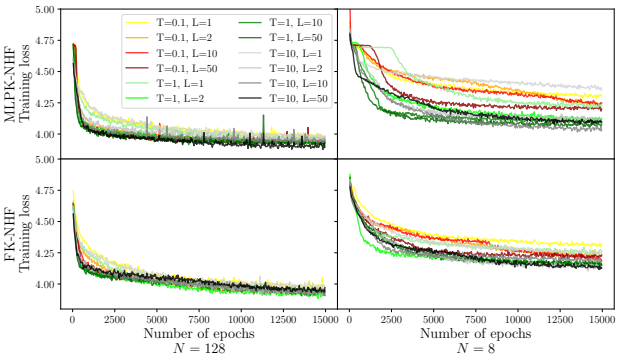

Figure 3: Training loss as a function of epochs for models with different N (number of neurons per hidden layer in each neural network of the model), L (number of Leapfrog steps) and T (integration time).

We tested both FK-NHF and MLPK-NHF with various choices of L, T and N and 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 B.2.

First, FK-NHF is more robust than the MLPK one to the choices of L and T, at fixed N, as discrepancy in the loss decay more clearly appears especially with $N = 8$. Then, regarding the tuning 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 FK-NHF. Increasing the number of leapfrog steps leads to better final performance even if the 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 FK-NHF, showing that the latter efficiently adjusts to this parameter. As

for the MLPK-NHF, the effect of the integration time is clearer, but mostly at $N = 8$, where performance improves for $T = 1, 10$ compared to $T = 0.1$. Overall, as the number of parameters in the model increases, the impact of the integration time becomes limited.

Thus, there are 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: the number of Leapfrog steps, whose tuning is less sensitive when using FK-NHF. Furthermore, compared to diffusion models (Sohl-Dickstein et al., 2015), the required amount of steps is quite low.

5.2 Impact of the prior distribution on the learned dynamics

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 $N = 128$, $T = 1$ and $L = 10$, with a 5,000 points training dataset and with the soft-uniform prior, a peaker Gaussian prior $\mathcal{N}(0, I_2)$ and a wide Gaussian prior $\mathcal{N}(0, 2.5^2 I_2)$.

All considered schemes recover the nine correct modes from the target distribution, as illustrated in Figure 2. We now consider the learned potential V. As the Hamiltonian evolution only involves its derivative, we represented a shifted version in Figure 4. Choosing a relatively flat soft-uniform prior distribution that covers the target region, multimodality transfers to the potential energy for both FK-NHF and MLPK-NHF. The potential exhibits local extrema centered at the modes of the target, which can either be minima or maxima for the MLPK-NHF but are minima for the FK one. Indeed, with a MLPK 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. Similar results were obtained using the wide Gaussian prior with a 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, with a "peaked" prior distribution $\mathcal{N}(0, I_2)$ for the MLPK-NHF, the momenta $\mathbf{p}_T$ generated by the Encoder inherit from the multimodality of the target distribution, with the same number of modes (see Figure 5). The learned energies are then different from the classical Physical ones and differ from one model to another. In the case of FK-NHF, multimodality is transferred to the potential energy, showing the robustness of the model to the choice of prior distribution.

Thus, using FK-NHF allows for a more robust transfer of important properties of the target distribution into the learned potential. More specifically, the model is able to learn an interpretable potential with extrema centered at the modes of the data. 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.

Finally, learning a potential approximating $-\log \pi$ is interesting in terms of interpretability but also renders the model more robust to hyperparameters. Figures 6 and 7 illustrate how learning an interpretable multimodal potential makes the model less sensitive to the choice of Leapfrog steps and robust to some dynamics extrapolation.

As shown in Figure 6 and Appendix B.3, we also investigated the possibility of enforcing the transfer of multimodality to V by removing the Encoder and having $\mathbf{p}_T$ drawn from a $\mathcal{N}(0, s^2 I)$, s being learned during training. While it improves MLPK-NHF for a peaked prior, we find it is less efficient and directly interpretable than fixing K. When fixing K, an Encoder-free model comes with a reduced complexity but we find that leaving as much flexibility as possible in the generation of momenta is the relevant option, especially for challenging problems and a small number of leapfrog steps.

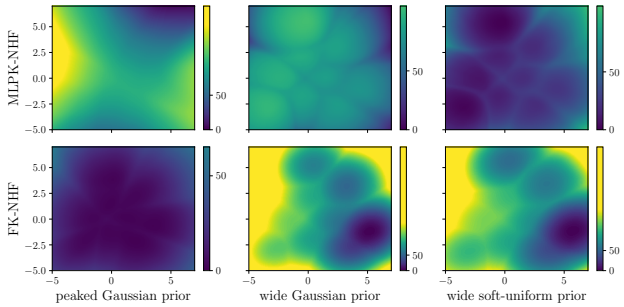

Figure 4: Shifted potential energies learned by the six models previously defined.

6 NHF FOR IMAGE GENERATION

To the best of our knowledge, NHF models have not been tested on high-dimensional image generation problems. We run experiments for sampling the MNIST handwritten digits (Deng, 2012) dataset, as well as additional tests on the Fashion MNIST dataset (Xiao et al., 2017). FK-NHF and MLPK-NHF are compared to a RealNVP (Dinh et al., 2017) with a similar number of learnable parameters. The RealNVP implementeation is based on an open-source GitHub repository [2]. Details

[2] https://github.com/bjlkeng/sandbox/tree/master/realnvp

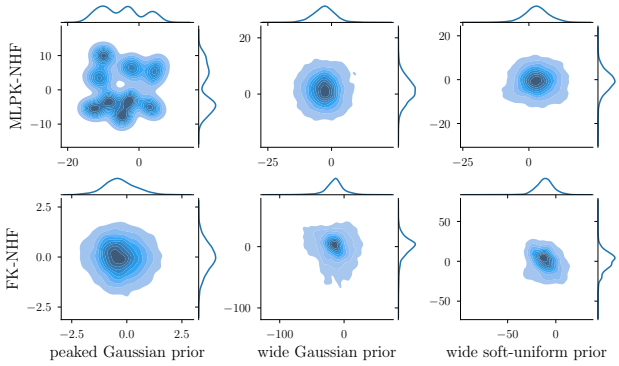

Figure 5: Density estimation of the artificial momenta from Encoder for the six models previously defined.

