Neural Langevin Dynamics: Towards Interpretable Neural Stochastic Differential Equations

Abstract

Neural Stochastic Differential Equations (NSDEs) have been trained as both Variational Autoencoders, and as GANs. However, the resulting Stochastic Differential Equations can be hard to interpret or analyse due to the generic nature of the drift and diffusion fields. By restricting our NSDE to be of the form of Langevin dynamics, and training it as a VAE, we obtain NSDEs that lend themselves to more elaborate analysis and to a wider range of visualisation techniques than a generic NSDE. More specifically, we obtain an energy landscape, the minima of which are in one-to-one correspondence with latent states underlying the used data. This not only allows us to detect states underlying the data dynamics in an unsupervised manner but also to infer the distribution of time spent in each state according to the learned SDE. In general, restricting an NSDE to Langevin dynamics enables the use of a large set of tools from computational molecular dynamics for the analysis of the obtained results.

1 Introduction

In recent years there has been widespread interest in Neural SDEs for generative modelling. Neural SDEs (NSDEs) learn the parameters in stochastic differential equations, which are natural extensions of ordinary differential equations to situations where uncertainty arises from many small and unobserved interactions, such as stock prices and molecular dynamics [20, 14]. As such, Neural SDEs can be seen as incorporating uncertainty in Neural Ordinary Differential Equations.

NSDEs are said to combine the strengths of deep learning and classical modelling [14, 11]. Deep learning brings performance and capacity, whereas classical modelling derives a lot of value from its interpretability. However, although from some perspective the the dynamical system learned by an NSDE is itself a classical model, it is still a very complicated object, which makes it questionable whether NSDEs actually provide interpretability.

Indeed it is often overlooked in Neural Differential Equation literature that finding the differential equation is only the start of the process. For interpretation, one needs to analyse stability, find stationary solutions, and analyse asymptotic behaviour, to name a few. In the particular case of NSDEs, one would want to find basins of attraction — as they could correspond to discrete states in the dynamics — and would like to extract how long the system spends in each state and at what rates transitions occur. For general dynamical systems, these tasks are prohibitively complicated, which severely hampers interpretability.

To make NSDEs more interpretable, we introduce Neural Langevin Dynamics (NLD), in which we replace the generic vector field, which serves as the drift, by a gradient field, and let the system evolve by Langevin dynamics. This has the following advantages.

There is a direct interpretation of “states” in the original system as the basins of attraction around local minima of the energy (e.g. [19]) These local minima can be found using the vast toolbox of scalar optimisation methods — e.g. [24, 18, 30, 12]. Sub-level sets of the energy function can then be used as approximations of the basins of attraction. This is illustrated by e.g. [Figure 2f] in which the contour lines clearly define regions that belong to specific states.

Moreover, there exist many more tools for visualising scalar functions than for visualising vector fields, aiding in the understanding by users of these models.

In this work, after introducing NLD in more detail, we show its potential and validity by training it
as a Variational Autoencoder (VAE) with a learned prior\(^1\)\(^2\)\(^3\)\(^4\)\(^5\) on an example problem. We show that the local minima indeed coincide with the true states underlying the data, and that information on the relative occurrence of states can be extracted from the learned energy landscape.

Our results indicate that NLD allows us to extract interpretable information about the data dynamics, and in such a way that it opens up the use of tools from optimization and analysis, effectively providing a bridge between deep learning and classical modelling.

\[ \text{2 Related Work} \]

The authors of \(^6\) view NSDEs as infinitely deep neural networks but don’t use them as time series models. In \(^7\), NSDEs are trained as a GANs instead of as VAEs, and the authors of \(^8\) train an NSDE in a GAN-related way. The authors of \(^9\) introduce the training of an NSDE as a VAE on sequential data. We use their training method to train our NLD models. The authors of \(^10\) also train NSDEs as VAEs but focus mostly on the dimensionality of the data space. To address problems around sample paths being far from observations during the start of training, the authors of \(^11\) devise a method of training NSDEs using importance sampling which could prove useful in the future to make the training of NLD models more stable. Another work addressing the learning of NSDE in data-space is\(^12\), where snapshots are used instead of trajectories.

The authors of \(^13\)\(^14\) extend the so-called adjoint sensitivity method to NSDEs and provide algorithms for storing Wiener processes. We do not use the adjoint sensitivity method but do use their algorithms for training NSDEs in the form of the torchsde package —https://github.com/google-research/torchsde—\(^15\).

NSDEs have found practical use, e.g. for modeling turbulence \(^16\) and for modeling quasar variability \(^17\).

NSDEs can be viewed in the wider context of neural differential equations. The seminal paper in this area was \(^18\). These were extended to models for sequential data in, among others \(^19\)\(^20\)\(^21\)\(^22\).

