Rapid Energy Sampling Hamiltonian Dynamics Without MCMC

Anonymous Author(s)
Affiliation
Address
email

Abstract

Sampling from an unnormalized probability distribution is a fundamental problem in machine learning with applications in Bayesian inference and energy-based model training. After decades of research, variations of MCMC remain the default approach to sampling despite slow convergence. Auxiliary neural models can learn to speed up sampling, but the overhead for training the extra model can be prohibitive. We propose a fundamentally different approach to this problem via a new Hamiltonian dynamics based on a non-Newtonian momentum. The proposed dynamics define a parameter-free continuous normalizing flow in an extended state space but exactly sample the unnormalized target distribution in the original coordinates under an assumption of ergodicity. In contrast to MCMC approaches like Hamiltonian MCMC, there is no stochastic step and no Metropolis-Hastings rejection step. Energy Sampling Hamiltonian (ESH) dynamics have a simple form that can be solved with existing ODE solvers, but we derive a specialized solver that exhibits much better performance. ESH dynamics converge faster than their MCMC competitors and finally enable fast, stable training of neural network energy models without auxiliary sampling models.

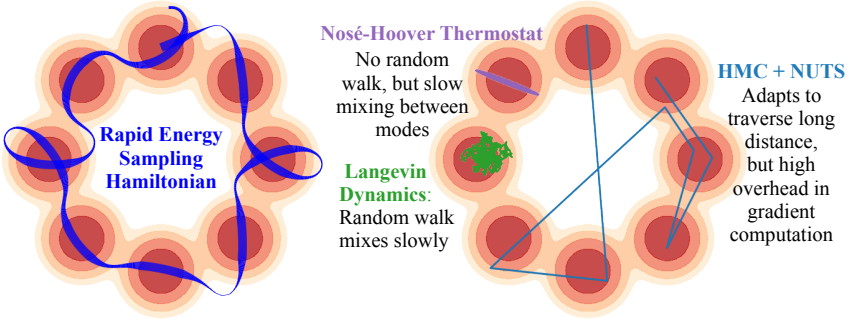

Figure 1: Sampling with *500 gradient evaluations per method*. Energy-Sampling Hamiltonian (ESH) dynamics solve an ODE and line thickness is proportional to dynamical time spent at each point.

1 Introduction

While probabilistic reasoning is a mathematical cornerstone of science and cognition [1], probability distributions that can be directly sampled are rare. Without sampling it is difficult to measure likely outcomes, especially for high-dimensional probability distributions. For elegantly providing a general purpose solution to sampling any target distribution, the Metropolis algorithm for Markov Chain Monte Carlo (MCMC) is considered one of the top algorithms of the 20th century [2]. The "Monte

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Carlo" in MCMC refers to the essential role of stochasticity, or chance, in the approach: start with any state, propose an update, and accept it according to a weighted coin flip.

The most widely used methods for sampling mainly differ in their choice of proposals for the next state of the Markov chain. The most prominent example, Hamiltonian Monte Carlo (HMC), uses Hamiltonian dynamics to suggest proposals that quickly move over large distances in the target space [3]. HMC using the No U-Turn Sampling (NUTS) heuristic for choosing hyper-parameters [4] forms the backbone of modern probabilistic programming languages like Stan [5]. In the limit of one discrete HMC step per proposal, we recover the widely used discrete Langevin dynamics [3, 6]. Many approaches use neural networks to improve sampling either through better proposal distributions or flexible variational distributions [7, 8, 9, 10, 11, 12, 13, 14], but these can be expensive if, for instance, the target distribution to be sampled is itself being optimized.

We propose a new Hamiltonian dynamics, based on the introduction of a non-Newtonian momentum, which leads to a deterministic dynamics that directly samples from a target energy-based model. The idea that deterministic dynamics in an extended space can directly sample from a Gibbs-Boltzmann distribution goes back to Nosé [15]. This principle is widely used in molecular dynamic (MD) simulations [16] but has had limited impact in machine learning. One reason is that direct application of techniques developed for MD often exhibit slow mixing in machine learning applications.

Our contribution stems from the recognition that many of the design considerations for molecular dynamics samplers are to enforce physical constraints that are irrelevant in the context of machine learning problems. By starting from the first principles and discarding those constraints, we are able to discover a Hamiltonian whose dynamics *rapidly* sample from a target energy function by using a non-physical form for the momentum. This is in contrast to a more physical (yet impractically slow) alternative of introducing auxiliary "thermostat" variables that extract energy from the system [17, 18, 19, 20]. The result is a fast and deterministic drop-in replacement for the MCMC sampling methods used in a vast range of machine learning applications. Alternately, our method can be viewed as a normalizing flow in an extended state space which can be used to directly sample from an unnormalized target density without additional training.

The most significant impact for our approach is for applications where minimizing memory and computation is the priority. For training energy-based models, for example, sampling appears in the inner training loop. Re-training a neural sampler after each model update is costly and entails complex design and hyper-parameter choices. While MCMC sidesteps the training of an extra model, MCMC converges slowly and has led to widespread use of a number of dubious heuristics to reduce computation [21, 22]. Because our energy sampling dynamics does not have the stochastic, random walk component of MCMC, it converges towards low energy states much faster, especially in the transient regime, and explores modes faster as shown in Fig. 1 and demonstrated in experiments.

2 Energy Sampling Hamiltonian (ESH) Dynamics

We define a separable Hamiltonian, H, over position $\mathbf{x} \in \mathbb{R}^d$ and velocity $\mathbf{v} \in \mathbb{R}^d$ (we use momentum and velocity interchangeably since we set mass to 1).

$$H(\mathbf{x}, \mathbf{v}) = E(\mathbf{x}) + K(\mathbf{v})$$

The potential energy function, $E(\mathbf{x})$, is the energy for the target distribution with unknown normalization, Z, that we would like to sample, $p(\mathbf{x}) = e^{-E(\mathbf{x})}/Z$ and is defined by our problem. We consider