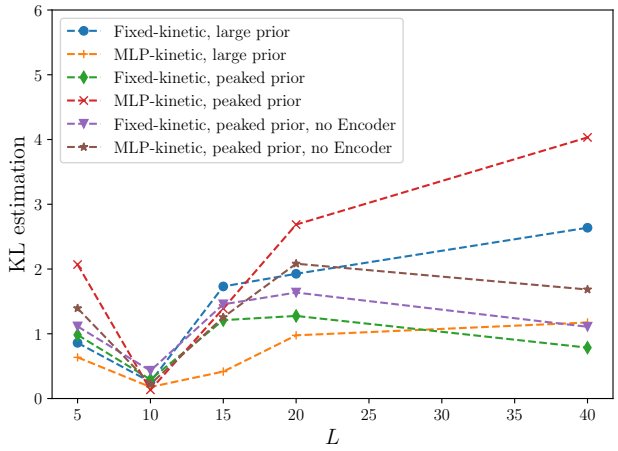

Figure 6: Evolution of KL estimation between the true and the model distribution as L increases, for models trained with $L = 10$. MLPK-NHF models with Encoder have $\sim 70,000$ learnable parameters while all the others have $\sim 50,000$ learnable parameters.

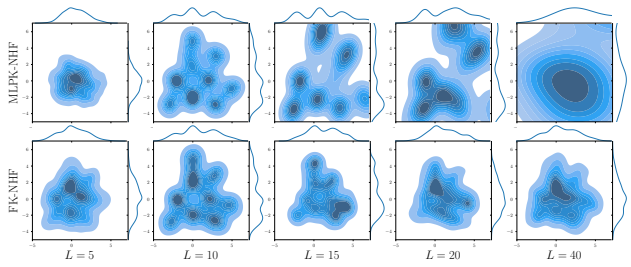

Figure 7: Density estimations as L increases of samples generated by two models trained with $L = 10$ and a peaked Gaussian prior $\mathcal{N}(0, I_2)$. First row: MLPK-NHF which has learned a non-interpretable unimodal potential. Second row: FK-NHF which has learned an interpretable multimodal potential.

on the experiments can be found in the Appendix C.1. We use a pre-processing step prescribed in Dinh et al.

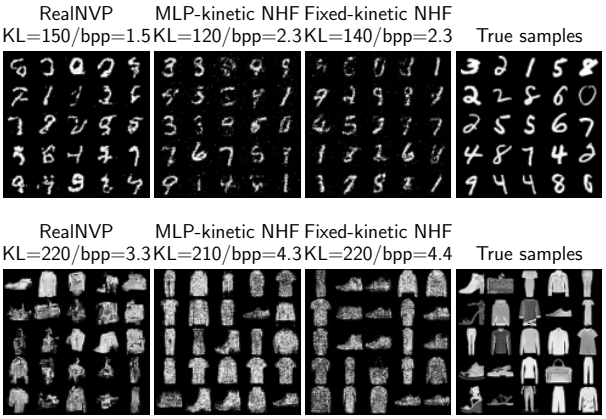

Figure 8: Samples produced after training on the MNIST and Fashion-MNIST datasets along with an estimation of the KL divergence between the true and model distributions and the number of bits per pixel.

(2017) consisting of learning the dequantized target distribution in logit space. The choice of mass matrix for FK-NHF is important for expressivity. This behaviour should be familiar to the HMC community (Duane et al., 1987) for which an optimal mass matrix is important for efficient exploration (Neal, 2012).

Quality of sampling is quantitatively assessed by the KL divergence, estimated following Perez-Cruz (2008), and the number of bits per pixel (Papamakarios et al., 2017). Our experiments (Figure 8 and Appendix C.2) show that RealNVP and NHF slightly outperform each other depending on the metric. The performance of FK-NHF is achieved with a simpler architecture though. Furthermore, the learned potentials have extrema located at the modes of the target distribution, see Appendix C.3. This capacity of the models to learn the modes of the data and to store the information in the potential energy underlines interpretability.

To compare ODE-driven flow-based models, such as NHF, with SDE-driven diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020), we also tested an implementation of Ho et al. (2020) based on an open-source GitHub repository [3]. We adapted it so that it has 2.7 million parameters (same order as the other algorithms) and uses the same pre-processing step. The experiments ran on a HPC cluster with the same number of epochs and batch size than for the previous experiments with normalizing flows. We tested the model with $L = 10$ and $L = 100$ denoising steps. Results are visible on Figure 9 and it appears that around 100 denoising steps are required for producing good samples. This should be directly compared with the 10

[3] `https://github.com/lucidrains/denoising-diffusion-pytorch/`

Leapfrog steps used within NHF models. This comes in addition to the extra cost of sampling with a trained diffusion model since it requires approximating the reverse process from noise to images with a Markov chain.

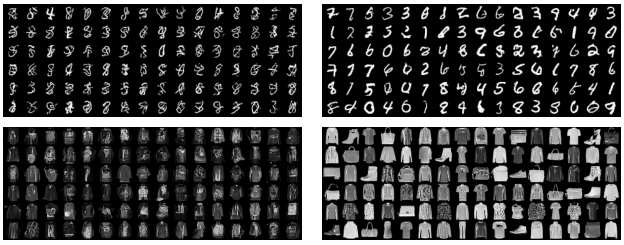

Figure 9: Samples produced by the diffusion model after training on the MNIST and Fashion-MNIST datasets. Left: $L = 10$ denoising steps (KL divergence between the true and model distributions: KL= 250 (MNIST), KL=340 (Fashion-MNIST)). Right: $L = 100$ denoising step (KL= 70 (MNIST), KL= 70 (Fashion-MNIST)).

