VIGN: Variational Integrator Graph Networks

Shaan A. Desai*, **Stephen J. Roberts**
Machine Learning Research Group
University of Oxford

Abstract

Rich, physically-informed inductive biases play an imperative role in accurately modelling the time dynamics of physical systems. In this paper, we introduce *Variational Integrator Graph Networks* (VIGNs), the first approach to combine a Variational Integrator (VI) inductive bias with a Graph Network (GN) and demonstrate an order of magnitude improvement in performance, both in terms of data-efficient learning and predictive accuracy, over existing methods. We show that this improvement arises because VIs induce coupled learning of generalized position and momentum updates which can be formulated as a Partitioned Runge-Kutta (PRK) method. We empirically establish that VIGN outperforms numerous methods in learning from existing datasets with noise.

1 Introduction

Accurately and efficiently learning the time evolution of physical systems has long been a challenge in the deep learning community. Numerous approaches exist to model time dynamics but the most straightforward methods involve learning functions which map inputs directly to their time derivatives [6] or to the next state [1, 3]. However, it has been shown that these approaches result in poor test time generalization and predictive accuracy because they do not precisely capture underlying physical laws, such as energy and momentum conservation. The natural question therefore arises as to how can we endow models to better capture underlying physical laws. Extensive research has shown that enriching models with well-chosen inductive biases such as Hamiltonian Neural Networks (HNNs) [6], ordinary differential equations (ODEs) [4, 5] and more recently, graphs [1, 14, 13] can significantly improve learning. Fundamentally, these inductive biases allow us to bridge our theoretical expectations with observed data and serve as a guide in learning the underlying physical process. Although most methods to date have focused on studying physically-informed inductive biases individually, more recent work has demonstrated the remarkable performance gains achievable through integrating inductive biases [12, 11].

Inspired by these advances, we developed *Variational Integrator Graph Networks* (VIGNs) - the first method to combine the inductive bias of variational integrator networks (VINs) [11] with graph networks (GNs) [2]. We show that this combination fundamentally allows us to couple multiple inductive biases, namely, ODEs, VINs and GNs which serves to unify existing approaches. Importantly, we empirically demonstrate that such coupling results in an order of magnitude improvement in learning over the same number of iterations in comparison to Hamiltonian Graph Networks (HOGNs) and ODE Graph Networks (OGNs) [12] on the N-body spring problem. Furthermore, we empirically show that VIGNs are robust at learning in noisy data settings in which most methods over-fit during training. We highlight the success of our method in predicting trajectories in Fig.2. Table 1 highlights the core differences between existing methods and provides an overview of how VIGNs unify the key attributes of other approaches.

*email: shaan@robots.ox.ac.uk

Preprint. Under review.

Table 1: Comparison of VIGN against Existing Methods

	Baseline NN	OGN	HNN	VIN	HOGN	VIGN
Can model dynamical systems	✓	✓	✓	✓	✓	✓
Learns differential equations			✓	✓	✓	✓
Conserves physical laws			✓	✓	✓	✓
Interpretable				✓	✓	✓
Integrator Coupled		✓		✓	✓	✓
Graph Coupled		✓			✓	✓
Position and Momentum Coupled				✓		✓
Robust to Noisy Data				✓		✓
Extendable to large N-body problems		✓			✓	✓

2 Background

Numerous recent approaches tackle learning from physical data, but of them three methods stand out; Graph Networks, Hamiltonian Neural Networks and Variational Integrator Networks. VIGNs allow us to combine the major strengths of each approach and hence form a simple, unifying framework for learning the time dynamics of physical systems. We briefly review the methods in the following sections.

2.1 Graph Neural Networks

The state of a physical system can be represented by a graph $G = (u, V, E)$ [1]. For example, a node (V) can be used to represent a particle in an N-body problem. These nodes can be used to represent the core features of the particle e.g. position, momentum, mass and particle constants. Edges (E) can represent forces between the particles and 'Globals' (u) can represent constants such as air density, the gravitational constant etc. We note that graphs can be fully connected or sparse, depending on the specific use case. In the particle system, it is possible to neglect particle interactions beyond a certain distance and sparsify what would otherwise be a fully connected graph.

In representing physical systems this way, we impart structure on our data which forms an important inductive bias when learning physics [2, 1, 14]. Representations of this form allow us to carry out learning in multiple complex domains because such graphs can be input to Graph Neural Networks that define updates for the graph attributes.

