Variational Integrator Networks for Physically Structured Embeddings

Steindór Sæmundsson
Imperial College London

Alexander Terenin
Imperial College London

Katja Hofmann
Microsoft Research

Marc Peter Deisenroth
University College London

Abstract

Learning workable representations of dynamical systems is becoming an increasingly important problem in a number of application areas. By leveraging recent work connecting deep neural networks to systems of differential equations, we propose *variational integrator networks*, a class of neural network architectures designed to preserve the geometric structure of physical systems. This class of network architectures facilitates accurate long-term prediction, interpretability, and data-efficient learning, while still remaining highly flexible and capable of modeling complex behavior. We demonstrate that they can accurately learn dynamical systems from both noisy observations in phase space and from image pixels within which the unknown dynamics are embedded.

1 Introduction

Deep learning has revolutionized application areas, such as image classification and reinforcement learning, in part via its ability to obtain representations of data that generalize well and are useful for downstream tasks. Deep networks have accomplished this by simultaneously being highly expressive, yet capable of learning effectively from a finite amount of data. A key determinant in this efficiency is the inductive bias encoded by the architecture of the network, such as in convolutional networks for image data, as well as long short-term memory networks for text and other sequential data. These structural assumptions allow the network to learn efficiently, while still enabling it to capture complex relationships that are prohibitively

difficult to feature engineer or write down manually.

We are interested in applying such networks to dynamical systems governed by the laws of physics. Such systems are highly flexible and capable of modeling complex phenomena. However, they also possess inherent structure, such as conservation laws. In machine learning, this important structure is often ignored, due to the black-box nature of off-the-shelf algorithms. To perform well on a given task, deep neural networks must learn to conserve these quantities as effectively as possible. Owing to the precise form of their equations, such networks generally do not conserve these quantities exactly (Greydanus et al., 2019). Greydanus et al. (2019) demonstrated that this flaw harms the networks' capacity for accurate long-term prediction.

As a workaround, Greydanus et al. (2019) proposed to parameterize the dynamical system's Hamiltonian using a neural network, and to learn it directly from data. The specification of the Hamiltonian fully determines the dynamics. The equations of motion are then reconstructed from the learned Hamiltonian via standard techniques from mechanics. One downside to this approach is the black-box nature of the neural network, which makes it difficult to encode properties of the dynamical system, such as its constraints or symmetries. Lutter et al. (2019) propose an architecture that imposes Lagrangian mechanics, and is optimized to minimize the violation of the equations of motion. A similar idea is also used in (Raissi et al., 2019) to learn general non-linear differential equations from physics. A potential drawback of encoding physical plausibility through the loss function is the need for training data that reasonably covers the configuration space.

The continuous-time equations of motion for a dynamical system are given by a set of differential equations that can be derived from its Lagrangian via variational calculus. These equations encode the underlying physical properties, such as conservation laws. In parallel, a deep residual network can be viewed as an Euler discretization of a system of ordinary differential equations, see Haber and Ruthotto (2017), E (2017), and Chen et al. (2018).

In this paper, we aim to bridge the viewpoint of neural ODEs (Haber and Ruthotto, 2017; E, 2017; Chen et al., 2018; Chang et al., 2018; Ruthotto and Haber, 2018), where neural networks are seen as discretized dynamical systems, with the viewpoint of geometric embeddings (Chamberlain et al., 2017; Nickel and Kiela, 2017; Ganea et al., 2018; Davidson et al., 2018), which impose structure on an embedding space. When data is concentrated on a manifold, Falorsi et al. (2018) argued that it is crucial to ensure the embedding space has the same topology as this manifold, motivating Lie group variational auto-encoders (Falorsi et al., 2018; de Haan and Falorsi, 2018; Falorsi et al., 2019).

We propose to model the dynamical system using a deep neural network, whose architecture matches the discrete-time equations of motion governing the dynamical system. This allows us to re-interpret the embedding learned by the network as a dynamical system in its own right. We focus on a class of discretization methods called *variational integration* (Marsden et al., 2001). This gives rise to our proposed *variational integrator networks*: a class of flexible neural network architectures that encode physical laws and manifold constraints by preserving the underlying geometry inherent to physical systems. These properties promote accurate long-term prediction, interpretability and more efficient learning than is possible with comparable black-box function approximators.

We demonstrate their effectiveness on a number of tasks, including inferring dynamical systems from noisy observations, and from the pixels of images, both in an interpretable and data-efficient manner[1].

2 Variational Integrators

In this section, we review variational integrators (VIs), a general class of discretization methods for dynamical systems. We study physical dynamical systems over a configuration space $\mathcal{Q}$, with generalized positions and velocities denoted by $\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)$. The systems are governed by the principle of least action, specified via the Lagrangian $L(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$, and expressible in Hamiltonian form. A brief review of these and related concepts of classical mechanics is given in Appendix A.

VIs approximate the trajectory of a continuous-time dynamical system by discretizing its action integral

$$L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h) \approx \int_t^{t+h} L(\boldsymbol{q}(\tau), \dot{\boldsymbol{q}}(\tau)) \, \mathrm{d}\tau. \quad (1)$$

This is a discrete-time quadrature-based approximation, denoted by L^d, defined by $\boldsymbol{q}_t = \boldsymbol{q}(t)$ and $\boldsymbol{q}_{t+1} = \boldsymbol{q}(t+h)$

[1] Code available on GitHub: HTTPS://GITHUB.COM/ STEINDORINGI/VARIATIONAL_INTEGRATOR_NETWORKS

with step size h. From a Lagrangian perspective, we arrive at the discrete equations of motion

$$\frac{\partial L^d(\boldsymbol{q}_{t-1}, \boldsymbol{q}_t, h)}{\partial \boldsymbol{q}_t} + \frac{\partial L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h)}{\partial \boldsymbol{q}_t} = \boldsymbol{0}, \quad (2)$$

by using a discrete analog of Hamilton's principle (Marsden et al., 2001). Following West (2004), (2) can be written in position-momentum form as

$$\boldsymbol{p}_t = -\frac{\partial L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h)}{\partial \boldsymbol{q}_t}, \quad \boldsymbol{p}_{t+1} = \frac{\partial L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h)}{\partial \boldsymbol{q}_{t+1}}, \quad (3)$$

where $\boldsymbol{p}_t = \partial L / \partial \dot{\boldsymbol{q}}_t$ are generalized momenta.