7 ADAPTING NHF FOR BAYESIAN INFERENCE

7.1 Methodology, derivation of the new loss function

NHF can be used to perform Bayesian inference, by 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 distribution ℓ (see Figure 10). The main difference with the above-described NHF lies in the loss inspired by the KL phase in Boltzmann Generators (Noé et al., 2019), as well as in the learning procedure. The NHF becomes a generator of a family of functions for variational inference. During training, this NHF takes batches of $\mathbf{q}_0$ from the prior distribution as inputs. For each $\mathbf{q}_0$, one $\mathbf{p}_0$ is drawn from a Gaussian distribution whose mean and deviation depend on the $\mathbf{q}_0$. The resulting point in phase-space evolves through L Leapfrog steps with integration time δt. The outputs consist in the final positions $\mathbf{q}_T$ and momenta $\mathbf{p}_T$, as well as the initial mean $\mu(\mathbf{q}_0)$, 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 can 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.

Computing the loss requires access to the likelihood distribution ℓ of the model, which encapsulates the

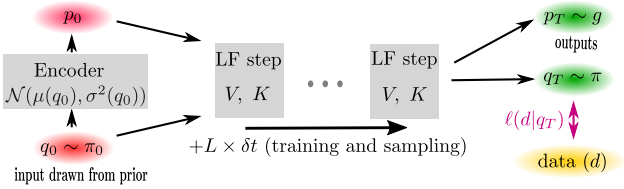

Figure 10: Schematic representation of NHF for Bayesian inference.

covariance matrix of the data as well as the underlying physical mapping between vectors of parameters and the corresponding data. In the framework of Hamiltonian dynamics, the full system is made of both positions (the parameters of interest) and artificial momenta. We call $\mathbf{q}_0, \mathbf{p}_0$ the initial position and momentum, respectively, and $\mathbf{q}_T, \mathbf{p}_T$ the corresponding final position and momentum, respectively, obtained after L Leapfrog transformations $\mathcal{T}_1^{\delta t}, ..., \mathcal{T}_L^{\delta t}$ with timestep δt, i.e: $(\mathbf{q}_T, \mathbf{p}_T) = \mathcal{T}_L^{\delta t} \circ ... \circ \mathcal{T}_1^{\delta t}(\mathbf{q}_0, \mathbf{p}_0) := \mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)$. 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 changing the variables, the model joint distribution M may be written as:

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

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 then 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 one depending on $\mathbf{p}$. Using Bayes' theorem, we have (see Appendix D):

$$\begin{aligned} D_{KL}(M(\mathbf{q}_T, \mathbf{p}_T) \,||\, \pi(\mathbf{q}_T|\boldsymbol{d})g(\mathbf{p}_T)) = \int \pi_0(\mathbf{q}_0, \mathbf{p}_0) \\ \times \big[\log \pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0) - \log \pi_0(\mathcal{T}_q(\mathbf{q}_0, \mathbf{p}_0)) \\ - \log \ell(\boldsymbol{d}|\mathcal{T}_q(\mathbf{q}_0, \mathbf{p}_0)) - \log g(\mathcal{T}_p(\mathbf{q}_0, \mathbf{p}_0))\big] d\mathbf{q}_0 d\mathbf{p}_0 + \text{cst}. \end{aligned} \tag{4}$$

7.2 Application to cosmology

We apply the above architecture to cosmological analysis: the determination of the cosmic expansion, and more generally of the cosmological parameters, from the observation of brightness and recession velocity of Type Ia supernovæ (e.g. Riess et al., 1998; Betoule et al., 2014). While this model used so far has been simple, it may be expanded in very complicated directions for which sampling from the probability distribution

becomes complex. New observatories are presently being built and expected to deliver tens of thousands of new supernovæ Ia over the next decade (Abell et al., 2009).

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, a database of type Ia supernovae reports 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^* is a function for which one can get a closed-form approximation in a flat Universe (Pen, 1999), see Appendix E for details.

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 $\text{data} = \{z_i, \mu_i\}_{1 \leq i \leq D}$ of type Ia supernovae, and the covariance matrix C of the observed distance moduli. We assume for simplicity that the likelihood of the problem is Gaussian, i.e. the observed data and the simulated output from parameters differ up to Gaussian noise. We note that the exact simulation for cosmological analysis of the supernova brightness is a complicated and expensive procedure, involving many nuisance parameters which participate in the final noise. It also acts as an example for more complex inference procedures, such as one relying on galaxy clustering, or weak lensing. It is thus crucial to use the least amount of parameters, and the least simulations possible to run the inference, which is the aim of this section. The final momenta distribution g is set to a Normal distribution. The trace plots in figure 11 compare the performance of an FK and MLPK-NHF with an HMC. They represent the cumulative means and standard deviations of the set of samples. These experiments first show that NHF, using learnt Hamiltonian flows, are competitive with HMC, using exact Hamiltonian flows. Indeed, NHF samples only show around 1% bias error but they are completely uncorrelated from each other. After training, for a fixed-computational budget, the convergence of the empirical average can then be faster than with HMC (clearly appearing for instance on the variance of h in figure 11). Furthermore, the experiments show that the KL-minimization approach is performing better. We leave for future work a possible correction using importance sampling methods at the end of training. More generally, a careful analysis of the trade-off between sampling quality and computational cost, typically on a complex and multimodal target, will be of interest.

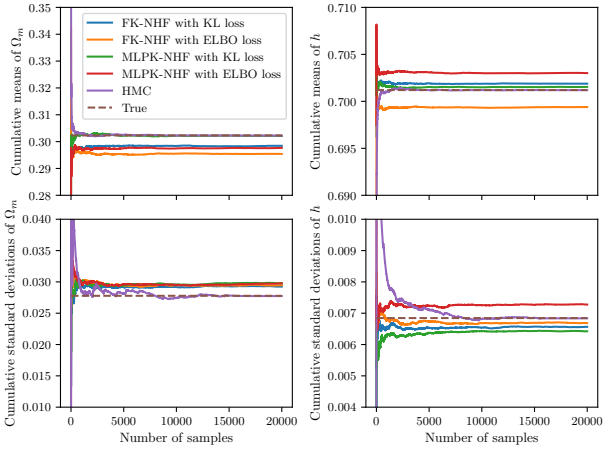

Figure 11: Trace plots of means and standard deviations of Ω_m and h produced by trained NHF models and for an HMC on a 20,000-sample dataset, compared to the ground truth. Soft-uniform prior, 30,000 training epochs, $g \sim \mathcal{N}(0, I_2)$.