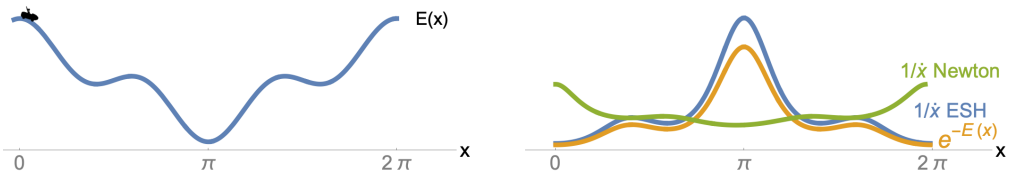

Figure 2: For Newtonian dynamics, the roller coaster spends most of its time in high energy states (left). For ESH dynamics, the time spent in the region $(x, x+dx)$ is $1/\dot{x}$ which is exactly proportional to the Boltzmann sampling probability, $e^{-E(x)}$ (right plot). Thus, sampling the ESH dynamics over time is equivalent to sampling from the target distribution.

an unexpected form for the kinetic energy, $K(\mathbf{v})$, where $v^2 \equiv \mathbf{v} \cdot \mathbf{v}$.

$$K_{ESH}(\mathbf{v}) = {}^{d}\!/_{2} \log(v^2/d) \tag{1}$$

For contrast, HMC borrows the same form from Newtonian physics, $K_{HMC}(\mathbf{v}) = {}^{1}\!/_{2}\, v^2$ (with an intriguing connection in Appendix A.1). Given a Hamiltonian, its associated dynamics are defined as follows, using the dot notation for time derivatives and defining $\mathbf{g}(\mathbf{x}) \equiv \partial_{\mathbf{x}} E(\mathbf{x})$. We also suppress the time dependence, $\mathbf{x}(t), \mathbf{v}(t)$, unless needed for clarity.

$$\begin{array}{lll} \text{General Hamiltonian} & \text{ESH Dynamics} & \text{HMC/Newtonian Dynamics} \\ \dot{\mathbf{x}} = \partial_{\mathbf{v}} H(\mathbf{x}, \mathbf{v}) & \dot{\mathbf{x}} = \mathbf{v}/(v^2/d) & \dot{\mathbf{x}} = \mathbf{v} \\ \dot{\mathbf{v}} = -\partial_{\mathbf{x}} H(\mathbf{x}, \mathbf{v}) & \dot{\mathbf{v}} = -\mathbf{g}(\mathbf{x}) & \dot{\mathbf{v}} = -\mathbf{g}(\mathbf{x}) \end{array} \tag{2}$$

Hamiltonian dynamics have a number of useful properties. First, they conserve energy. Secondly, via Liouville's theorem we see that they are volume preserving (or more generally, *symplectic*). Finally, the dynamics are reversible. ESH dynamics look *nearly* identical to HMC dynamics, except for the scaling term for $\dot{\mathbf{x}}$ that depends on the magnitude of the velocity. Note that $v^2 = 0$ is a singularity in the dynamics in Eq. 2. For 1-D dynamics, this is an issue, as the velocity can never change signs. However, in higher dimensions, dynamics always avoid the singularity (Appendix A.2).

Imagine riding a roller-coaster running infinitely around a circular track shown in Fig. 2. $E(x)$ is the potential energy from gravity at a point in the loop specified by a scalar angle $x \in [0, 2\pi]$. At the bottom, potential energy is small while kinetic energy is large. Under Newtonian physics the speed is slowest at the top so most of a rider's time is spent in dreadful anticipation of the fall, before quickly whipping through the bottom. For the ESH kinetic energy, the dynamics is altered. When the rider's "speed" (magnitude of the "velocity/momentum") is maximum at the bottom, the updates in position are slowed down accordingly. That is, the roller-coaster goes into slow motion at the bottom and then goes into fast-forward at the top of the track where the "speed" is low. A spectator will see the rider spending most of their time at the bottom, with low potential energy, and the time spent in each region is exactly proportional to the Boltzmann distribution, as we now show.

2.1 Ergodic ESH Dynamics Sample $p(\mathbf{x}) = e^{-E(\mathbf{x})}/Z$

The reason for defining the non-standard Hamiltonian dynamics with a logarithmic kinetic energy is that the uniform distribution over all states with some fixed, constant energy, c, in the joint space gives us samples from the energy distribution in the x-space.

$$p(\mathbf{x}, \mathbf{v}) = \delta(E(\mathbf{x}) + K_{ESH}(\mathbf{v}) - c)/Z' \qquad \rightarrow \qquad p(\mathbf{x}) = e^{-E(\mathbf{x})}/Z \tag{3}$$

The proof is as follows.

$$\begin{aligned} p(\mathbf{x}) &= \int d\mathbf{v}\, p(\mathbf{x}, \mathbf{v}) = 1/Z' \int d\mathbf{v}\, \delta\left(E(\mathbf{x}) + d/2 \log v^2/d - c\right) \\ &= 1/Z' \int J(\boldsymbol{\phi})\, d\boldsymbol{\phi}\, \rho^{d-1}\, d\rho\, \delta\left(E(\mathbf{x}) + d \log \rho - c - d/2 \log d\right) \\ &= 1/Z' \int J(\boldsymbol{\phi})\, d\boldsymbol{\phi}\, \rho^{d-1}\, d\rho\, \rho/d\, \delta(\rho - e^{-E(\mathbf{x})/d + c/d + {}^{1}\!/_{2} \log d}) = e^{-E(\mathbf{x})}/Z \qquad \square \end{aligned}$$

In the second line, we switch to hyper-spherical coordinates, with $\rho = |\mathbf{v}|$ and angles combined in $\boldsymbol{\phi}$. In the third line, we use an identity for the Dirac delta that for any smooth function, $f(z)$, with one simple root, z^*, we have $\delta(f(z)) = \delta(z - z^*)/|f'(z^*)|$ [23]. Finally we integrate over ρ using the Dirac delta, with the ϕ integral contributing constants absorbed into the normalizing constant.

This proof resembles results from molecular dynamics [15], where an auxiliary scalar "thermostat" variable is integrated over to recover a "canonical ensemble" in both the coordinates and momentum. In our case, the momentum variables are already auxiliary so we directly use the velocity variables in a role usually filled by the thermostat. The form of the energy is inspired by the log-oscillator [24]. Although log-oscillator thermostats never gained traction in molecular dynamics because they violate several physical properties [25], this is not relevant for machine learning applications.