VIs are *symplectic* as they conserve phase-space volume exactly. Symplectic integrators also approximately conserve energy, often only introducing third-order (and above) discretization error with respect to the energy. Such integrators yield discrete-time dynamical systems that closely resemble the continuous-time systems under study, and evolve in a way that is globally consistent with the true solution.

VIs are also *momentum-preserving*. This means that for any symmetry in the discrete system, there is a quantity that is exactly conserved. These properties help to ensure their accuracy. In the dissipative and forced cases, VIs have been both theoretically and empirically shown to produce stable long-term predictions and to capture statistically important quantities, even in chaotic regimes (Lew et al., 2004).

3 Variational Integrator Networks

To define a variational integrator network, we begin with the viewpoint of neural ODEs (Haber and Ruthotto, 2017; Chen et al., 2018). In this setting, we specify an ODE whose right-hand-side is a single-layer neural network. We then obtain a deep residual network using an Euler discretization scheme, where the depth of the network is determined by the number of discretization steps.

We mirror this viewpoint with the goal of developing network architectures that learn dynamical systems faithfully, by having their learned embeddings be dynamical systems in their own right. Compared to neural ODEs, we introduce two key differences.

1. Rather than constructing a free-form system of ODEs, we construct a system of ODEs arising from the Euler-Lagrange equations governing a free-form dynamical system.

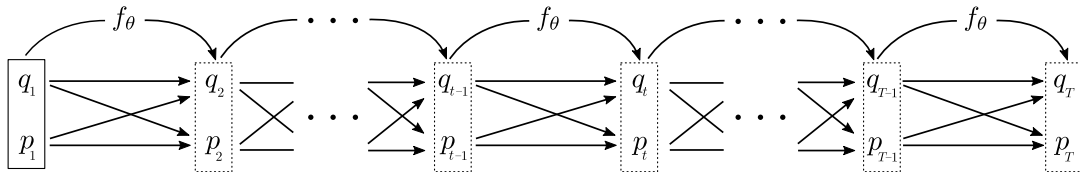2. Instead of an Euler discretization, we use a structure-preserving discretization given by a VI.

Steindór Sæmundsson, Alexander Terenin, Katja Hofmann, Marc Peter Deisenroth

Figure 1: Variational integrator network. Here, $(\boldsymbol{q}, \boldsymbol{p})$ are the hidden states, and f_θ is a residual block. The full variational integrator network is built by stacking free-form residual blocks in the manner prescribed by a variational integrator to obtain a deep network.

We focus on VIs with explicit discrete update equations arising from the discrete equations of motion (2) and (3). This results in network architectures that do not require fixed-point algorithms to evolve the dynamics.

We begin by considering separable *Newtonian networks*, i.e. networks that follow Newton's laws of physics. These are constructed by considering a parameterized Lagrangian of the form

$$L_\theta(\boldsymbol{q}, \dot{\boldsymbol{q}}) = T_\theta(\dot{\boldsymbol{q}}) - U_\theta(\boldsymbol{q}) = \frac{1}{2}\dot{\boldsymbol{q}}^T \mathbf{M}_\theta \dot{\boldsymbol{q}} - U_\theta(\boldsymbol{q}), \quad (4)$$

where T_θ and U_θ are the kinetic and potential energy of the system, and $\mathbf{M}_\theta$ is a symmetric, positive definite inertia matrix. We omit time dependence for ease of notation. From a Lagrangian perspective, approximating the action by the quadrature rule

$$L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h) = hL_\theta\Big(\boldsymbol{q}_t, \frac{(\boldsymbol{q}_{t+1} - \boldsymbol{q}_t)}{h}\Big), \qquad (5)$$

we arrive at the Störmer-Verlet (SV) integrator

$$\boldsymbol{q}_{t+1} = 2\boldsymbol{q}_t - \boldsymbol{q}_{t-1} - h^2 \mathbf{M}_\theta^{-1} \frac{\partial U_\theta(\boldsymbol{q}_t)}{\partial \boldsymbol{q}_t}. \qquad (6)$$

The symmetric variant of (5), given by

$$L^d(\boldsymbol{q}_t, \boldsymbol{q}_{t+1}, h) = \frac{h}{2}\Big(L_\theta\Big(\boldsymbol{q}_t, \frac{(\boldsymbol{q}_{t+1} - \boldsymbol{q}_t)}{h}\Big) \qquad (7)$$

$$+ L_\theta\Big(\boldsymbol{q}_{t+1}, \frac{(\boldsymbol{q}_{t+1} - \boldsymbol{q}_t)}{h}\Big)\Big), \qquad (8)$$

yields the velocity Verlet (VV) integrator

$$\boldsymbol{q}_{t+1} = \boldsymbol{q}_t + h\mathbf{M}_\theta^{-1}\dot{\boldsymbol{q}}_t - \frac{h^2}{2}\mathbf{M}_\theta^{-1}\frac{\partial U_\theta(\boldsymbol{q}_t)}{\partial \boldsymbol{q}_t}, \qquad (9)$$

$$\boldsymbol{p}_{t+1} = \boldsymbol{p}_t - \frac{h}{2}\Big(\frac{\partial U_\theta(\boldsymbol{q}_t)}{\partial \boldsymbol{q}_t} + \frac{\partial U_\theta(\boldsymbol{q}_{t+1})}{\partial \boldsymbol{q}_{t+1}}\Big), \qquad (10)$$

where $\boldsymbol{p}_t = \mathbf{M}_\theta^{-1}\dot{\boldsymbol{q}}_t$. Compared to (6), the VV integrator explicitly incorporates the momentum/velocity.

Combining variational integrators with the neural ODE viewpoint, we arrive at network architectures that enjoy the following properties.

1. Physical properties, such as conservation laws, are automatically enforced by preserving the underlying geometric structure.

2. Flexibility to model complex phenomena is retained, as U_θ can be a black-box neural network. We opt for a single-layer fully connected network.

3. Interpretability is increased, by considering that the embedding evolves in a phase-space, having notions of kinetic and potential energy.

4. Modeling specificity is increased, since the mass term can either be modeled explicitly or taken to be the identity matrix.

To illustrate how variational integrators enable us to build further geometric structure into the model, consider a *Newtonian rotation network* in 2D. The idea is to exploit the knowledge that a system's evolution takes place entirely on a manifold, here the space of rotations, by incorporating this structure into the network.