8 CONCLUSION

In this work, we analyzed and improved Normalizing Hamiltonian Flows algorithms for 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, as reversibility is ensured by the symplectic integrator, they allow for flexibility in the neural network architecture. This flexibility allowed us to propose a NHF variant based on classical kinetic energy. By exploring a 2D mixture problem, we illustrated how the explicit classical design of the kinetic energy is a way to increase robustness and facilitate interpretability while reducing the computational cost. While testing NHF models for image generation, both show similar generative performance and are able to preserve their interpretability properties. It is noteworthy that, compared to diffusion models, they only require a short dynamics integration. Finally, we explained how to adapt NHF to the context of Bayesian inference to obtain a sampler of the posterior distribution. Further work will address methodological issues as to how the bias generated by a trained model could be corrected by importance sampling techniques, typically on high dimensional cosmological models but also more fundamental questions regarding a more precise comparison of NHF with diffusion models.

Acknowledgments

All the authors are grateful for 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 performed using HPC resources from GENCI–IDRIS (Grant 2023-AD011014013). It was supported by the French ANR under the grant ANR-20-CE46-0007 (*SuSa* project) and by the Simons Collaboration on "Learning the Universe". AG acknowledges the support of the Institut Universitaire de France. JJ acknowledges support from the Swedish Research Council (VR) under the project 2020-05143 – "Deciphering the Dynamics of Cosmic Structure". This work was carried out within the Aquila consortium (`https://www.aquila-consortium.org`).

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. (2009). LSST Science Book, Version 2.0. *arXiv e-prints*, page arXiv:0912.0201.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. (2022). *Gradient-Based Attribution Methods*, page 169–191. Springer-Verlag, Berlin, Heidelberg.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*.

Berard, H., Gidel, G., Almahairi, A., Vincent, P., and Lacoste-Julien, S. (2020). A closer look at the optimization landscapes of generative adversarial networks. In *International Conference on Learning Representations*.

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. (2014). Improved cosmological constraints from a joint analysis of the SDSS-II and SNLS supernova samples. *Astronomy & Astrophysics*, 568:A22.

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

Celledoni, E., Leone, A., Murari, D., and Owren, B. (2023). Learning hamiltonians of constrained mechanical systems. *Journal of Computational and Applied Mathematics*, 417:114608.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142.

Dhulipala, S. L. N., Che, Y., and Shields, M. D. (2022). Bayesian inference with latent hamiltonian neural networks.

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

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2017). Density estimation using real NVP. In *International Conference on Learning Representations*.

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

Eghbal-zadeh, H., Zellinger, W., and Widmer, G. (2019). 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*.

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

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

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

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

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

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

Jimenez Rezende, D., Racanière, S., Higgins, I., and Toth, P. (2019). Equivariant Hamiltonian Flows. *arXiv e-prints*, page arXiv:1909.13739.

Kingma, D. P. and Ba, J. (2015). 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*.

Landau, L. and Lifshitz, E. (1982). *Mechanics: Volume 1*. Number vol. 1. Elsevier Science.

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

Lin, Z., Khetan, A., Fanti, G., and Oh, S. (2018). 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., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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

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

Neal, R. (2012). Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*.

Noé, F., Olsson, S., Köhler, J., and Wu, H. (2019). Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147.

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

Papamakarios, G., Pavlakou, T., and Murray, I. (2017). Masked autoregressive flow for density estimation. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Pen, U.-L. (1999). Analytical fit to the luminosity distance for flat cosmologies with a cosmological constant. *The Astrophysical Journal Supplement Series*, 120(1):49–50.

Perez-Cruz, F. (2008). Kullback-leibler divergence estimation of continuous distributions. In *2008 IEEE International Symposium on Information Theory*, pages 1666–1670.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019). 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.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France. PMLR.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "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, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

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. (1998). Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. *The Astronomical Journal*, 116(3):1009–1038.

Robert, C. P. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer, 2nd edition.

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

Rossky, P. J., Doll, J. D., and Friedman, H. L. (1978). Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633.

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

Shen, J., Chowdhury, J., Banerjee, S., and Terejanu, G. (2023). Machine fault classification using hamiltonian neural networks.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France. PMLR.

Tabak, E. and Vanden-Eijnden, E. (2010). Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233.

Toth, P., Rezende, D. J., Jaegle, A., Racanière, S., Botev, A., and Higgins, I. (2020). Hamiltonian generative networks. In *International Conference on Learning Representations*.

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

Winkler, C., Worrall, D., Hoogeboom, E., and Welling, M. (2019). Learning likelihoods with conditional normalizing flows.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [**Yes**/No/Not Applicable]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [**Yes**/No/Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [**Yes**/No/Not Applicable] **We attached a ready-to-run Python script for testing a NHF during submission. Also, a public version of the code [4] has been released.**

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [**Yes**/No/Not Applicable]
   (b) Complete proofs of all theoretical results. [**Yes**/No/Not Applicable] **Mathematical derivations can be found in the Appendices.**
   (c) Clear explanations of any assumptions. [**Yes**/No/Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [**Yes**/No/Not Applicable] **See Appendices.**
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [**Yes**/No/Not Applicable] **See Appendices.**
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [**Yes**/No/Not Applicable] **See Appendices.**
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [**Yes**/No/Not Applicable] **See Appendices.**

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [**Yes**/No/Not Applicable]
   (b) The license information of the assets, if applicable. [**Yes**/No/Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [**Yes**/No/Not Applicable]
   (d) Information about consent from data providers/curators. [**Yes**/No/Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes/No/**Not Applicable**]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Yes/No/**Not Applicable**]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/**Not Applicable**]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/**Not Applicable**]

[4] `https://plmlab.math.cnrs.fr/stoch-algo-phys/generative-models/fixed-kinetic-NHF/`

Supplementary Materials

A METRICS FOR EVALUATING THE QUALITY OF SAMPLING

In a D-dimensional space, we consider a set of S samples $\{X_s\}_{s=1}^S := \{(X_{s,1}, ..., X_{s,D})\}_{s=1}^S$ generated by a model and S samples $\{X_s^0\}_{s=1}^S := \{(X_{s,1}^0, ..., X_{s,D}^0)\}_{s=1}^S$ drawn from the true (test) dataset.