Moreover, NLD can be viewed in a line of physics-inspired neural network models such as those in\(^23\)\(^24\)\(^25\)\(^26\). The focus in the cited works however was more on performance and theoretical generalisability whereas our work focuses more on interpretability.

Langevin dynamics themselves have been extensively researched since the start of the 20th century and remain a topic of active research, see e.g. \(^27\)\(^28\)\(^29\) to learn more about this topic. Besides their relevance for physical modelling, they play an important role in MCMC sampling — see e.g. \(^30\).

\[ \text{3 Method} \]

\[ \text{3.1 Neural Stochastic Differential Equations} \]

When training Neural Stochastic Differential Equations as Variational Auto-Encoders on some data \(X = \{ x^{(i)} = \{ x^{(i)}_t \}_{t=0}^T | i = 1, \ldots, N \} \) as in \(^31\), one has two stochastic differential equations parameterising the prior \(p(\cdot | z_0)\) and approximate posterior \(q(\cdot | x, z_0)\) distributions:

\[ p(\cdot | z_0) \sim dz_t = h_\theta(z_t, t)dt + g_\theta(z_t, t)dW_t \]

\[ q(\cdot | x, z_0) \sim dz_t = f_\phi(z_t, t, c_t)dt + g_\theta(z_t, t)dW_t \]

with both differential equations starting in \(z_0\).

Here \(c_t\) is a function of the data \(x\) obtained by some encoder architecture, in this work a GRU

\[ \frac{1}{2} \int_0^T |u(z_t, t, c_t)|^2dt, \]

\[ g_\theta(z_t, t) u(z_t, t, c_t) = f_\phi(z_t, t, c_t) - h_\theta(z_t, t), \]

where \(T \) is the final time.

\[^1\]See Section 3 for further details.
Additionally, one has a decoding distribution \(p_{\theta}(x_1 | z_1)\), a prior distribution over the initial latent state \(p_{\theta}(z_0)\), and an approximate prior distribution over the initial latent state \(q_{\varphi}(z_0 | x)\).

Then the evidence lower bound (ELBO) can be written as

\[
\log p_{\theta}(x_0^{(i)} \ldots x_T^{(i)}) \geq \mathbb{E}_{z^{(i)} \sim q_{\varphi}(z|x^{(i)})} \left[ \sum_{t=0}^{T} p_{\theta}(x_t^{(i)} | z_t^{(i)}) - \frac{1}{2} \int_{0}^{T} |u(z_t, t, c_t)|^2 dt \right] - D_{KL}\left(q_{\varphi}(z_0 | x^{(i)}) \mid\mid p_{\theta}(z_0)\right),
\]

where \(q_{\varphi}(z | x) = q_{\varphi}(z | x, z_0)q_{\varphi}(z_0 | x)\).

To find the optimal parameters \(\theta, \varphi\), we can approximate the ELBO for a batch of datapoints by sampling from \(q_{\varphi}(z_0 | x)\) using the reparametrisation trick (16), and sampling \(z\) using an SDE solver such as the Euler-Maruyama method. To obtain gradients for the stochastic integral term, we can either back-propagate through the SDE solver or use the adjoint method (20 15). We can then use stochastic gradient ascent to optimise the ELBO.

Note that \(p_{\theta}(z)\) is a learned distribution, parameterised by an NSDE, representing the latent dynamics of the dataset as a whole. However, it is not easy to extract useful information about these latent dynamics from this representation of the distribution due to the generality of Equation (1).

### 3.2 Neural Langevin Dynamics

We introduce Neural Langevin Dynamics as a more interpretable alternative to NSDE. In NLD, we replace the drift \(H_\theta\) in Equation (1) by the scaled gradient of an energy \(E_\theta : \mathbb{R}^d \rightarrow \mathbb{R}\) that is given by a neural network, and modify the evolution to either overdamped or underdamped Langevin dynamics.

In the case of overdamped Langevin dynamics, the prior and posterior evolutions are determined by

\[
\begin{align*}
\mathrm{d}z_t &= -\gamma^{-1}\nabla z E_\theta(z_t)\mathrm{d}t + \sqrt{2\beta^{-1}\gamma^{-1}}\mathrm{d}W_t \\
\mathrm{d}z_t &= -\gamma^{-1}\nabla z E_\theta(z_t)\mathrm{d}t + f_\varphi(z_t, t, c_t)\mathrm{d}t + \sqrt{2\beta^{-1}\gamma^{-1}}\mathrm{d}W_t
\end{align*}
\]