For this, we consider a particular class of variational integrators: *Lie group variational integrators* (LGVIs). LGVIs exploit the properties of Lie groups to construct integrators that automatically evolve on a specified Lie group. The key idea is to approximate the change in position over integration steps using group elements (Leok, 2007). Since the state space is closed under the group action (e.g. matrix multiplication when represented by matrices), the constraints are automatically enforced. For instance, the Lie group $SO(2)$ (with matrix multiplication as the group action) is a natural way to encode the underlying manifold of 2D rotations, like the evolution of the angle of a pendulum. A Newtonian network in a uniform gravitational potential that evolves automatically on $SO(2)$ is specified as follows. Denoting the angle by ϑ, the corresponding rotation network is given by

$$\sin \Delta\vartheta_t = \sin \Delta\vartheta_{t-1} + h^2 r_\varphi(\vartheta), \qquad (11)$$

$$\vartheta_{t+1} = \vartheta_t + \Delta\vartheta_t, \qquad (12)$$

where $r_\varphi(\vartheta_t)$ is a neural network with $\sin(\cdot)$ activations at the last layer. Appendix B provides further details.
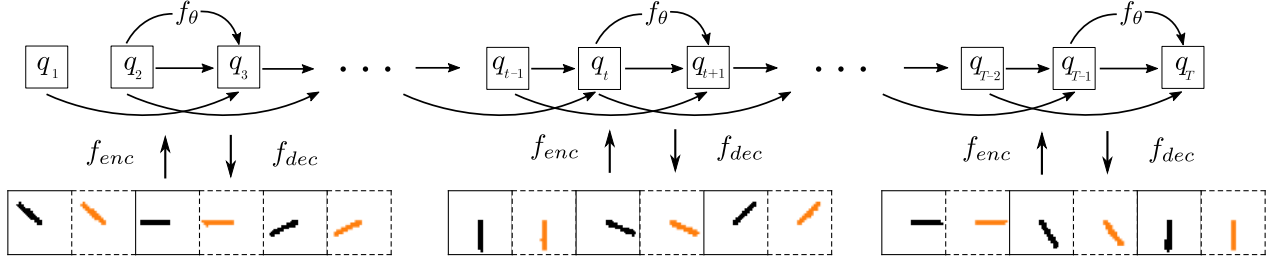
Figure 2: Learning the dynamics of a pendulum from pixel observations. Here, a variational autoencoder maps the pixels into the latent space q using f^{enc}, and maps the latent space back into pixels using f^{dec}. A Lagrangian variational integrator is used, for which q is the hidden state. Unlike an ordinary residual network, the skip connections used are intertwined. We display the observations in black, and predicted values given by the decoder in orange. Experimental details for this setup are given in Section 4.

3.1 Learning VINs from Noisy Observations

Given initial conditions of a system, the state evolution is given by a solution $\boldsymbol{q}(t)$ to the equations of motion. Denoting the state in phase space by $\boldsymbol{x}_t = (\boldsymbol{q}_{t-1}, \boldsymbol{q}_t)$ from the Lagrangian perspective or $\boldsymbol{x}_t = (\boldsymbol{q}_t, \boldsymbol{p}_t)$ from the Hamiltonian perspective, VINs represent an approximation to the solution between the initial condition $\boldsymbol{x}_1$ and terminal state $\boldsymbol{x}_T$. We represent a layer in the network by

$$\boldsymbol{x}_t = f_\theta(\boldsymbol{x}_1, h, t), \tag{13}$$

as a function of the initial condition, step size and time step (layer index) t. Figure 1 gives an illustration of a VIN. Given a path of noisy observations $\boldsymbol{y}_{1:T}$ of the state of a system, we specify a Gaussian likelihood

$$p(\boldsymbol{y}_{1:T} \mid \boldsymbol{x}_{1:T}, \sigma^2) = \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{y}_t \mid \boldsymbol{x}_t, \sigma^2 \mathbf{I}). \tag{14}$$

Define $\Theta = (\theta, \boldsymbol{x}_1, \sigma^2)$ where θ are the parameters of the VIN, $\boldsymbol{x}_1$ is the initial condition , and σ^2 is the error variance. We train the model by maximizing the log of the likelihood (14) with respect to Θ using stochastic optimization.

3.2 VINs for High Dimensional Observations

It is possible that the dynamical system of interest is not observed directly, but indirectly through a set of intermediate data not of primary interest. For example, we can observe a swinging pendulum by seeing images of its location at a given set of time instances. We propose to address this problem using variational autoencoders (VAEs) (Kingma and Welling, 2014; Rezende et al., 2014). VAEs enable approximate inference in latent variable models that model high-dimensional observations as being generated by some lower dimensional latent space. We aim to combine this setup with

VINs to learn physical systems that evolve in a latent phase-space.

We start by placing a standard Gaussian $p_\theta(\boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{x}_1 \mid \boldsymbol{0}, \mathbf{I})$ over the initial condition. The joint distribution over a path is

$$p_\theta(\boldsymbol{x}_{1:T}) = p_\theta(\boldsymbol{x}_1)p_\theta(\boldsymbol{x}_{2:T} \mid \boldsymbol{x}_1), \tag{15}$$

which we can sample from by sampling $\boldsymbol{x}_i^s \sim p(\boldsymbol{x}_1)$, and propagating the samples through the network $\boldsymbol{x}_t^s = x_\theta(\boldsymbol{x}_1^s, h, t)$. Assuming noise-free dynamics the uncertainty over the dynamics is fully induced by the distribution of the initial condition.

We specify the joint distribution over observations and paths in latent space as

$$p_\theta(\boldsymbol{y}_{1:T}, \boldsymbol{x}_{1:T}) = p_\theta(\boldsymbol{x}_{1:T}) \prod_{t=1}^{T} p_\theta(\boldsymbol{y}_t \mid \boldsymbol{x}_t). \tag{16}$$

The likelihood $p_\theta(\boldsymbol{y}_t \mid \boldsymbol{q}_t)$ is parameterized by a decoder neural network $f_\theta^{dec}(\boldsymbol{q}_t)$, which depends only on the position component $\boldsymbol{q}_t$ of $\boldsymbol{x}_t$.