Although Eq. 3 gives us the target distribution as the marginal of a uniform distribution over constant energy states, this is not helpful at first glance because it is not obvious how to sample the uniform

distribution either. At this point, we invoke the ergodic hypothesis: the dynamics will equally occupy all states of fixed energy over time. More formally, the (*phase space*) average over accessible states of equal energy is equal to the *time* average under Hamiltonian dynamics for sufficiently long times.

$$\textit{Ergodic hypothesis}: \quad \frac{1}{Z'}\int d\mathbf{x}d\mathbf{v}\ \delta(H(\mathbf{x},\mathbf{v})-c)\ f(\mathbf{x},\mathbf{v}) = \lim_{T\to\infty}\frac{1}{T}\int_0^T dt f(\mathbf{x}(t),\mathbf{v}(t)) \quad (4)$$

The importance and frequent appearance of ergodicity in physical and mathematical systems has spurred an entire field of study [26] with celebrated results by Birkhoff and von Neumann [27, 28, 29] explaining why ergodicity holds under general conditions. Whether ergodicity holds for a specific system depends on the details of each system through $E(\mathbf{x})$, and the appearance of hidden invariants of the system breaks ergodicity [19, 30, 23]. While we do not expect hidden invariants to emerge in highly nonlinear energy functions specified by neural networks, systems can be empirically probed through fixed point analysis [20] or Lyapunov exponents [31]. Standard MCMC also makes an assumption of ergodicity, but the use of stochastic steps typically suffices to ensure ergodicity [3].

The key distinction to MCMC is that our scheme is fully deterministic, and therefore we use chaotic dynamics, rather than randomness, to ensure that the sampler explores the entire space. As an alternative to relying on ergodicity, we also give a normalizing flow interpretation in Sec. 2.3.

2.2 Numerical Integration of the Energy Sampling Hamiltonian (ESH) ODE

The leapfrog or Stormer-Verlet integrator [32] is the method of choice for numerically integrating standard Hamiltonian dynamics because the discrete dynamics are explicitly reversible and volume preserving. Additionally, the error of these integrators are $O(\epsilon^3)$ for numerical step size, ϵ. Volume preservation is not guaranteed by off-the-shelf integrators like Runge-Kutta [18]. The analogue of the leapfrog integrator for our proposed dynamics in Eq. 2 follows.

$$\begin{aligned}
\mathbf{v}(t+\epsilon/2) &= \mathbf{v}(t) - \epsilon/2\ \mathbf{g}(\mathbf{x}(t)) && \text{Newtonian/HMC} && \text{ESH}\\
\mathbf{x}(t+\epsilon) &= \mathbf{x}(t) + \epsilon\ s\ \mathbf{v}(t+\epsilon/2) && s=1 && s = d/v^2(t+\epsilon/2)\\
\mathbf{v}(t+\epsilon) &= \mathbf{v}(t+\epsilon/2) - \epsilon/2\ \mathbf{g}(\mathbf{x}(t+\epsilon)) && && \qquad (5)
\end{aligned}$$

Unfortunately, for ESH the effective step size for $\mathbf{x}$ can vary dramatically depending on the magnitude of the velocity. This leads to very slow integration with a fixed step size, as we will illustrate in the results. We consider two solutions to this problem: adaptive step sized Runge-Kutta integrators and deriving a leapfrog integrator in transformed coordinates. Although Runge-Kutta integrators do not give the same guarantees in terms of approximate conservation of the Hamiltonian as symplectic integrators, the Hamiltonian was stable in experiments (App. D.1). Moreover, looking at Eq. 3, we see that the exact value of the Hamiltonian is irrelevant, so results may be less sensitive to fluctuations.

Transforming the ODE The adaptive time-step in the Runge-Kutta ODE solver is strongly correlated to the magnitude of the velocity (App. D.1). This is not particularly surprising given the form of the dynamics, but it suggests that we might be able to make the integration more efficient if we chose a time-rescaling such that the optimal step size was closer to a constant. We chose a new time variable, $\bar{t}$, so that $dt = d\bar{t}\ |\mathbf{v}|/d$, leading to the transformed dynamics $\dot{\mathbf{x}} = \mathbf{v}/|\mathbf{v}|, \dot{\mathbf{v}} = -|\mathbf{v}|/d\ \mathbf{g}(\mathbf{x})$ (for notational convenience, we omit the bar over t and continue using dot notation to indicate derivative with respect to the scaled time). Next, we re-define the variables as follows: $\mathbf{u} \equiv \mathbf{v}/|\mathbf{v}|, r \equiv \log|\mathbf{v}|$. The transformed ODE follows.

$$\dot{\mathbf{x}} = \mathbf{u} \qquad \dot{\mathbf{u}} = -(\mathbb{I} - \mathbf{u}\mathbf{u}^T)\mathbf{g}(\mathbf{x})/d \qquad \dot{r} = -\mathbf{u}\cdot\mathbf{g}(\mathbf{x})/d \quad (6)$$

Interestingly, the previously troublesome magnitude of the velocity, captured by r, plays no role in the dynamics of $\mathbf{x}$. However, we must still solve for r because at the end of the integration, we need to re-scale the time coordinates back so that the time average in Eq. 4 can be applied. Again, we can solve this ODE with general-purpose methods like Runge-Kutta or with a leapfrog integrator. Correctly deriving a leapfrog integrator for the time-scaled ODE requires us to go back to first principles, but turns out to be well worth the effort.

Algorithm 1: Leapfrog integrator for the transformed ESH ODE Define the operator

$$\mathcal{O} \equiv \mathbf{u}\cdot\partial_{\mathbf{x}} - 1/d\ \mathbf{g}(\mathbf{x})^T(\mathbb{I} - \mathbf{u}\mathbf{u}^T)\partial_{\mathbf{u}} - 1/d\ \mathbf{u}\cdot\mathbf{g}(\mathbf{x})\partial_r \text{ and } \mathbf{z} \equiv (\mathbf{x},\mathbf{u},r).$$

Expanding the ODE $\dot{\mathbf{z}} = \mathcal{O}\mathbf{z}$ shows that it is equivalent to Eq. 6. The solution to this ODE can then formally be written as $\mathbf{z}(t) = e^{t\mathcal{O}}\mathbf{z}(0)$. The leapfrog comes from the following approximation [18].

$$e^{\epsilon(\mathcal{O}_1+\mathcal{O}_2)} = e^{\epsilon/2\mathcal{O}_2}e^{\epsilon\mathcal{O}_1}e^{\epsilon/2\mathcal{O}_2} + O(\epsilon^3)$$