whereas in the case of underdamped Langevin dynamics, they are determined by

\[
\begin{align*}
\mathrm{d}z_t^{q} &= M^{-1}z_t^{q}\mathrm{d}t \\
\mathrm{d}z_t^{p} &= \left[ -\nabla E_\theta(z_t^{q}) - \gamma M^{-1}z_t^{p} \right] \mathrm{d}t + \sqrt{2\gamma\beta^{-1}} \mathrm{d}W_t \\
\mathrm{d}z_t^{q} &= M^{-1}z_t^{q}\mathrm{d}t \\
\mathrm{d}z_t^{p} &= -\left[ \nabla E_\theta(z_t^{q}) + \gamma M^{-1}z_t^{p} \right] \mathrm{d}t + f_\varphi(z_t^{q}, z_t^{p}, t, c_t)\mathrm{d}t + \sqrt{2\gamma\beta^{-1}} \mathrm{d}W_t.
\end{align*}
\]

Here, \(M\) is a positive diagonal matrix representing mass. The constants \(M, \gamma, \beta\) can either be fixed or can be learned while training the model.

In the underdamped case, \(z^q\) plays the role of location, and \(z^p\) plays the role of momentum. Only the location, \(z^q\), should be used for decoding. Note that \(z^q\) follows a second-order stochastic differential equation.

One of the great advantages of using Langevin dynamics is that this way, the SDE of the prior distribution has a known stationary distribution of the form

\[
p_{\infty}(z) \sim \frac{\exp(-\beta E_\theta(z))}{\int_{\mathbb{R}^d} \exp(-\beta E_\theta(y)) \mathrm{d}y}
\]

in the overdamped case, and

\[
p_{\infty}(z_q, z_p) \sim \frac{\exp(-\beta E_\theta(z_q))}{\int_{\mathbb{R}^d} \exp(-\beta E_\theta(y)) \mathrm{d}y} \otimes \mathcal{N}(z_p; 0, M^{-1})
\]

in the underdamped case. Due to ergodicity, the proportion of time spent in a region of the latent space, in the long run, corresponds to the mass attributed to that region by the stationary distribution, which only relies on the \(\beta\) and \(E_\theta\).

### 4 Experimental setup

We evaluate the capability of our method to develop an interpretable representation of data dynamics by training a model on data coming from a Markov chain. We then analyse the energy landscape of this model to recover the discrete states of the Markov chain and infer the time spent in each state.

The data is generated as random walks on a graph with three nodes. At each time step, a signal is emitted as a random vector sampled from a 15-dimensional multivariate normal distribution with

\[\text{Again, in this work } z_0 \text{ only depends on } x_0, \text{ so that the model can operate online.}\]
5 Visualisation advantages of NLD

Visualisation is an important tool for obtaining human knowledge in general and can be useful for extracting knowledge from a trained model. To visualise trained models, if we only have a drift field, as is the case for a general NSDE, all we can really do is make a quiver plot of the drift field or plot the flow lines of the drift. For the two-dimensional case, this is shown in Figure 2a and Figure 2b.

As shown in Figure 2c, a quiver plot of a three-dimensional vector field is already rather hard to interpret. One can restrict these visualisation techniques to a plane to make them more understandable, as is shown for a flow-line plot in Figure 2d.

If on the other hand, the drift field of the model is a gradient field of an energy function, as is the case for the Markov process underlying the data.

mean and covariance matrix specified by the node the random walk is at.

The transition probability from any state to any different state is 0.025 at each time step, resulting in a stationary distribution (1/3, 1/3, 1/3), and an expected transition time of 20 time-steps. An overview of this is shown in Figure 1.

Code for both the generation of the dataset and the training of the models is provided in https://github.com/anongh4nldl/NLD-public.

![Figure 1: Graph of the Markov process underlying the data.](image)

![Figure 2: Visualisations of an NSDE (a-b) respectively an NLD (c-g) trained as a VAE.](image)

---

3At the time of submission, our GitHub repo has been flagged and made invisible. We are fighting this, but in case that is unsuccessful, here is a temporary download link for our code and data: [https://ufile.io/ez3pspdp0](https://ufile.io/ez3pspdp0)
Figure 3: Results of unsupervised sequence segmentation of the first test sequence by both an underdamped and an overdamped NLD model.

case with the NLD models, we can additionally visualise the energy function, e.g. by plotting a graph or making a contour plot. If the dimensionality of the latent space is more than three, making it impossible to just plot the graph or the contours, we can restrict ourselves to a two-dimensional plane within the larger space, as is shown in Figure 2e and Figure 2f, or to a three-dimensional subspace, resulting in a visualisation like the one in Figure 2g.

These additional tools can aid a practitioner in interpreting what the model has learned and translating the obtained results to either existing or new knowledge about the data.

6 Results

We evaluate whether the learned energy landscape accurately represents the distribution of states underlying the data.