We aim to approximate the posterior distribution $p_\theta(\boldsymbol{x}_{1:T} \mid \boldsymbol{y}_{1:T})$, which is intractable due to the non-linear relationships introduced by the decoder f_θ^{dec} and the dynamics x_θ in (13). In the VAE setup, we specify an approximation $q_\phi(\boldsymbol{x}_{1:T} \mid \boldsymbol{y}_{1:T})$ to the posterior, parameterized by an encoder network $f_\phi^{enc}(\boldsymbol{y}_{1:T})$, where ϕ are called the variational parameters. Figure 2 illustrates the VIN-VAE setup.

We learn the parameters by variational inference. We choose the variational family

$$q_\phi(\boldsymbol{x}_{1:T} \mid \boldsymbol{y}_{1:T}) = q_\phi(\boldsymbol{x}_1)p_\theta(\boldsymbol{x}_{2:T} \mid \boldsymbol{x}_1), \tag{17}$$

$$q_\phi(\boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{x}_1 \mid \boldsymbol{m}_1, \boldsymbol{s}_1^2). \tag{18}$$

Note that the conditional $p_\theta(\boldsymbol{x}_{2:T} \mid \boldsymbol{x}_1)$ is the same in the variational family as in the model. The mean

and variance of the initial condition are in general estimated from the full trajectory $\boldsymbol{y}_{1:T}$ by the encoder f_ϕ^{enc}. To train the model, we minimize Kullback-Leibler divergence with respect to the model parameters θ and the variational parameters ϕ, which is equivalent to maximizing the evidence lower bound

$$\sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{x}_t^s \sim q_\phi(\boldsymbol{x}_t|\cdot)} \log p(\boldsymbol{y}_t \mid \boldsymbol{x}_t) - \mathbb{KL}\big[q_\phi(\boldsymbol{x}_1)||p(\boldsymbol{x}_1)\big],$$

where $\boldsymbol{x}_t^s \sim q_\phi(\boldsymbol{x}_t(\cdot))$ denotes a sample from $q_\phi(\boldsymbol{x}_1)$ propagated through the network x_θ, and $\mathbb{KL}\big[q \,||\, p\big]$ denotes the Kullback-Leibler divergence between q and p. This objective is maximized with respect to θ and ϕ jointly using stochastic optimization.

4 Experiments

To study the performance of VINs, we implemented them for two reference systems: (a) an ideal pendulum, (b) an ideal mass-spring system. We study the ability of VINs to infer a useful representation of the system when given a small quantity of data, in cases where the dynamical system is observed both directly and indirectly. Full details for network architectures and hyperparameters are given in Appendix C.

4.1 Learning from Noisy Observations

We consider VINs in a noisy setting. Specifically, the model is given noisy position and velocity measurements from which it needs to learn the dynamics. We compare our proposed VINs with Hamiltonian neural networks (HNNs) (Greydanus et al., 2019) and standard feed-forward neural networks (NNs) without additional structure that would explicitly incorporate physical or mechanical constraints. We use the VIN given by (6). HNNs are trained on observations of the form $(\boldsymbol{q}_t, \boldsymbol{p}_t, \dot{\boldsymbol{q}}_t, \dot{\boldsymbol{p}}_t)$. We replicate the setup from Greydanus et al. (2019) with one key difference: we introduce noise in all observations, rather than only introducing it in $(\boldsymbol{q}_t, \boldsymbol{p}_t)$ and observing $(\dot{\boldsymbol{q}}_t, \dot{\boldsymbol{p}}_t)$ noise free. This makes the setting more realistic, but system identification harder. To account for the noise, we add a noise variable to all models and maximize the log-likelihood, rather than only mean-squared error.

We examine two scenarios: (a) a moderate-data regime, where models are trained using 25 training trajectories with a total of 750 data points, (b) a low-data regime using 5 training trajectories with a total of 150 data points. Figures 3 and 4 show that prediction performance differs between the models. In the low-data regime, despite learning to approximately conserve the system's energy, the HNN does not capture the correct dynamics, and performs poorly on prediction in terms

of RMSE on both systems. On the mass-spring system (Figure 3), with sufficient data, the HNN prediction error is low over a small horizon, but exhibits two large jumps as the trajectory evolves. We suggest that in both cases the HNN fits the noise in the training data (overfits) and fails to identify the underlying system. The NN baseline performs better than the HNN in the low-data regime, whereas the HNN demonstrates better predictive performance in the moderate-data regime on the pendulum system (Figure 4). The VIN exhibits good predictive performance, outperforming the baselines on both systems, in both the low-data and moderate-data regimes.

Figures 3 and 4 show that the energy behaviors of HNNs, VINs, and NNs differ. Given sufficient data, both the HNN and VIN learn a model that conserves a quantity that approximates the energy of the system. However, the HNN overfits in the low-data regime on both systems. The NN baseline incorrectly dissipates/adds energy in both scenarios for the pendulum system, particularly as time passes, but learns to approximately conserve energy for the mass-spring system given 25 training trajectories. This contributes to the worse predictive performance of the NN baseline compared to the HNN and VIN.

Overall, VINs can effectively identify the system from noisy observations, even in small-data scenarios, where HNNs and NNs can overfit. We attribute this to their architecture: their embedded space is a dynamical system in its own right, which enforces physical constraints automatically when forecasting so that their long-term predictions better match the true system. In contrast, the HNN relies on generalization to conserve energy, as demonstrated by the difference in performance in the low-data and moderate-data regimes.

4.2 Learning from Pixel Observations

We study VINs in a variational auto-encoder (VAE) setting, which adds an auxiliary image processing task to prediction. Here, we observe 28×28 pixel images depicting the mass-spring and pendulum systems; see, e.g. Figures 7–8. For the mass-spring, we use (6) for the dynamics (VIN-SV). For the pendulum, we run experiments using both (9) (VIN-VV) as well as the dynamics imposing $SO(2)$ manifold constraint in (11) (VIN-$SO(2)$). As a baseline, we use a parameter-tied deep recurrent residual network (ResRNN) having the same number of layers as the VINs, with each layer sharing the same single-layer neural network. This mirrors the structure that arises from the time independence of the Lagrangian in VINs. Each model is trained within a VAE framework as described in Section 3.2.