2.2 Hamiltonian Neural Networks

In designing a network, the core operation of interest for many physical systems is one which accurately models the time evolution of the system. Recently, [6] demonstrated that dynamic predictions through time can be improved using Hamiltonian Neural Networks (HNNs) which endow models with a Hamiltonian constraint. The Hamiltonian is an important representation of a dynamical system because it is one of two approaches that generalizes classical mechanics. The Hamiltonian $\mathcal{H}$ is a scalar function of position $\mathbf{q} = (q_1, q_2, ..., q_M)$ and momentum $\mathbf{p} = (p_1, p_2, ..., p_M)$. It can be shown that this representation of a physical system is a symplectic form, which means the derivatives of the Hamiltonian with respect to its inputs yields the time derivatives of the inputs (see Eqn. 1.)

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}}, \quad \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial \mathbf{q}} \tag{1}$$

As a consequence, it is noted in [6] that by accurately learning a Hamiltonian, the system's dynamics can be naturally extracted through backpropagation. This information allows us to build two 1st-order differential equations which can be used to update the state space, $(\mathbf{q}, \mathbf{p})$. Equation 2 shows this integral, in which we define $\mathbf{S} = [\frac{\partial \mathcal{H}}{\partial \mathbf{p}}, -\frac{\partial \mathcal{H}}{\partial \mathbf{q}}]$:

$$(\mathbf{q}, \mathbf{p})_{t+1} = (\mathbf{q}, \mathbf{p})_t + \int_t^{t+1} \mathbf{S}(\mathbf{q}, \mathbf{p}) \mathrm{d}t \tag{2}$$

However, this is not the only benefit in learning a Hamiltonian. Another key attribute of the Hamiltonian is that the vector field $\mathbf{S}$ is a symplectic gradient meaning $\mathcal{H}$ remains constant as long as state vectors are integrated along $\mathbf{S}$ [6]. This result links the Hamiltonian with the total energy of the system $\mathcal{H}(\mathbf{q}, \mathbf{p}) = E_{tot}$. Therefore, the Hamiltonian is a powerful inductive bias that can evolve a physical state and maintain energy conservation.

2.3 Variational Integrator Networks

Lagrangian mechanics offers an alternative to the Hamiltonian in generalizing a dynamical system. Rather than position and momentum (canonical coordinates) defining the state space, Lagrangian mechanics is defined using a generalized coordinate state space $(\mathbf{q}, \dot{\mathbf{q}})$. This is particularly useful in physical settings where the description and measurement of generalized coordinates may be easier to work with than canonical coordinates. Given these coordinates, Joseph-Louis Lagrange showed that a scalar value $\mathcal{A}$ referred to as the action can be defined as the integral of a Lagrangian, $\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})$:

$$\mathcal{A} = \int_{t}^{t+1} \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) \mathrm{d}t \tag{3}$$

The integral can be thought as inducing multiple paths between points in state space i.e. multiple walks in the domain of $(\mathbf{q}, \dot{\mathbf{q}})$. However, only one path is a stationary state of the action integral. This state lets us move from $t \rightarrow t + 1$ with minimal energy. It can be shown, through variational calculus, that this stationary state must satisfy the Euler-Lagrange equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}\right) = \frac{\partial \mathcal{L}}{\partial \mathbf{q}} \tag{4}$$

Although complex in form, the action integral and the Euler-Lagrange equations can be discretized and collectively form the basis for variational integrators. The work in [11] shows that, by adopting this approach, one can develop Variational Integrator Networks which make network learning in noisy data-settings more robust. Similar to Hamiltonians, Lagrangians in classical mechanics are also connected to the kinetic energy $\mathcal{T}$ and potential energy $\mathcal{V}$ via:

$$\mathcal{L} = \mathcal{T}(\mathbf{q}, \dot{\mathbf{q}}) - \mathcal{V}(\mathbf{q}, \dot{\mathbf{q}}) \tag{5}$$

Furthermore, Variational Integrators are symplectic and momentum conserving [8].

2.4 Related Work

Numerous approaches have identified that learning gradients during training can drastically improve learning. For example, early work by Witkoskie and Doren [16] highlights the significant improvement in predicting potential energy surfaces for materials borne out of learning both the potential energy hyper-surface and its gradients. The authors demonstrate that in contrast to directly learning the potential alone, the inclusion of gradients drives the network to accurately model both the forces and the potential surface. This bias therefore induces data-efficient learning and more accurate representations of the potential energy surface. Pukkritayakamee *et al.* [10] show that this simple approach can be extended to more complex, non-separable potentials, highlighting the flexibility of well-chosen biases.