Note that we cannot break up the exponent in the usual way because the operators do not commute. Instead the result above uses the Baker-Campbell-Hausdorff formula. If we set $\mathcal{O}_1 = \mathbf{u}\cdot\partial_{\mathbf{x}}$ then $\mathbf{x}(t+\epsilon) = e^{\epsilon\mathcal{O}_1}\mathbf{x}(t) = \mathbf{x}(t) + \epsilon\,\mathbf{u}(t)$, as in the standard leapfrog formula, Eq. 5. This can be seen by expanding the exponential as a Taylor series and noting that terms second order and above are zero. The derivation of the step for $\mathbf{u}, r$ is more complicated and is shown in App. B.1.

$$\begin{aligned}
\mathbf{u}(t+\epsilon) &= \frac{\mathbf{u}(t) + \mathbf{e}\left(\sinh\left(\epsilon\,|\mathbf{g}|/d\right) + \mathbf{u}(t)\cdot\mathbf{e}\cosh\left(\epsilon\,|\mathbf{g}|/d\right) - \mathbf{u}(t)\cdot\mathbf{e}\right)}{\cosh\left(\epsilon\,|\mathbf{g}|/d\right) + \mathbf{u}(t)\cdot\mathbf{e}\sinh\left(\epsilon\,|\mathbf{g}|/d\right)} && \text{Using:} \\
r(t+\epsilon) &= r(t) + \log(\cosh\left(\epsilon\,|\mathbf{g}|/d\right) + \mathbf{u}\cdot\mathbf{e}\sinh\left(\epsilon\,|\mathbf{g}|/d\right)) && \mathbf{g} \equiv \mathbf{g}(\mathbf{x}(t)) \qquad (7) \\
\mathbf{x}(t+\epsilon) &= \mathbf{x}(t) + \epsilon\,\mathbf{u}(t+\epsilon/2) && \mathbf{e} \equiv -\mathbf{g}/|\mathbf{g}|
\end{aligned}$$

The direction of the update for $\mathbf{u}$ is towards the direction of gradient descent, $\mathbf{e}$. If the norm of the gradient is large, $\mathbf{u} \to \mathbf{e}$. This is convenient as it allows the dynamics to perform like gradient descent when samples are far from the low energy landscape. For leapfrog integration, we have to sandwich a full step in $\mathbf{x}$ between half steps in $\mathbf{u}, r$ as in Eq. 5 and written out explicitly in App. C.

Algorithm 2: Ergodic sampling with the transformed ODE The solution we get from iterating Eq. 7 gives us $\mathbf{x}(\bar{t}), \mathbf{u}(\bar{t}), r(\bar{t})$ at discrete steps for the scaled time variable, $\bar{t} = 0, \epsilon, 2\epsilon, \ldots$, where we re-introduce the bar notation to distinguish scaled and unscaled solutions. We can recover the original time-scale by numerically integrating the scaling relation, $dt = d\bar{t}\;|\mathbf{v}(\bar{t})|/d$ to get $t(\bar{t}) = \int_0^{\bar{t}} d\bar{t}'|\mathbf{v}(\bar{t}')|/d = \int_0^{\bar{t}} d\bar{t}' e^{r(\bar{t}')}/d$. For ergodic sampling, we pick a random $t \in [0, T]$, then linearly interpolate between grid points to find $\mathbf{x}(t)$. An unexplored alternative is reservoir sampling.

2.3 Algorithm 3: Jarzynski Sampling Using ESH Dynamics as a Normalizing Flow

Ergodic sampling is not convenient in high-dimensional spaces. For sampling images, for instance, we do not want to store the entire time-series $\mathbf{x}(t)$. For high-dimensional sampling, we introduce a more efficient method of sampling, by interpreting the ESH dynamics as a normalizing flow. The idea of interpreting Hamiltonian dynamics as a normalizing flow that can be weighted to sample a target distribution goes back to Radford Neal's unpublished [33] but influential [34, 12, 35] work on Hamiltonian Importance Sampling. Jarzynski's equality [36, 37] can be related to importance sampling [38] and has been applied to Hamiltonian dynamics [37, 23] which motivates our approach. Neal's Hamiltonian importance sampling required an annealing schedule to move the initial high energy Newtonian dynamics closer to the low energy target distribution. This is not required for ESH dynamics as our Hamiltonian directly samples the target.

We initialize our normalizing flow as $\mathbf{x}(0), \mathbf{v}(0) \sim q_0(\mathbf{x}, \mathbf{v}) = e^{-E_0(\mathbf{x})}/Z_0\; q_0(\mathbf{v})$, where E_0 is taken to be a simple energy model to sample with a tractable partition function, Z_0, like a unit normal. Then we transform the distribution using deterministic, invertible ESH dynamics to $q_t(\mathbf{x}(t), \mathbf{v}(t))$. The change of variables formula for the ODE in Eq. 6 (App. B.2) is

$$\log q_t(\mathbf{x}(t), \mathbf{v}(t)) = \log q_0(\mathbf{x}(0), \mathbf{v}(0)) + \log|\mathbf{v}(t)| - \log|\mathbf{v}(0)|.$$

Then, using importance sampling, we can relate q_t to the target distribution $p(\mathbf{x}) = e^{-E(\mathbf{x})}/Z$.

$$\begin{aligned}
\mathbb{E}_{q_t(\mathbf{x}(t),\mathbf{v}(t))}[e^{w(\mathbf{x}(t),\mathbf{v}(t))}\; f(\mathbf{x}(t))] &= \frac{Z}{Z_0}\mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})] \qquad (8) \\
w(\mathbf{x}(t), \mathbf{v}(t)) &\equiv E_0(\mathbf{x}(0)) - E(\mathbf{x}(0)) + \log|\mathbf{v}(t)| - \log|\mathbf{v}(0)|
\end{aligned}$$

For the weights, we should formally interpret $\mathbf{x}(0), \mathbf{v}(0)$ as functions of $\mathbf{x}(t), \mathbf{v}(t)$ under the inverse dynamics. This expression holds for any f, including $f(\mathbf{x}) = 1$, which gives us the partition function ratio $\mathbb{E}_{q_t}[e^w] = Z/Z_0$. We use this relation to replace Z/Z_0 when calculating expectations, which is known as self-normalized importance sampling. Note that as in ergodic sampling in the time-scaled coordinates, the weight is proportional to $|\mathbf{v}(t)|$. We give a full derivation and an example using ESH-Jarzynski flows to estimate partition functions in App. B.2 and give detailed specification for all algorithms with implementation notes and link to code in App. C.