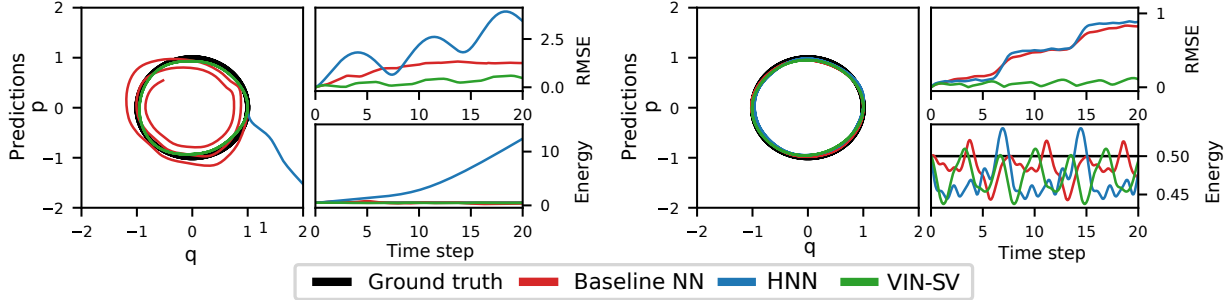We evaluate the structure of the latent space learned

Figure 3: Learning physics from noisy observations for the ideal mass-spring. Given a set of initial conditions, we forecast a path in configuration space and compare against the ground truth. We show model predictions, total root-mean-squared error between coordinates and the total energy of the dynamical system in the embedding.
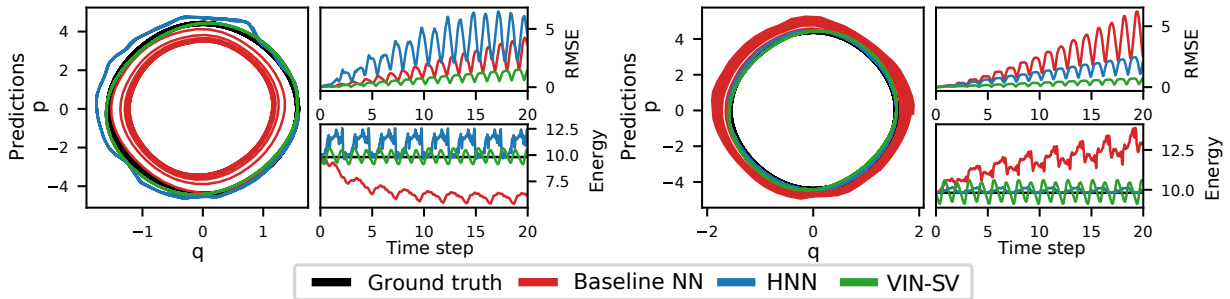

Figure 4: Learning physics from noisy observations for the ideal pendulum. Given a set of initial conditions, we forecast a path in configuration space and compare against the ground truth. We show model predictions, total root-mean-squared error between coordinates and the total energy of the dynamical system in the embedding.

by VIN-$SO(2)$ and compare it with representations learned by a standard VAE (Kingma and Welling, 2014; Rezende et al., 2014), a VAE with free-form dynamics governed by a feed-forward network (DVAE), and a Lie group VAE (LG-VAE) (Falorsi et al., 2018) with no dynamic structure. Figure 5 visualizes the latent spaces after training on $4s$ (40 observations) and mapping an additional 80 test images into latent space using the encoder f_ϕ^{enc}, including the dynamics in the case of the VIN-$SO(2)$. The VAE captures local structure: observations close together in image space are mapped to points close together in latent space. However, it fails to capture the global structure of the state space and has discontinuities with respect to the sequential nature of the dataset. Figure 5(b) shows that adding an unrestricted neural network to capture the dynamics does not solve the problem. The LG-VAE captures the correct global structure by restricting the manifold, but still exhibits discontinuities with respect to the time dimension, since it does not model the dynamics. The embedding for VIN-$SO(2)$ does not have such discontinuities: it learns both the global structure and respects the sequential nature of the data due to the structure encoded by the VIN.

For both systems, we generate 6 seconds of training

observations, sampled at a frequency of 10Hz (60 observations). Training data is split into overlapping $1s$ image trajectories (10 observations), matching the depth of the networks, which was 10 in all experiments.

We assess the models qualitatively by looking at the properties of their latent spaces. In particular, we infer a distribution over the initial condition using the learned encoder f_ϕ^{enc} given 10 initial observations from the pendulum system. We then evolve the learned system for 20 seconds using the mean of the variational posterior.

Figure 6 shows how the ResRNN does not learn dynamics that match the geometric properties of the true system (i.e. symplectic) but instead spirals away from the initial condition (denoted by the large circle). This is because the Euler discretization scheme used by residual networks ignores the underlying geometry. On the other hand, both the VIN-VV and the VIN-$SO(2)$ models automatically preserve symplectic structure and evolve strictly on a sub-manifold in their respective latent phase-spaces. Importantly, while the flexibility afforded by the decoder allows the ResRNN setup to generate plausible observations up to some fixed horizon, the unbounded behavior of the evolution eventually causes significant failures.

(a) VAE   (b) DVAE   (c) LG-VAE   (d) VIN-$SO(2)$ (e) VIN-$SO(2)$ (fixed)   (f) Ground Truth
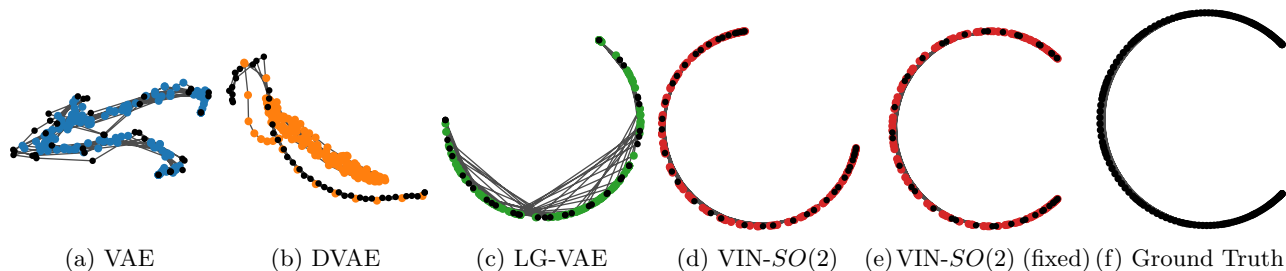
Figure 5: Example embedded representations of an ideal pendulum system. Black/colored dots represent embedded train/test images, gray lines connect points sequentially in time. The embeddings learned by the baseline models fail to capture the global structure (a)–(b) and/or are discontinuous with respect to the time dimension (c). The VIN-$SO(2)$ (d)–(e) learns an embedding that is consistent with the ground truth (f). In (e), we fix the mass matrix, which is not identifiable from pixel data, to the true value. Here, the VIN-$SO(2)$ faithfully reconstructs the ground truth.
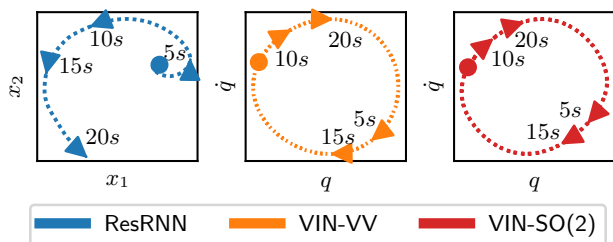