More recently, NeuralODE [5], has re-sparked an interest in inductive biases. In this paper, the authors highlight the core link between Euler discretization steps and residual blocks, which allows them to create an end-to-end neural ODE solver. Inspired by this work, [6] and [15] show that one can take a physical system (position and momentum values, or e.g. images of a moving pendulum) and map to a latent space which can encode the dynamics of the system. One way to learn these dynamics is to train the network to directly output time derivatives. However, [6] show that using this simple approach is not enough to accurately predict energy conserving phase space trajectories. Instead, the authors show that if we treat the latent space as a vector field (symplectic gradient) i.e. as a parametric function for the Hamiltonian, then we can differentiate it with respect to the input (p and q) and obtain the time derivatives of the system. With these derivatives accurately learnt, we can use a NeuralODE-style integration scheme to evolve a system. In this way, the neural network is parametrized to encode the Hamiltonian of the system as well as the integrative inductive-bias.

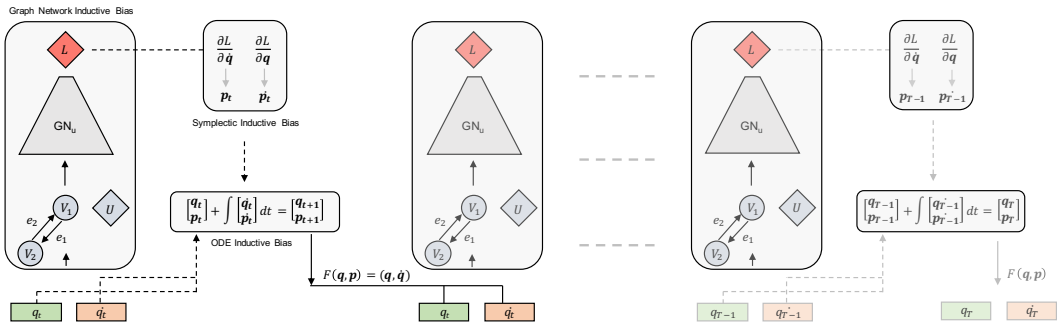


Figure 1: Variational Integrator Graph Network (VIGN) architecture highlighting the combination of the three inductive biases used to rollout an initial state. The figure highlights the extension of single step learning (to q_{t+1}) to long range learning (to q_T).

Although Hamiltonian neural nets predict dynamics for a single particle in a single dimension well (e.g. a swinging pendulum or mass spring system) their performance suffers in predicting large N-body problems. The work in [12] shows that graph networks are ideal for resolving this type of system. Nodes are treated as particles, with node information including position, momentum and any constants, such as spring coefficients and masses. A graph network can be trained to update the state of the nodes and output (e.g.) the Hamiltonian. Differentiating the latter with respect to the input nodes provides the time dynamics, which can in turn be fed through a Runge-Kutta integrator.

An alternative approach to this method is that of Variational Integrator Networks, proposed in [11]. Inspired by Neural-ODEs, this work proposes a neural net whose architecture matches the discrete equation of motion governing the dynamical system, as derived by applying the Euler-Lagrange equations to a discretized action integral. The paper indicates major benefits when using the method for noisy data settings, as well as providing precise energy and momentum conservation.

Our method connects the latter two approaches and allows us to propose a solution to the challenge of simultaneously solving large N-body problems and noisy data settings.

3 Method

The architecture for our method is shown in Fig.1. The network takes as input state vectors $\mathbf{s} = (\mathbf{q}, \dot{\mathbf{q}})$ representing generalized, or canonical, coordinates and learns to predict the Lagrangian and its derivatives with respect to these inputs. The graph network is built using the DeepMind graph network library. Each of the node, edge and global networks consist of two 64 multi-layer perceptrons with softplus activations. In the noiseless setting, the training loss is defined as the mean-square error across all time steps and across all state vectors. In noisy settings, we follow a similar approach to [11] and compute the full log-likelihood of the predicted state vector, $\mathbf{s}_{pred}$, using Eqn. 6:

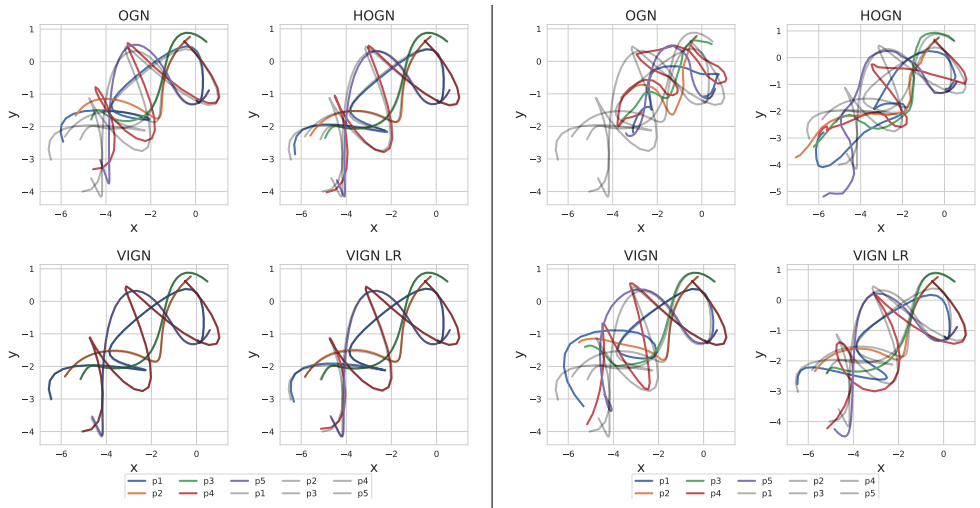
$$P(\mathbf{s}_{pred}|\mathbf{s}, \sigma^2) = \prod_{t=1}^{T} \mathcal{N}(\mathbf{s}_{pred}|\mathbf{s}, \sigma^2 I) \tag{6}$$

The function $\mathbf{F}(\mathbf{q}, \mathbf{p})$ is a coupling equation which maps $(\mathbf{q}, \dot{\mathbf{p}}) \rightarrow (\mathbf{q}, \dot{\mathbf{q}})$. In simple, classical systems the function is $[I, M^{-1}]$. One of the key innovations we introduce in the VIGN is the interwoven symplectic and ODE inductive bias learnt by a Graph Network. Both Hamiltonian and Lagrangian approaches allow us to arrive at the coupled symplectic inductive bias used to update the state in Fig. 1.

3.1 Hamiltonian Approach

In the Hamiltonian approach we know the following must hold:

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\mathrm{d}\mathcal{H}}{\mathrm{d}p}, \quad \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}\mathbf{q}}. \tag{7}$$



(a) Noiseless Data Setting

(b) Noisy Data Setting

Figure 2: A 40-step rollout of the 5-body spring problem. In (a) models are trained on noiseless data, and in (b), on noisy data. In both settings, VIGN and its long range counterpart (VIGN LR) outperform OGN and HOGN. Colours represent predicted particle trajectories.

However, we also know from classical mechanics that:

$$\mathbf{p} = M\mathbf{v} = M\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} \tag{8}$$

Substituting 8 in 7 yields:

$$\dot{\mathbf{q}} = -M^{-1}\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}\mathbf{q}}. \tag{9}$$

The result is a second order differential equation which can be split into two first order differential equations and integrated, allowing us to learn a single potential and couple the learning of generalized position and momentum. This method therefore only requires the derivative of $\mathcal{H}$ with respect to $\mathbf{q}$ and as such, avoids the redudancy of learning a position update which is already embedded in the input data.

3.2 Lagrangian Method

It is also possible to arrive at the same result using Variational Integrators which provide a more general solution than the one in Eqn.9. Variational Integrators discretize the action integral $\mathcal{A}$ of the Lagrangian $\mathcal{L}$:

$$\mathcal{L}^d(\mathbf{q}_t, \mathbf{q}_{t+1}, h) \approx \int_t^{t+h} \mathcal{L}(\mathbf{q}, \dot{\mathbf{q}})\mathrm{d}\tau \tag{10}$$

Once discretized, the Euler-Lagrange equations, coupled with the separable Newtonian Lagrangian (Eqn. 11), can be used to obtain the Störmer-Verlet equation (Eqn. 12):

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = \mathcal{T}(\dot{\mathbf{q}}) - \mathcal{V}(\mathbf{q}) = \frac{1}{2}\dot{\mathbf{q}}^T M\dot{\mathbf{q}} - \mathcal{V}(\mathbf{q}) \tag{11}$$

$$\mathbf{q}_{t+1} = 2\mathbf{q}_t - \mathbf{q}_{t-1} - h^2 M^{-1}\frac{\partial \mathcal{V}(\mathbf{q})}{\partial \mathbf{q}} \tag{12}$$

Notice, the Störmer-Verlet equation looks like a discretized second-order differential equation and has a truncation error of $O(h^2)$. The key distinction between this approach and the Hamiltonian is that it allows us to represent information in terms of generalized coordinates and naturally couples the dynamics of momentum with position.