A.1 Kullback-Leibler divergence

A natural (pseudo-)distance is the KL-divergence $D_{KL}(\pi||m)$ between the true target distribution with density π and the model distribution with density m. This pseudo-distance quantifies the loss of information when using the model distribution instead of the true target for describing the data - so the lower the better. The KL-divergence is defined as

$$D_{KL}(\pi||m) = \int \pi(x) \ln \frac{\pi(x)}{m(x)} dx \geq 0.$$

When one of these probability distributions is intractable, as it is the case here, we estimate D_{KL} by comparing samples from the true target distribution with samples from the model distribution. The procedure described in Perez-Cruz (2008) proceeds according to a k-neighborhood density estimate of the two distributions:

$$D_{KL}(\pi||m) \approx -\frac{D}{S} \sum_{s=1}^{S} \ln \frac{r_k(X_s)}{s_k(X_s)} + \ln \frac{S}{S-1}$$

where $r_k(X_s)$ is the k-th closest neighbor of X_s in $\{X_s\}_{s=1}^S \backslash \{X_s\}$ and $s_k(X_s)$ is the k-th closest neighbor of X_s in $\{X_s^0\}_{s=1}^S$. Parameters chosen in our estimation are $S = 1024$ and $k = 1$.

A.2 Bits per pixel

For evaluating the quality of images generated by the considered models, we compute the number of bits per pixel, decaying as the quality of the image increases. We start with a pre-processing step. First, pixels of training images are dequantized by adding uniform noise $\varepsilon \sim \mathcal{U}]0, 1[$ and ranging them back to interval $[0, 1]$ as $x \leftarrow (255x + \varepsilon)/256$. Then, the models are trained on the target distribution in logit space by transforming the resulting noisy pixels as $x \leftarrow \text{logit}((1 - 2\lambda)x + \lambda)$ with $\lambda = 10^{-6}$.

Once trained, we evaluate the number of bits per pixel of a pre-processed image $\tilde{X}_s^0 := (\tilde{X}_{s,1}^0, ..., \tilde{X}_{s,D}^0)$ in logit space from the test dataset following Papamakarios et al. (2017):

$$b(\tilde{X}_s^0) = -\frac{\ln m(\tilde{X}_s^0)}{D \ln 2} - \log_2(1 - 2\lambda) + \frac{1}{D} \sum_{d=1}^{D} \left[\log_2(\text{logit}(\tilde{X}_{s,d}^0)) + \log_2(1 - \text{logit}(\tilde{X}_{s,d}^0)) \right].$$

In the above equation, we evaluate the probability distribution in the logit space of the model, namely $m(\tilde{X}_s^0)$. For a classical flow-based model, it is straightforward since $m(\tilde{X}_s^0) = \pi_0(T^{-1}(\tilde{X}_s^0)) \times \left| \text{Jac}_{T^{-1}}(\tilde{X}_s^0) \right|$ where T^{-1} is the transformation from logit space to latent space learned by the model.

However, for NHF, the change of variable formula is only valid in phase-space for the model joint distribution of the positions and momenta: $M(\tilde{X}_s^0, V_s) = \Pi_0(T^{-1}(\tilde{X}_s^0, V_s)) \times 1$. We use a Monte Carlo approximation of $m(\tilde{X}_s^0)$ by drawing N momenta $V_{s,1}, ..., V_{s,N}$ from the Gaussian distribution $f(.|\tilde{X}_s^0)$ parameterized by the Encoder as:

$$m(\tilde{X}_s^0) = \int M(\tilde{X}_s^0, V_s) dV_s \approx \frac{1}{N} \sum_{i=1}^{N} \frac{M(\tilde{X}_s^0, V_{s,i})}{f(V_{s,i}|\tilde{X}_s^0)}.$$

For evaluating the number of bits per pixel of a model on the two MNIST datasets, we average the bits per pixel values obtained with 1024 images from the test dataset, using $N = 10$ for NHF. The histograms of bits per pixel are exhibited in Appendix C.2.

B NUMERICAL EXPERIMENTS ON THE 2D PROBLEM

B.1 Experimental details

For the 2D Gaussian mixture problem, we tested four different NHF models:

- MLP-kinetic NHF with Encoder: μ and σ are MLPs with size $(2, N, N, 2)$, V and K are MLPs with size $(2, N, N, 1)$. According to the experiments, we used $N = 8,\ 32,\ 128$.
- Fixed-kinetic NHF with Encoder: μ and σ are MLPs with size $(2, N, N, 2)$, V is a MLP with size $(2, N, N, 1)$. Kinetic energy K is a positive quadratic form whose mass matrix is learned during training. According to the experiments, we used $N = 8,\ 32,\ 128$.
- MLP-kinetic NHF without Encoder: we use this model for experiments in Section 5. Artificial momenta $\mathbf{p}_T$ are drawn from a $\mathcal{N}(0, C)$ where $C = \mathrm{Diag}(s_1^2, s_2^2)$, s_1, s_2 being learned during training. V and K are MLPs with size $(2, 156, 156, 1)$. The number of neurons per hidden layer was chosen so that the resulting model has about the same number of parameters as an Encoder-based Fixed-kinetic NHF with $N = 128$.
- Fixed-kinetic NHF without Encoder: this model is considered in Section 5. Artificial momenta $\mathbf{p}_T$ are drawn from a $\mathcal{N}(0, C)$ where $C = \mathrm{Diag}(s_1^2, s_2^2)$, s_1, s_2 being learned during training. V is a MLP with size $(2, 220, 220, 1)$. Kinetic energy K is a positive quadratic form whose mass matrix is learned during training. The number of neurons per hidden layer was chosen so that the resulting model has about the same number of parameters as an Encoder-based Fixed-kinetic NHF with $N = 128$.

We used Softplus activation functions in between hidden layers. All models were trained on a 5,000 points dataset with minibatches of size 512. Weights and biases were optimized with the Adam algorithm Kingma and Ba (2015), setting the learning rate to 5×10^{-4}. The experiments were run on a HPC cluster, each of them using one GPU.