Figure 6: Latent embeddings learned from pixel observations of an ideal pendulum. Circles denote the inferred initial condition, dots denote predictions forward in time. Triangles mark 5-second intervals in the forecasts. The ResRNN fails to capture the underlying geometric structure and spirals far beyond the initial condition. VINs preserve this structure automatically.

System	Model	RMSE	$\log p(\boldsymbol{y} \mid \boldsymbol{x}) \times 10^2$
	ResRNN	6.1 ± 0.2	-246.7 ± 79.2
Pendulum	VIN-VV	4.3 ± 0.6	-13.4 ± 5.8
	VIN-$SO(2)$	$\mathbf{3.4 \pm 0.6}$	$\mathbf{-3.2 \pm 1.9}$
Mass-	ResRNN	6.1 ± 0.1	-4.7 ± 2.4
spring	VIN-SV	$\mathbf{3.2 \pm 0.2}$	$\mathbf{-0.2 \pm 0.0}$

Table 1: RMSE and log-likelihood (with standard errors) for the pendulum and mass-spring systems over $5s$ forecasts on pixel observations.

Figure 7 shows the reconstructions obtained by mapping the latent paths from Figure 6 through the decoder f_θ^{dec}. Between $10s$–$15s$ of forecasting, the ResRNN predictions are unreliable: going through discontinuous jumps in pixel space, suddenly reversing the dynamics and generating half-formed pendula (see, e.g. the final step in Figure 6).

Conversely, the VINs do not exhibit such non-physical behavior, since the latent path remains bounded on the data manifold despite forecasting for effectively arbitrary long horizons. The VIN-VV does display signs of going out of phase with the ground truth around $15s$ in Figure 7, becoming more pronounced around the $20s$ mark. One explanation is that we only consider the path traversed by the mean of the variational posterior, and ignore the build-up in uncertainty as the prediction horizon increases. However, looking at the same reconstructions from the VIN-$SO(2)$ model, we see that it does not suffer from this problem within the

$20s$ prediction horizon. Therefore, we assume that the error from assuming an Euclidean manifold contributes to the mismatch as well.

We perform the same qualitative analysis on reconstructions of the mass-spring system, shown in Figure 8. Although the underlying system is simpler in this instance, the performance of the ResRNN deteriorates even quicker with increasing prediction horizon. The VIN-SV also exhibits small errors in the reconstruction at the $10s$ mark, but captures the underlying dynamics well, as can be seen by its long-term predictions.

We perform a quantitative analysis with a similar setup on both systems. Specifically, we run 10 randomized trials, where we generated 6 seconds of observations to train on and use the same architectures as before. In each trial, we then infer a distribution for the initial condition on the same trajectory and evaluated the RMSE and log-likelihood for $5s$ forecasts. We evaluate on the training trajectory to isolate properties of the dynamics, which is only trained on $1s$ forecasts (i.e. having 10 layers). Table 1 shows the results with standard errors. Both VINs perform significantly better than the ResRNN in terms of both RMSE and log-likelihood. The VIN-$SO(2)$ shows a meaningful improvement in terms of log-likelihood when compared
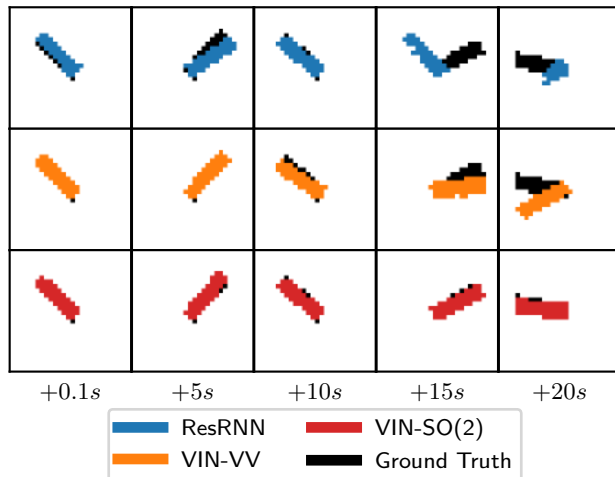
Figure 7: Reconstructions of the pendulum system from forecasts in latent space, using a step-size of $0.1s$, up to 20 seconds. The ground truth (black) is occluded by the model predictions, shown in color.

Figure 8: Reconstructions of the mass-spring system from forecasts in latent space, using a step-size of $0.1s$, up to 20 seconds. The ground truth (black) is occluded by the model predictions, shown in color. The mass oscillates left-and-right based on the initial tension in the spring (not rendered in images).

to the VIN-VV, whereas the RMSE is inconclusive.

5 Discussion

VINs can be used to create embeddings that faithfully represent dynamical systems. This enables them to learn with less data and provides greater interpretability compared to other network architectures, while facilitating accurate long-term predictions. Provided their state space is chosen appropriately, VINs preserve the topological and geometric structure of the dynamical systems they encode. This assists with performance, mirrors recent developments in VAEs designed to accurately encode physical systems (Gong and Cheng, 2019; Haber and Ruthotto, 2017; Lutter et al., 2019; Caterini et al., 2018), and is well-motivated by recent theoretical observations made in the context of neural ODEs (Dupont et al., 2019).