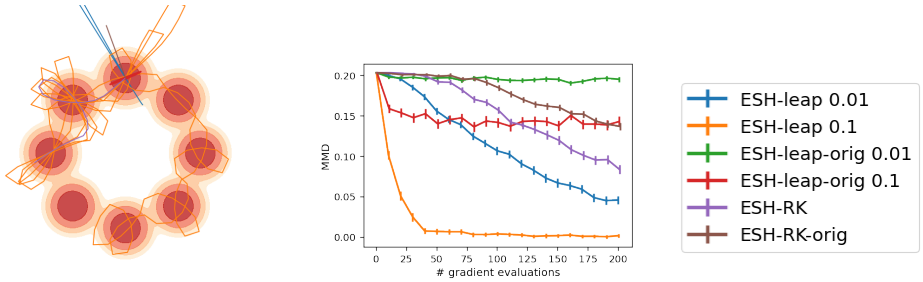

Figure 3: (Left) A single chain integrated with 200 gradient evaluations with various ESH ODE solvers on the 2D MOG energy function. (Right) Maximum Mean Discrepancy (MMD) as a function of the number of gradient evaluations. Solutions to the original, unscaled ODE are labeled `orig`.

3 Results

ESH dynamics provide a fundamentally different way to sample distributions, so we would like experiments to build intuition about how the approach compares to other sampling methods, before considering applications like sampling neural network energy functions. We introduce some standard synthetic benchmarks along with variations to help contrast methods.

- *Mixture of Gaussian* (2D MOG) is a mixture of 8 Gaussians with separated modes.
- *Mixture starting with informative prior* (2D MOG-prior) is the same as 2D MOG but initializes samplers from a mode of the distribution. This simulates the effect of using "informative priors"[22] to initialize MCMC chains, as in persistent contrastive divergence (PCD) [39]. The danger of informative initialization is that samplers can get stuck and miss other modes.
- *Ill Conditioned Gaussian* (50D ICG) from [3] has different length scales in different dimensions.
- *Strongly Correlated Gaussian* (2D SCG) with a Pearson correlation of 0.99.
- *Strongly Correlated Gaussian with bias* (2D SCG-bias) is the same as 2D-SCG except we bias the initialization toward one end of the long narrow distribution (as in Fig. 5). This tests the ability of samplers to navigate long, low energy chasms.
- *Funnel* (20D Funnel) A challenging test case where the length scale in one region is exponentially smaller than other regions. Our implementation is from [10].

Metrics Our goal is to obtain high quality samples from the target distribution. With the rise of multi-core CPUs, GPUs, and cloud computing, most practitioners prefer to simulate many Markov chains in parallel for the minimum amount of time, rather than to simulate one chain for a very long time [40, 21]. Therefore, we imagine running a number of samplers in parallel, and then we measure how well they resemble the target distribution after some time, as a function of *number of serial gradient evaluations* per chain. For neural network energy functions, gradient evaluation dominates the computational cost. To measure how well the samples resemble true samples from the distribution we use the (unbiased) kernel estimator of the squared **Maximum Mean Discrepancy (MMD)** [41], with Gaussian kernel with bandwidth chosen via the median heuristic. MMD is easy to compute with minimal assumptions, converges to zero if the sampling distributions are indistinguishable, and has high statistical power. In line with previous work, we also calculate the popular **Effective Sample Size (ESS)** using the method in [4]. We caution readers that ESS is a measure of the variance of our estimate *assuming* we have sampled the Markov chain long enough to get unbiased samples. For this reason, many MCMC samplers throw away thousands of initial steps in a "burn-in" period. While we perform our experiments without burn-in in the transient regime, ESS can be interpreted as a proxy for mixing speed, as it depends on the auto-correlation in a chain.

3.1 Comparing ESH Integrators

For ESH dynamics, we compare a variety of methods to solve the dynamics. First of all, we consider solving the original dynamics (`orig`), Eq. 2, versus the time-scaled dynamics, Eq. 6. For each dynamics, we compare using an adaptive Runge-Kutta (`RK`) solver [42] (fifth order Dormand-Prince [43]) to the leapfrog solvers (`leap`) in Eq. 5 and 7 respectively. In Fig. 3 the scaled dynamics are preferable to the original and the leapfrog integrator is preferable to Runge-Kutta. Sec. D.1 confirms that the leapfrog integrator for the scaled ODE is the best approach across datasets.

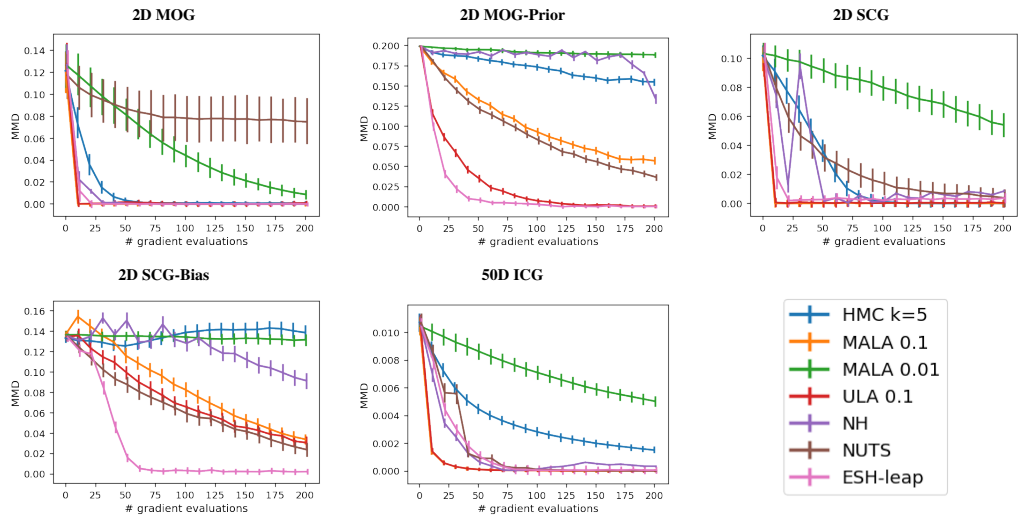

Figure 4: Maximum Mean Discrepancy (MMD) as a function of the number of gradient evaluations performed for different samplers.