B.2 Additional experiments for showing robustness of the models

We also present the results of additional experiments with $N = 32$ in Figure 12. Removing more than 90% of parameters (passage from $N = 128$ to $N = 32$), 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 13. The latter clearly illustrates the robustness of the fixed-kinetic model, see Figure 12. It also shows that models with $L = 1$ perform poorer than those with $L = 2,\ 10,\ 50$.

B.3 Additional results on Encoder-free NHF models

We tested different Encoder-free NHF models on the 2D Gaussian mixture target. As can be seen in Figure 14, all models have learned an interpretable multimodal potential. The corresponding samples are presented in Figure 15. These experiments clearly illustrate that the transfer of multimodality and thus interpretability of the model can be achieved more easily by fixing the distribution of the artificial momenta $\mathbf{p}_T$ to a unimodal Gaussian. However, we chose to push the investigation further only with Encoder-based models as leaving as much flexibility as possible in the generation of momenta may be useful in sampling more challenging target distributions.

C NUMERICAL EXPERIMENTS ON THE MNIST DATASETS

C.1 Experimental details

The considered models were tested on the MNIST handwritten digits and Fashion MNIST datasets, which contain 60,000 images of size 28×28. The energy functions within the MLP-kinetic and Fixed-kinetic NHF are parameterized by 3-hidden layer MLPs with size $(784, 512, 256, 128, 1)$. As for μ and σ, they are 3-hidden layer MLPs with size $(784, 256, 256, 256, 784)$. We use LeakyReLU activation functions in between hidden layers with slope 0.1 for μ and σ and Softplus activation functions in between hidden layers for the energies. For the Fixed-Kinetic model, the mass matrix is optimized on the fly during training by learning its Cholesky decomposition, which has $D(D + 1)/2$ parameters, D being the dimension of data, as was done in Celledoni et al. (2023). This represents a grand total of 1.94 million learnable parameters for Fixed-kinetic NHF and

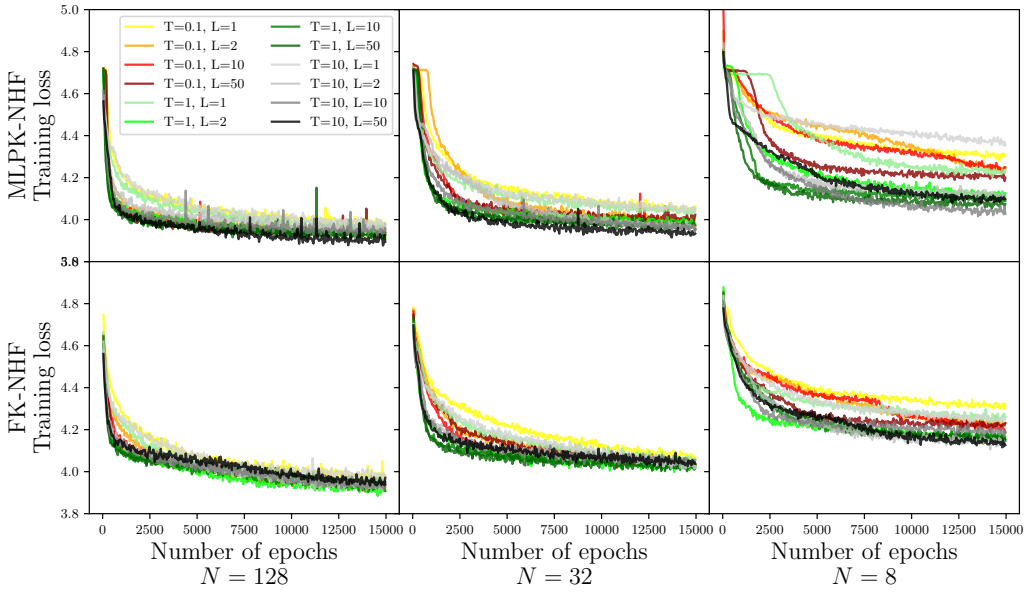

Figure 12: Training loss as a function of epochs for models with different N (number of neurons per hidden layer in each neural network of the model), L (number of Leapfrog steps) and T (integration time).

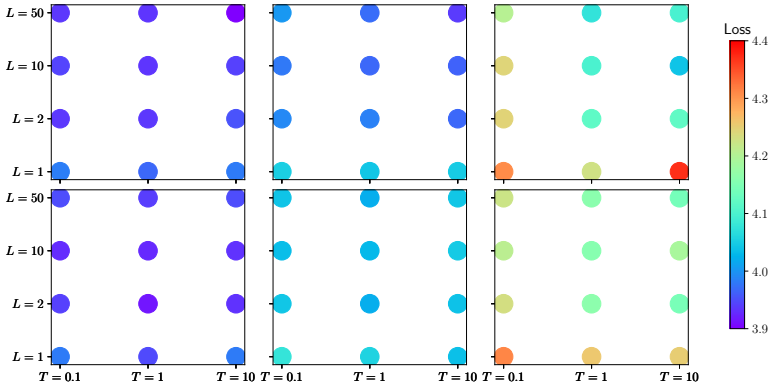

Figure 13: Scatter plots L vs. T of final values of the loss averaged on last 500 epochs. First row: MLP-kinetic NHF; second row: fixed-kinetic model. From left to right: $N = 128,\ 32,\ 8$. Models were trained for $15,000$ epochs on a 5,000-point dataset with minibatches of size 512.

2.20 million for MLP-Kinetic NHF, that both use 10 Leapfrog steps with integration timestep $dt = 0.1$. As for the Real NVP to which they are compared, it uses 6 coupling layers and 32 planes for a total of 2.27 million learnable parameters. We adapted an architecture from a public open source GitHub project available at `https://github.com/bjlkeng/sandbox/tree/master/realnvp` under MIT Licence. All models were trained using a $\mathcal{N}(0, I_{784})$ base distribution, except for the NHF models on the MNIST handwritten digits which use a $\mathcal{N}(0, 2^2 I_{784})$. All models were trained for 50 epochs on minibatches with size 32 and optimization was performed using the Adam algorithm Kingma and Ba (2015). The experiments were run on a HPC cluster, each of them using one GPU.