The imposition of additional geometric structure does not cause VINs to lose their capacity to model flexible classes of phenomena. In particular, they are still parameterized by an underlying neural network. This mirrors the design of residual networks and other architectures related to differential equations (Haber and Ruthotto, 2017; Chen et al., 2018). Thus, VINs are more interpretable than purely black-box approaches to network design, while still being highly expressive. VINs can be trained directly on noisy observations. They may also be used as part of larger and more complex learning pipelines, e.g. by incorporating them into an auto-encoding framework. Performance in both settings is discussed in Section 4.
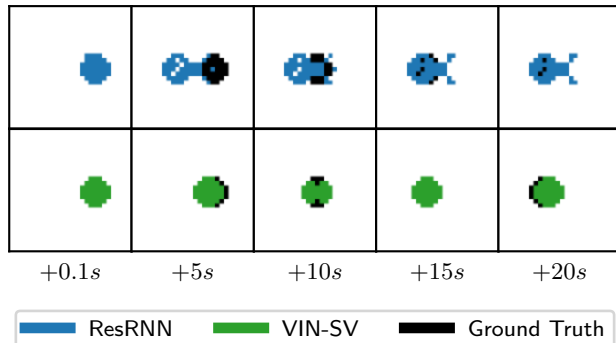
A number of directions could be pursued to improve these ideas. In particular, one could study these ideas with time-varying Lagrangians, improving expressivity by greatly expanding the class of dynamical systems faithfully representable by the embedding. This would bring VINs closer in line with residual networks and general neural ODEs (Chen et al., 2018). While we focused on data efficiency and representation learning in settings where the underlying dynamics are fairly simple, it would be interesting to study such networks on more complex tasks. This could pave the way toward better performance on currently difficult problems in areas where the phenomena under study are dynamical systems, such as robotics and reinforcement learning.

6 Conclusion

In this work, we introduced *variational integrator networks*, a class of deep network architectures for creating neural embeddings, which encode and represent dynamical systems. VINs ensure faithful representation of dynamical systems by using an embedding that forms a dynamical system in its own right. This facilitates data-efficient learning, enhances interpretability, and allows for accurate long-term predictions when compared to other classes of networks.

Recent trends in deep learning have sought to improve the performance of deep networks on physical systems by designing networks whose behavior is more understandable and better matched to the underlying physics. Variational integrator networks take a step toward progressing this line of work.

Acknowledgments

References

A. L. Caterini, A. Doucet, and D. Sejdinovic. Hamiltonian variational auto-encoder. In *Advances in Neural Information Processing Systems*, 2018. Cited on page 8.

B. P. Chamberlain, J. Clough, and M. P. Deisenroth. Neural embeddings of graphs in hyperbolic space. *arXiv:1705.10359*, 2017. Cited on page 2.

B. Chang, L. Meng, E. Haber, L. Ruthotto, D. Begert, and E. Holtham. Reversible architectures for arbitrarily deep residual neural networks. In *AAAI Conference on Artificial Intelligence*, 2018. Cited on page 2.

R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018. Cited on pages 1, 2, 8.

T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. Hyperspherical variational auto-encoders. In *Uncertainty in Artificial Intelligence*, 2018. Cited on page 2.

P. d. Haan and L. Falorsi. Topological constraints on homeomorphic auto-encoding. In *NeurIPS Workshop on Integration of Deep Learning Theories*, 2018. Cited on page 2.

E. Dupont, A. Doucet, and Y. W. Teh. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, 2019. Cited on page 8.

W. E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017. Cited on pages 1, 2.

L. Falorsi, P. d. Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. In *ICML Workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018. Cited on pages 2, 6.

L. Falorsi, P. d. Haan, T. R. Davidson, and P. Forré. Reparameterizing distributions on Lie groups. In *International Conference on Artificial Intelligence and Statistics*, 2019. Cited on page 2.

O. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic neural networks. *Advances in Neural Information Processing Systems*, 2018. Cited on page 2.

L. Gong and Q. Cheng. Lie group auto-encoder. *arXiv:1901.09970*, 2019. Cited on page 8.

S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, 2019. Cited on pages 1, 5, 11.

E. Haber and L. Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017. Cited on pages 1, 2, 8.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014. Cited on pages 4, 6.

M. Leok. An overview of Lie group variational integrators and their applications to optimal control. In *International Conference on Scientific Computation and Differential Equations*, 2007. Cited on page 3.

A. J. Lew, J. E. Marsden, M. Ortiz, and M. West. An overview of variational integrators. In *Finite Element Methods: 1970s and beyond*. CIMNE, 2004. Cited on page 2.

M. Lutter, C. Ritter, and J. Peters. Deep Lagrangian networks: using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2019. Cited on pages 1, 8.

J. E. Marsden, S. Pekarsky, S. Shkoller, and M. West. Variational methods, multisymplectic geometry and continuum mechanics. *Journal of Geometry and Physics*, 38(3–4):253–284, 2001. Cited on page 2.

R. A. Meyers. *Mathematics of complexity and dynamical systems*. Springer, 2009. Cited on page 11.

M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, 2017. Cited on page 2.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378(686–707), 2019. Cited on page 1.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014. Cited on pages 4, 6.

L. Ruthotto and E. Haber. Deep neural networks motivated by partial differential equations. *arXiv:1804.04272*, 2018. Cited on page 2.

M. West. *Variational integrators*. PhD thesis, California Institute of Technology, 2004. Cited on page 2.

A   Appendix: Short Review of Lagrangian and Hamiltonian Mechanics

Hamiltonian and Lagrangian mechanics are two intricately related formulations of classical mechanics. In classical mechanics, we assume that we are given a continuous-time dynamical system defined on a space $\mathcal{Q} \subseteq \mathbb{R}^d$, which we call the *configuration space*. A state of the system is taken to be a set of parameters $\boldsymbol{q} \in \mathcal{Q}$ that uniquely identify the configuration of the system. Continuous-time evolution of the dynamics in $\mathcal{Q}$ yields a path in configuration space. Lagrangian and Hamiltonian mechanics formulate the laws of physics in terms of properties of these paths.

Specifically, *Hamilton's principle*, also called the *Principle of Least Action*, states that there exists a real-valued function L such that all paths in configuration space which occur in nature minimize the path integral

$$S(\boldsymbol{q}) = \int_0^T L(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) \, \mathrm{d}t \tag{19}$$

where $\dot{\boldsymbol{q}}$ is the velocity, which is the time-derivative of position. For a given L, it can be shown using the calculus of variations that minimization of A is equivalent to solving a system of partial differential equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}^q}\right) - \frac{\partial L}{\partial \boldsymbol{q}^q} = \boldsymbol{0}, \tag{20}$$

called the *Euler-Lagrange Equations*, or the *equations of motion*. Given a set of initial conditions $(\boldsymbol{q}(0), \dot{\boldsymbol{q}}(0))$, the solutions to the equations of motion describe the trajectory of the system.