The data-generating Markov chain has an invariant distribution of (1/3, 1/3, 1/3), and we investigate whether this distribution can be recovered from the prior in the trained Neural Langevin Dynamics network.

We identify the three states in the Markov chain with the three wells in the energy landscape. To evaluate whether the energy landscape accurately represents the distribution of states underlying the data, we estimate the distribution \( p = (p_1, p_2, p_3) \) over the three wells according to the learned prior in three different ways and compare the results to the invariant distribution of the Markov chain. The three ways of estimating \( p \) are:

(a) by taking a thousand samples from the stationary distribution of the prior SDE, performing gradient flow on them, and seeing what percentage of samples ends up in each well; (b) by taking the zeroth order approximation \( p_i \approx \frac{\exp(-\beta E_\theta(x_i))}{\sum_j \exp(-\beta E_\theta(x_j))} \) where \( x_i \) is the location of the local minimum in well \( i \); (c) by taking the second-order approximation:

\[
p_i \approx \alpha \frac{\exp(-\beta E_\theta(x_i))}{\rho_i} = \alpha \left( \sum_j \exp(-\beta E_\theta(x_j)) \right)^{-1},
\]

\[
\rho_i(x_i) = \left( (2\pi)^d \det(\beta^{-1} d^2 E_\theta(x_i)^{-1}) \right)^{-1/2},
\]

where \( d^2 E_\theta(x_i) \) is the Hessian of the energy function at \( x_i \), and \( d \) is the dimension of the latent space. This second-order approximation corresponds to approximating the three wells by a mixture of three Gaussians.

We report the \( \ell^2 \)-distance between \( p \) and \((1/3, 1/3, 1/3)\) for each approximation for the top five best performing models ranked by second-order approximation performance for both the underdamped and overdamped models in Table 1a. For a more easily interpretable representation of the results, we also report the first coordinate of \( p \) in Table 1b.

A second question is whether the discovered “states”, i.e. the energy wells, in the trained prior dynamics coincide with the ground-truth latent states in the data-generating Markov chain and whether the trained encoder network successfully maps unseen incoming data to the latent states investigated above. This is a measure of the degree to which the training has managed to model the whole dynamics. We generated 500 test sequences from the same three-state Markov chain and mapped them to the latent space using the encoder of each of the models used for Table 1a. Then we followed the gradient flow from each time step in each sequence of latent-space points to find out which well they belong to; this leads to a sequence of labels \((0, 1, 2)\). To compare these labels to the ground truth, we selected the permutation of state labels \((0, 1, 2)\) for which the overlap is maximised.
We then computed the mean and standard deviation of the accuracy over the 500 sequences. The results are shown in Table 1a.

We provide an example of the segmentation in Figure 3. Note that the models operate in an online fashion resulting in the encoded latent state lagging slightly behind the true latent state. Nonetheless, these results clearly show that the wells in the energy landscape coincide with the discrete states behind the data. Moreover, the models were trained on sequences of length 200, but are tested on sequences of length 500 and clearly generalise well.

### 7 Limitations and future research

There is an inherent limitation to the types of dynamics our method can represent. Langevin dynamics can not represent periodic behaviour. Filtering out periodic signals may be a feature. However, when this behaviour is not wanted, another term needs to be introduced in the drift. The addition of a divergence-free rotation field to the drift might alleviate this problem without sacrificing the interpretability of the energy landscape.

We had some trouble getting our NLD models to train consistently: the loss would become NaNs or jump several orders of magnitude. We encountered the same problems with NSDE models we trained on the same data, which indicates how involved the training of NSDEs is in general, when using path-based methods like [20]. There are several other methods of training NSDEs, such as [4], [23], and [11], but there is no clear overview of which method works best in what cases. A comparison of the various training strategies for NSDEs would be invaluable.

Moreover, neither the value of the loss on the training or validation set nor the values of any of the loss components were a clear indication of how well the learned landscape or vector field represented the dynamics underlying the data. This need not be a problem if there is a downstream task, like segmentation, that can be used to evaluate the quality of the learned dynamics. Nonetheless, this does restrict the applicability of the method.

### 8 Conclusion

To provide a more interpretable alternative to Neural Stochastic Differential Equations, we introduce Neural Langevin Dynamics, in which we replace the general drift term by the gradient of a trainable energy function and let the system evolve by Langevin dynamics. The gain in interpretability comes both from the better options for visualisations of scalar functions over general vectors fields, and the better options for downstream processing in the form of classical methods from optimisation and analysis. With these methods, we can extract important properties such as stationary distributions and discrete states. When we apply NLD to an example problem, it recovers the discrete states underlying the data. These results show promise that NLD is able to combine the performance of neural networks with the interpretability of classical differential equation based models.
References