C.2 Additional quantitative results

It should be noted that the reported values of bits per pixel for MNIST handwritten digits and Fashion MNIST correspond to an average over some samples drawn from the true test dataset. In Figure 16 and Figure 17 are plotted the histograms of the bits per pixel values for 1024 samples drawn from the true test datasets, for each model. Figure 16 on the MNIST dataset exhibits comparable results for the three tested architectures whereas

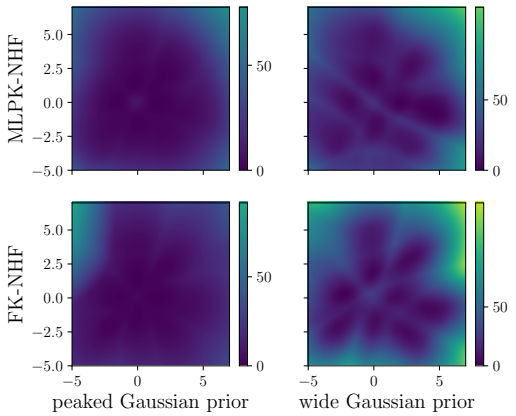

Figure 14: Shifted potential energy learned by 4 different Encoder-free models. All of them have recovered the 9 correct modes of the data.

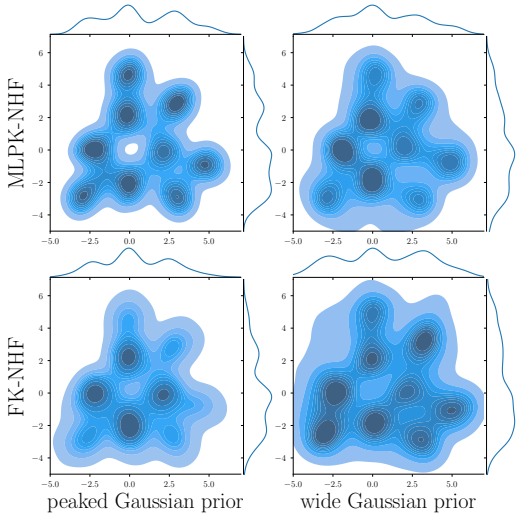

Figure 15: Density estimation of samples generated by 4 different Encoder-free models.

the Fashion MNIST experiment shows slightly better bits per pixel values for RealNVP.

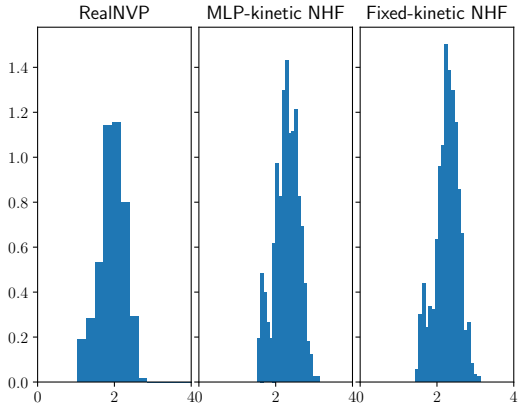

Figure 16: Histogram of bits per pixel values from the MNIST handwritten digits dataset. The standard deviation is approximately equal to 0.3 for each model.

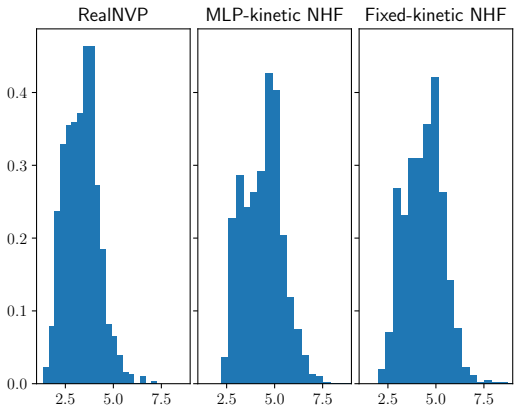

Figure 17: Histogram of bits per pixel values from the Fashion MNIST dataset. The standard deviation is approximately equal to 1 for each model.

C.3 Interpretability of NHF models in higher dimension

Both NHF models preserve their interpretability properties even in high dimension image generation problems. They have learned extrema at the modes of data. As can be seen in Figure 18, the fixed-kinetic model has learned local minima at the modes of data while the usual MLP-kinetic model has learned local maxima. Imposing a positive quadratic term for the kinetic energy ensures that it will always be minima.

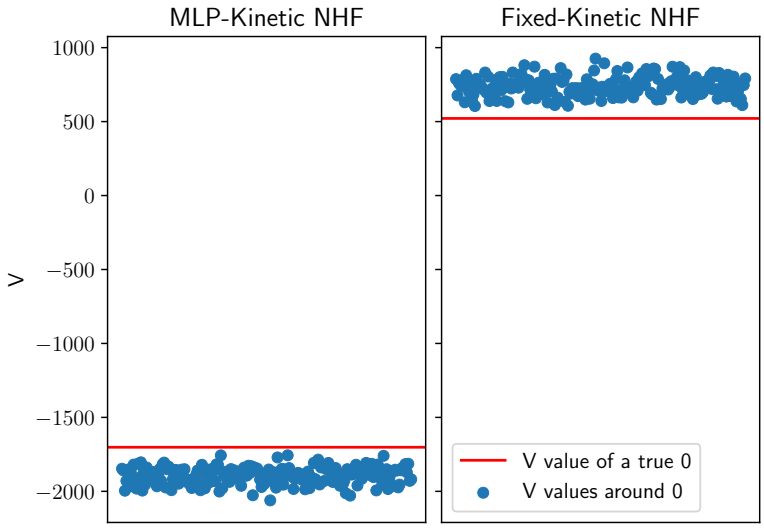

Figure 18: For two NHF models trained on the MNIST handwritten digits, we plot the value of the potential for a true '0' corresponding to a point x_0 in a 784-dimensional logit space, as well as the values of the potential for 200 perturbations of the form $x = x_0 + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, 2^2 I_{784})$.