For leapfrog integrators, there is only one hyper-parameter, the step size, and for Runge-Kutta integrators we need to choose the tolerance. In experiments we set these to be as large as possible on a logarithmic grid without leading to numerical errors. This leads us to use a value of $\epsilon = 0.1$ in subsequent experiments. For the Runge-Kutta integrators we were forced to use rather small sizes for the relative tolerance (10^{-5}) and absolute tolerance (10^{-6}) to avoid numerical errors. Since the Hamiltonian should be conserved, it is a good measure of a solver's stability. We plot the Hamiltonian for different solvers in App. D.1. Monitoring the error in the Hamiltonian could potentially make an effective scheme for adaptive updating of the step size. The scaled solver makes a step of fixed distance in the input space, but for distributions with varying length scales, this may be undesirable.

3.2 Comparing ESH with Other Samplers

We compare with the following sampling methods in our experiments that also use gradient information to sample from energy models.

- *Metropolis-Adjusted Langevin Algorithm* (MALA) [44] is a popular sampling method that uses gradient information in the Langevin step, along with a Metropolis-Hastings rejection step to ensure convergence to the target distribution.
- *Unadjusted Langevin Algorithm* (ULA) skips the Metropolis-Hastings rejection step, with the argument that if the step size is made small enough, no rejections will be necessary [6].
- *Hamiltonian Monte Carlo* (HMC) [3] uses Hamiltonian dynamics to propose Markov steps that traverse a large distance while still having a high likelihood of being accepted. In the experiments we use $k = 5$ steps to see if it improves over MALA/ULA. If more steps are beneficial, we expect this to be discovered by automatic hyper-parameter selection using NUTS.
- *No-U-Turn Sampler* (NUTS) [4] is a version of HMC with automatic hyper-parameter selection.
- *Nosé-Hoover Thermostat* (NH) is a deterministic dynamics in an extended state space that introduces a "thermostat" variable in addition to velocity. Based on the original works of Nosé[15] and Hoover [17], we used a general numerical integration scheme proposed by Martyna et al. [20] with the specific form taken from [45].

Results are summarized in Fig. 4 and Table 3.2. Langevin dynamics give a strong baseline. Nosé-Hoover and HMC give mixed results, depending on the energy model. While automatic hyper-parameter selection with NUTS is useful in principle, the overhead in gradient computation makes this approach uncompetitive in several cases. ESH is competitive in all cases and a clear favorite in certain situations. Fig. 5 illustrates why ESH is particularly effective for the biased initialization examples. Langevin dynamics have a hard time navigating long low energy chasms quickly because of random walk behavior. On the other hand, far from deep energy wells with large gradients, Langevin makes large steps, while ESH is limited to constant length steps in the input space. For this

Table 1: Effective Sample Size (with Standard Deviation)

Sampler Dataset	ESH-leap	HMC k=5	MALA 0.1	NH	NUTS	ULA 0.1
20D Funnel	**1.0e-03** (3.0e-04)	9.6e-04 (1.6e-04)	8.8e-04 (1.1e-04)	9.4e-04 (1.5e-04)	**1.0e-03** (1.8e-04)	8.8e-04 (1.1e-04)
2D MOG	**2.1e-02** (1.6e-02)	2.5e-03 (1.6e-03)	4.1e-03 (1.3e-03)	3.6e-03 (1.3e-03)	2.2e-03 (2.8e-03)	8.8e-03 (4.6e-03)
2D MOG-prior	**2.6e-02** (1.8e-02)	3.0e-03 (8.4e-04)	4.2e-03 (1.5e-03)	2.7e-03 (3.8e-04)	4.6e-03 (2.0e-03)	8.5e-03 (4.5e-03)
2D SCG	**2.4e-02** (1.4e-02)	7.6e-03 (4.8e-03)	1.3e-02 (8.0e-03)	1.0e-02 (5.7e-03)	1.2e-02 (1.1e-02)	1.3e-02 (8.1e-03)
2D SCG-bias	**8.9e-03** (1.2e-02)	9.6e-04 (2.1e-03)	2.9e-03 (5.1e-03)	1.8e-03 (3.5e-03)	2.5e-03 (4.9e-03)	3.7e-03 (6.0e-03)
50D ICG	1.6e-04 (2.6e-04)	2.8e-05 (4.0e-05)	7.4e-04 (5.0e-04)	1.8e-04 (2.2e-04)	1.2e-04 (1.4e-04)	**7.8e-04** (5.1e-04)

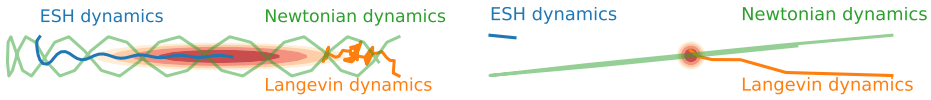

Figure 5: (Left) Navigating long energy valleys with 50 gradient evaluations. (Right) Entering deep energy wells with 5 gradient evaluations.

reason, ESH slightly lags Langevin in the ill-conditioned and strongly correlated Gaussians. While we may get the best of both worlds by smartly initializing ESH chains, we continue to investigate only the basic algorithm. The large scores for effective sample size in Table 3.2 show that ESH dynamics mix quickly. All methods, including ESH, failed on the Funnel example, see App. D.2

3.3 Sampling from Neural Network Energy Models

We consider sampling from a pre-trained neural network energy model, JEM [46]. Fig. 6 shows example chains for samplers starting from random initialization, and Fig. 13 shows samplers initialized from a replay buffer constructed during training. We also plot the average energy for a batch of examples. In both cases, ESH finds much lower energy samples than the algorithm used for training, ULA. Nosé-Hoover makes a surprisingly strong showing considering its poor performance on several synthetic datasets. We included an ESH leapfrog integrator with a larger step size of $\epsilon = 1$, which performed the best, especially with few gradient evaluations. HMC and MALA do poorly using noise initialization because most proposal steps are rejected. This is because JEM and similar works implicitly scale up the energy in order to take larger Langevin gradient steps that overcome random walk noise [47, 46, 22] as described in [21]. Those papers use the common argument that ULA approximates MALA for small step size [6], but with the large energy scaling employed this argument becomes dubious because the gap between ULA and MALA grows. ESH gives the best of both worlds as it avoids reject steps but still samples the target distribution.