This gives the starting point of Lagrangian mechanics – physical phenomena that satisfy it are called *classical*, and span virtually all areas of physics. The behavior of particular phenomena varies according to choice of the Lagrangian L, which fully characterizes how the system evolves over time.

For example, for $\boldsymbol{q} \in \mathbb{R}^d$, take $L(\boldsymbol{q}, \dot{\boldsymbol{q}}) = T(\boldsymbol{q}, \dot{\boldsymbol{q}}) - U(\boldsymbol{q})$ where T is the kinetic energy, and U is the potential energy of the system. This describes a conservative Newtonian system.

B   Appendix: Lie Group Variational Integrator for $SO(2)$

We start by formulating a Lagrangian with the Lie group $SO(2)$ using matrix representations. First, define the map from scalars $\omega \in \mathbb{R}$ to 2×2 skew-symmetric matrices

$$\mathbf{S}(\omega) = \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}. \tag{21}$$

The set of 2×2 skew-symmetric matrices forms the Lie algebra $\mathfrak{so}(2)$. The matrix exponential map, takes elements of the Lie algebra to elements of the group $SO(2)$

$$\mathbf{R}(\omega) = \exp \mathbf{S}(\omega) = \begin{bmatrix} \cos \omega & -\sin \omega \\ \sin \omega & \cos \omega \end{bmatrix}. \tag{22}$$

Kinematics for group elements $R \in SO(2)$ can be written in terms of Lie algebra elements as

$$\dot{\mathbf{R}} = \mathbf{R}\mathbf{S}(\omega), \tag{23}$$

where ω is analogous to angular velocity. A conservative Newtonian Lagrangian in a uniform gravitational potential can be written in terms of the Lie group $SO(2)$ as

$$L(\mathbf{R}, \mathbf{S}(\omega)) = \frac{1}{2}ml^2\omega^2 + mgl\boldsymbol{e}_2^T\mathbf{R}\boldsymbol{e}_1 \tag{24}$$

where $\mathbf{R} = \mathbf{R}(\theta)$ is a rotation matrix parameterized by θ, g is the gravitational acceleration and $\boldsymbol{e}_1, \boldsymbol{e}_2$ are orthogonal unit vectors in the inertial frame of reference, $\boldsymbol{e}_1 = [1, 0], \boldsymbol{e}_2 = [0, 1]$.

To develop a Lie group variational integrator, define $\mathbf{F}_t \in SO(2)$ such that

$$\mathbf{R}_{t+1} = \mathbf{R}_t\mathbf{F}_t. \tag{25}$$

Since $\mathbf{F}_t \in SO(2)$, the update enforces $\mathbf{R}_{t+1} \in SO(2)$ since Lie groups are closed under the group action. Here, group action is given by matrix multiplication. Then define the discretization of the action integral as

$$L^d(\mathbf{R}_k, \mathbf{F}_k) = \frac{1}{2h}ml^2\langle\mathbf{F}_k - \mathbf{I}, \mathbf{F}_k - \mathbf{I}\rangle + \frac{hmgl}{2}\left(\boldsymbol{e}_2^T\mathbf{R}_t\boldsymbol{e}_1 + \boldsymbol{e}_2^T\mathbf{R}_{t+1}\boldsymbol{e}_1\right), \tag{26}$$

which approximates the angular velocity as

$$\mathbf{S}(\dot{\theta}) = \frac{\mathbf{F}_k - \mathbf{I}}{h}. \tag{27}$$

Using the discrete form of Hamilton's principle, one obtains (Meyers, 2009) the equation

$$(\mathbf{F}_t - \mathbf{F}_t^T) - (\mathbf{F}_{t+1} - \mathbf{F}_{t+1}^T) - \frac{2h^2g}{l}\mathbf{S}(\boldsymbol{e}_2^T\mathbf{R}_{t+1}\boldsymbol{e}_1) = \mathbf{0}, \tag{28}$$

which, when taken with (25), defines the Lie group variational integrator. One arrives at (11), written in terms of the elements of the matrices, by subsuming the force terms into the neural network.

C   Appendix: Hyperparameters for Experiments

C.1   Noisy System Observations

The setup resembles the one of Greydanus et al. (2019) closely. The neural network architecture for the baseline NN, the network that parameterizes the Hamiltonian in HNNs and the one that parameterizes the VIN was the same throughout. This was a single hidden layer feed-forward network with 200 hidden units and $\tanh(\cdot)$ activations on the hidden layer. The noise added to the observations was sampled from a standard Gaussian with standard deviation $\sigma = 0.1$. For the mass-spring system, we set the spring constant and mass to $k = m = 1$, as was done by Greydanus et al. (2019). For the pendulum, unlike the original work, we use $m = l = 1$, and $g = 9.81$. Training trajectories were sampled uniformly from energies ranging from $[0.2, 1]$ for the mass-spring system and $[1.3, 2.3]$ for the pendulum. We trained the models using ADAM with a learning rate of 10^{-3}. We did a hyperparameter search over $[2000, 5000, 10000]$ training steps and chose the best performing models for comparison.

For predictions with the baseline NN and HNN, we use the procedure of Greydanus et al. (2019), which uses fourth order Runga-Kutta with an error tolerance of 10^{-9}, implemented in SCIPY.INTEGRATE.SOLVE_IVP. For the VIN we simply predict forwards in time using the trained network.

C.2   Pixel Observations

In all VAE experiments we used the same encoder and decoder structure. Both the encoder and decoder consisted of two fully connected hidden layers with a 1000 hidden units and ReLU activation functions.

- Encoder: two fully-connected hidden layers with 1000 units and ReLU activation functions, followed by an LSTM with a 50 dimensional hidden state that processed the embedded sequence in reverse to give the variational parameters for the initial condition.

- Decoder: two fully-connected hidden layers with 1000 units and ReLU activation functions.

The dynamics networks (i.e. ResRNN, VIN-VV, VIN-$SO(2)$, VIN-SV) all had a depth of 10 and used 10 observations as input to the encoder. The step size for the networks was chosen to be 1.0 in latent space. The underlying fully connected network had 1000 hidden units and tanh activation functions.

We train using ADAM with a learning rate of 3.0×10^{-4} until the ELBO converges on the training set, up to a maximum of 6000 epochs through the datasets and use the parameters with the highest ELBO for evaluation.