Again, by learning an interpretable potential we observe that the model is less sensitive to the number of Leapfrog steps. This is particularly visible for MLP-kinetic NHF on both MNIST handwritten digits and Fashion MNIST datasets, as well as for FK-NHF especially on Fashion MNIST, see Figure 19 and Figure 20.

D DERIVATION OF THE KL-DIVERGENCE AND ELBO FOR THE INFERENCE PROBLEM

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

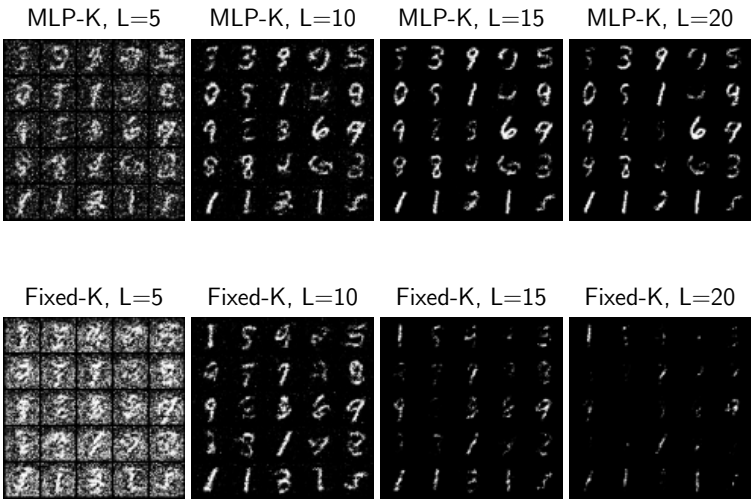

Figure 19: Stability of both NHF models trained with $L = 10$ to the number of Leapfrog steps, for the MNIST handwritten digits dataset.

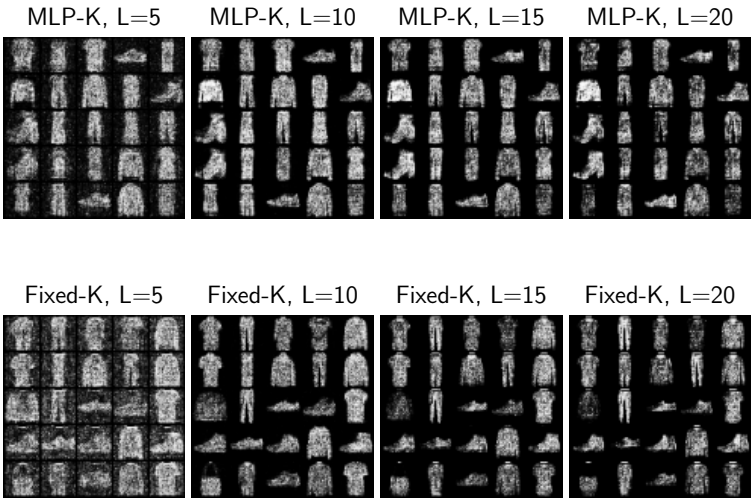

Figure 20: Stability of both NHF models trained with $L = 10$ to the number of Leapfrog steps, for the Fashion MNIST dataset.

$$
\begin{aligned}
&D_{KL}(M(\mathbf{q}_T,\mathbf{p}_T)\ ||\ \pi(\mathbf{q}_T|\boldsymbol{d})g(\mathbf{p}_T)) = \int M(\mathbf{q}_T,\mathbf{p}_T)\log M(\mathbf{q}_T,\mathbf{p}_T)d\mathbf{q}_T d\mathbf{p}_T - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi(\mathbf{q}_T|\boldsymbol{d}) + \log g(\mathbf{p}_T)\right] d\mathbf{q}_T d\mathbf{p}_T \\
&= \int \Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T,\mathbf{p}_T))\log\Pi_0(\mathcal{T}^{-1}(\mathbf{q}_T,\mathbf{p}_T))d\mathbf{q}_T d\mathbf{p}_T - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi_0(\mathbf{q}_T) + \log\ell(\boldsymbol{d}|\mathbf{q}_T) - \log p(\boldsymbol{d}) + \log g(\mathbf{p}_T)\right] d\mathbf{q}_T d\mathbf{p}_T \\
&= \int \Pi_0(\mathbf{q}_0,\mathbf{p}_0)\left[\log\pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0)\right] d\mathbf{q}_0 d\mathbf{p}_0 - \int M(\mathbf{q}_T,\mathbf{p}_T)\left[\log\pi_0(\mathbf{q}_T) + \log\ell(\boldsymbol{d}|\mathbf{q}_T) + \log g(\mathbf{p})\right] d\mathbf{q}_T d\mathbf{p}_T + \text{cst} \\
&= \int \Pi_0(\mathbf{q}_0,\mathbf{p}_0)\left[\log\pi_0(\mathbf{q}_0) + \log f(\mathbf{p}_0|\mathbf{q}_0) - \log\pi_0(\mathcal{T}_q(\mathbf{q}_0,\mathbf{p}_0)) - \log\ell(\boldsymbol{d}|\mathcal{T}_q(\mathbf{q}_0,\mathbf{p}_0)) - \log g(\mathcal{T}_p(\mathbf{q}_0,\mathbf{p}_0)\right] d\mathbf{q}_0 d\mathbf{p}_0 + \text{cst}.
\end{aligned}
\tag{5}
$$

We can also adapt the ELBO to the 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 $M(q, p) = \pi_0(q)\ell(d|q)g(p)$:

$$\text{ELBO}(\mathbf{q}_0) = E_f \left[\ln \left[\pi_0(T_q(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)))\ell(d|T_q(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0)))g(T_p(\mathcal{T}(\mathbf{q}_0, \mathbf{p}_0))) \right] - \ln f(\mathbf{p}_0|\mathbf{q}_0) \right]. \tag{6}$$

E SOME TECHNICAL DETAILS ABOUT THE COSMOLOGICAL PROBLEM

In Section 7, we defined a relationship between the distance modulus μ and the redshift z which also depends on two cosmological parameters Ω_m and h. To be more specific:

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

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 from 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

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

Note that 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 $\mathbf{q}_T$.