3.4 Training Neural Energy-Based Models

We consider an energy model, $p(\mathbf{x}) = e^{-E_\theta(\mathbf{x})}/Z$, where $E_\theta(\mathbf{x})$ is specified by a neural network. Estimating gradients to train such a model to maximize the likelihood of the data requires sampling

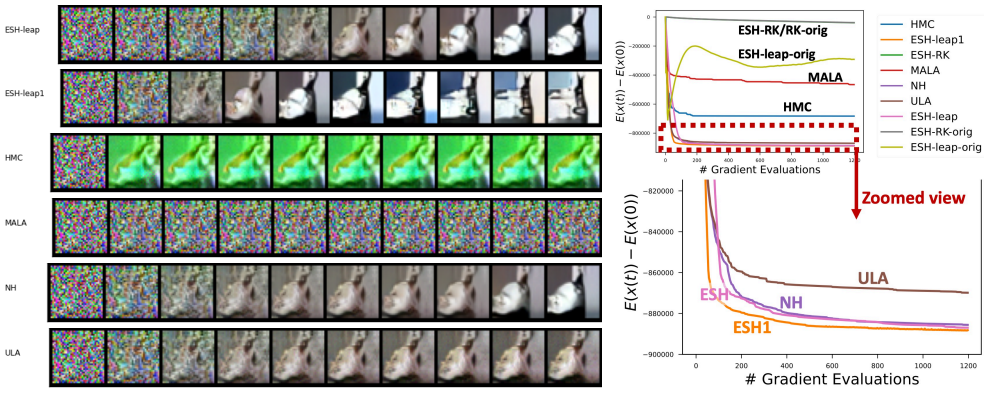

Figure 6: (Left) Example of sampling chains from random initialization with 200 gradient evaluations per method. (Right) Average energy over time for a batch of 50 samples using different samplers.

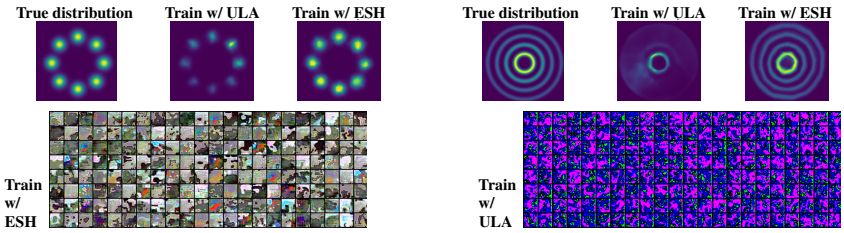

Figure 7: Results training a neural energy model using ULA versus ESH sampling. (Top) Energy landscape learned for synthetic datasets with 50k gradient evaluations. (Bottom) Examples from the training buffer using persistent contrastive divergence on CIFAR-10.

from the energy model (App. B.3). Fig. 7 visualizes the energy landscape learned from data for data sampled from a 2-D distribution. The architecture and hyper-parameters are taken from [21] and included in App. C.1. Samplers are initialized with uniform noise and then run for 50 steps per training iteration, using the Jarzynski sampler for ESH. Better sampling leads to better gradients and therefore the model trained with ESH sampling more crisply matches the true distribution.

We also trained a convolutional network energy model on CIFAR-10 (details in C.1). Informative priors like contrastive divergence and persistent contrastive divergence(PCD) [48, 39, 49] are typically used to speed up training, and our synthetic results suggest ESH may be especially beneficial in this setting. Using PCD with short sampling chains of 50 steps, ESH begins to learn visually plausible and diverse signals in just 30 epochs. Training with ULA, on the other hand, fails to learn anything with chains this short. See App. D.3 for more results.

4 Related Work

Over half a century of research on simulating Molecular Dynamics (MD) has focused on a core issue that also vexes machine learning: how to sample from an energy model. The methods employed in both fields have little overlap because MD is concerned with accurately simulating all physical properties of the target system, including momentum, dynamics of thermalization, and interaction between system and heat bath. Our experiments with Nosé-Hoover and other thermostats [15, 17] often oscillated or converged slowly because the thermostat variable is only weakly coupled to the momenta, which are then coupled to the only variables of interest in our case, the original coordinates. A recent review of MD research outlines different methods for constructing thermostats [50] including iso-kinetic thermostats [51] which have some similarities to ESH.

Ideas from physics like Hamiltonian dynamics and nonequilibrium thermodynamics have inspired many other ideas in machine learning [3, 33, 34, 35, 38, 12, 52, 53, 54, 55]. Another twist on HMC is Riemannian HMC [56], where $K(\mathbf{v}, \mathbf{x}) = {}^1\!/_2 \mathbf{v}^T M(\mathbf{x})\mathbf{v}$ still represents a Newtonian momentum but in curved space. This approach requires second order information like Hessians to define the curvature, and we only considered first order approaches in this paper. Another promising line of research recognizes that many of the properties that make Hamiltonian dynamics useful for MCMC come from being part of the more general class of involutive operators [57].

5 Conclusion

We presented a new approach for sampling based on deterministic, invertible dynamics and demonstrated its benefits for machine learning applications. Removing stochasticity leads to faster mixing between modes and could enable applications where backpropagation through the sampler is desired. Although ESH outperformed Langevin dynamics in direct comparisons on neural energy models, it remains to be seen if this advantage will finally allow energy-based models to beat competing approaches based on score matching [58] or auxiliary models [13] when trained at industrial scales. While we introduced an ergodic assumption to motivate the approach, we also provided an intuitive normalizing flow interpretation that does not require ergodicity. ESH dynamics provide a simple, fast, and deterministic drop-in replacement for sampling methods like HMC and Langevin dynamics with the potential to impact a wide range of machine learning applications.

References

[1] Edwin T Jaynes. *Probability theory: the logic of science*. Cambridge university press, 2003.

[2] Jack Dongarra and Francis Sullivan. Guest editors' introduction: The top 10 algorithms. *IEEE Annals of the History of Computing*, 2(01):22–23, 2000.

[3] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[4] Matthew D Hoffman and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

[5] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: a probabilistic programming language. *Grantee Submission*, 76(1):1–32, 2017.

[6] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[7] Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

[8] Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. Generalizing hamiltonian monte carlo with neural networks. *arXiv preprint arXiv:1711.09268*, 2017.

[9] Jianwen Xie, Yang Lu, Ruiqi Gao, Song-Chun Zhu, and Ying Nian Wu. Cooperative training of descriptor and generator networks. *IEEE transactions on pattern analysis and machine intelligence*, 42(1):27–45, 2018.

[10] Zengyi Li, Yubei Chen, and Friedrich T Sommer. A neural network mcmc sampler that maximizes proposal entropy. *Entropy*, 23(3):269, 2021.

[11] Yunfu Song and Zhijian Ou. Learning neural random fields with inclusive auxiliary generators. *arXiv preprint arXiv:1806.00271*, 2018.

[12] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226. PMLR, 2015.

[13] Will Grathwohl, Jacob Kelly, Milad Hashemi, Mohammad Norouzi, Kevin Swersky, and David Duvenaud. No mcmc for me: Amortized sampling for fast and stable training of energy-based models. *arXiv preprint arXiv:2010.04230*, 2020.

[14] Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

[15] Shūichi Nosé. A molecular dynamics method for simulations in the canonical ensemble. *Molecular physics*, 52(2):255–268, 1984.

[16] Ben Leimkuhler and Charles Matthews. *Molecular Dynamics.* Springer, 2016.

[17] William G Hoover. Canonical dynamics: Equilibrium phase-space distributions. *Physical review A*, 31(3):1695, 1985.

[18] Haruo Yoshida. Construction of higher order symplectic integrators. *Physics letters A*, 150(5-7):262–268, 1990.

[19] Glenn J Martyna, Michael L Klein, and Mark Tuckerman. Nosé–hoover chains: The canonical ensemble via continuous dynamics. *The Journal of chemical physics*, 97(4):2635–2643, 1992.

[20] Glenn J Martyna, Mark E Tuckerman, Douglas J Tobias, and Michael L Klein. Explicit reversible integrators for extended systems dynamics. *Molecular Physics*, 87(5):1117–1157, 1996.

[21] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run mcmc toward energy-based model. In *Advances in Neural Information Processing Systems*, pages 5233–5243, 2019.

[22] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. *arXiv preprint arXiv:1903.12370*, 2019.

[23] Michel A Cuendet. The jarzynski identity derived from general hamiltonian or non-hamiltonian dynamics reproducing nvt or npt ensembles. *The Journal of chemical physics*, 125(14):144109, 2006.

[24] Michele Campisi, Fei Zhan, Peter Talkner, and Peter Hänggi. Logarithmic oscillators: ideal hamiltonian thermostats. *Physical review letters*, 108(25):250601, 2012.

[25] Puneet Kumar Patra and Baidurya Bhattacharya. Zeroth law investigation on the logarithmic thermostat. *Scientific reports*, 8(1):1–11, 2018.

[26] Peter Walters. *An introduction to ergodic theory*, volume 79. Springer Science & Business Media, 2000.

[27] George D Birkhoff. Proof of the ergodic theorem. *Proceedings of the National Academy of Sciences*, 17(12):656–660, 1931.

[28] J v Neumann. Proof of the quasi-ergodic hypothesis. *Proceedings of the National Academy of Sciences*, 18(1):70–82, 1932.

[29] Calvin C Moore. Ergodic theorem, ergodic theory, and statistical mechanics. *Proceedings of the National Academy of Sciences*, 112(7):1907–1911, 2015.

[30] Paul F Tupper. Ergodicity and the numerical simulation of hamiltonian systems. *SIAM Journal on Applied Dynamical Systems*, 4(3):563–587, 2005.

[31] Puneet Kumar Patra, Julien Clinton Sprott, William Graham Hoover, and Carol Griswold Hoover. Deterministic time-reversible thermostats: chaos, ergodicity, and the zeroth law of thermodynamics. *Molecular Physics*, 113(17-18):2863–2872, 2015.

[32] Loup Verlet. Computer" experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical review*, 159(1):98, 1967.

[33] Radford M Neal. Hamiltonian importance sampling. In *talk presented at the Banff International Research Station (BIRS) workshop on Mathematical Issues in Molecular Dynamics*, 2005.

[34] Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. Hamiltonian variational auto-encoder. *arXiv preprint arXiv:1805.11328*, 2018.

[35] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015.

[36] Christopher Jarzynski. Nonequilibrium equality for free energy differences. *Physical Review Letters*, 78(14):2690, 1997.

[37] Christopher Jarzynski. Equalities and inequalities: Irreversibility and the second law of thermodynamics at the nanoscale. *Annu. Rev. Condens. Matter Phys.*, 2(1):329–351, 2011.

[38] Michael Habeck. Model evidence from nonequilibrium simulations. In *Advances in Neural Information Processing Systems*, pages 1753–1762, 2017.

[39] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071, 2008.

[40] Jackson Gorham and Lester Mackey. Measuring sample quality with stein's method. *arXiv preprint arXiv:1506.03039*, 2015.

[41] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[42] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[43] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[44] Peter J Rossky, JD Doll, and HL Friedman. Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978.

[45] Dana S Kleinerman, Cezary Czaplewski, Adam Liwo, and Harold A Scheraga. Implementations of nosé–hoover and nosé–poincaré thermostats in mesoscopic dynamic simulations with the united-residue model of a polypeptide chain. *The Journal of chemical physics*, 128(24):06B621, 2008.

[46] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[47] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

[48] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40. Citeseer, 2005.

[49] Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models. *arXiv preprint arXiv:2012.01316*, 2020.

[50] M Sri Harish and Puneet Kumar Patra. Temperature and its control in molecular dynamics simulations. *Molecular Simulation*, pages 1–29, 2021.

[51] Gary P Morriss and Carl P Dettmann. Thermostats: analysis and application. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(2):321–336, 1998.

[52] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[53] Jascha Sohl-Dickstein, Mayur Mudigonda, and Michael R DeWeese. Hamiltonian monte carlo without detailed balance. *arXiv preprint arXiv:1409.5191*, 2014.

[54] Jascha Sohl-Dickstein, Eric A Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, 2015.

[55] Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. *arXiv preprint arXiv:2002.06707*, 2020.

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

[57] Kirill Neklyudov, Max Welling, Evgenii Egorov, and Dmitry Vetrov. Involutive mcmc: a unifying framework. In *International Conference on Machine Learning*, pages 7273–7282. PMLR, 2020.

[58] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] The main limitation is the ergodic assumption discussed in Sec. 5 and 2, with an alternative sampling method that does not rely on ergodicity also provided. We also illustrated in Fig. 5 a potential drawback of our approach compared to Langevin dynamics.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] The ergodic assumption is stated in Eq. 4
   (b) Did you include complete proofs of all theoretical results? [Yes] The main result with proof is Eq. 3 in Sec. 2
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] A github link with code and methods to generate or download standard datasets is included in the paper, and the code is included in the supplementary material zip file for submission.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. C
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec. C.3
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. C.4
   (b) Did you mention the license of the assets? [Yes] See Sec. C.4
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]