To obtain higher order variational integrators, the work in [9] considers discretizing the Lagrangian by setting the elements of the Lagrangian to polynomials of degree s:

$$\mathcal{L}^d(\mathbf{q}_0, \mathbf{q}_1) = h \sum_{i=1}^{s} b_i \mathcal{L}(\mathbf{Q}_i, \dot{\mathbf{Q}}_i) \tag{13}$$

where

$$\mathbf{Q}_i = \mathbf{q}_0 + h \sum_{i=1}^{s} a_{ij} \dot{\mathbf{Q}}_\mathbf{j}, \mathbf{q}_1 = \mathbf{q}_0 + h \sum_{i=1}^{s} b_i \dot{\mathbf{Q}}_i \tag{14}$$

If we extremize this Lagrangian with respect to $\dot{\mathbf{Q}}$ (details of which can be found in the Appendix), [9] show that we obtain a variational integrator which allows us to update position and momentum through the following equations:

$$\mathbf{Q}_i = \mathbf{q}_0 + h \sum_{i=1}^{s} a_{ij} \dot{\mathbf{Q}}_\mathbf{j}, \;\; \mathbf{P}_i = \mathbf{p}_0 + h \sum_{j=1}^{s} \hat{a}_{ij} \dot{\mathbf{P}}_j \tag{15}$$

$$\mathbf{q}_1 = \mathbf{q}_0 + h \sum_{i=1}^{s} b_i \dot{\mathbf{Q}}_i, \;\; \mathbf{p}_1 = \mathbf{p}_0 + h \sum_{i=1}^{s} b_i \dot{\mathbf{P}}_i \tag{16}$$

The result of extremizing the integral in this way is a partitioned Runge-Kutta method. The resulting equations with appropriately selected coefficients can build higher order integrators which are equivalent to off-the-shelf Runge-Kutta methods in scientific computing libraries. For example, it can readily be shown that the equations provide a generalized higher-order form of the Störmer-Verlet equation. If we set the Lagrangian to the separable Newtonian in Eqn.12 then we obtain the following equations:

$$M\dot{\mathbf{Q}}_i = \mathbf{P}_i, \dot{\mathbf{P}}_i = -\frac{\partial \mathcal{V}(\mathbf{Q}_i)}{\partial \mathbf{q}} \tag{17}$$

By combining Equations 15,16 and 17, we arrive at the following update equations and partitioned Runge-Kutta method:

$$\mathbf{q}_1 = \mathbf{q}_0 + h \sum_{i=1}^{s} b_i k_i, \mathbf{p}_1 = \mathbf{p}_0 + h \sum_{i=1}^{s} b_i l_i \tag{18}$$

$$k_i = \mathbf{P}_i * M^{-1} = M^{-1}\mathbf{p}_0 + hM^{-1} \sum_{j=1}^{s} a_{ij} l_j \tag{19}$$

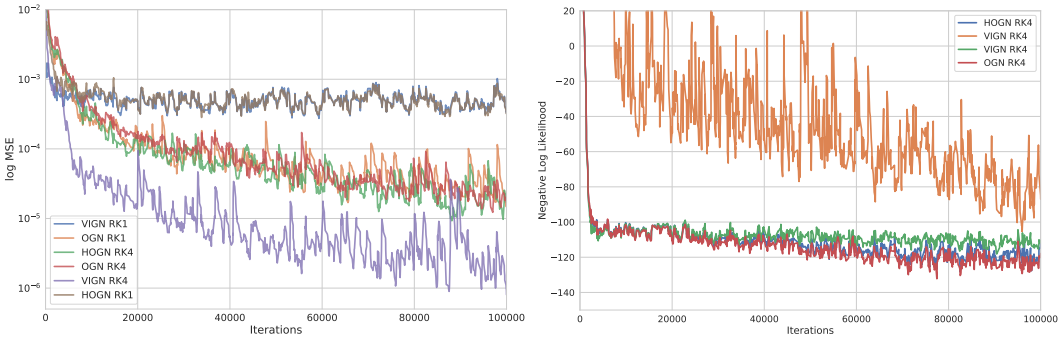
$$l_i = -\frac{\mathrm{d}\mathcal{V}(\mathbf{q}_0 + \sum_{j=1}^{s} a_{ij} k_j)}{\mathrm{d}\mathbf{q}} \tag{20}$$

Setting all b coefficients to $1/2$ and a coefficients to $1/2$, the above equations reduce to the explicit Störmer-Verlet equation [7] which provides mid-point updates for both position and momentum.

4 Experiments

We carry out our experiments on a dataset similar to that found in [12]. We develop an N-body dataset, where the interaction force between particles is modelled by $\mathbf{F}_{ij} = -k_i k_j (\mathbf{q}_i - \mathbf{q}_j)$. We randomly sample initial conditions according to the following conditions:

- Mass $m_i \in [0.1, 1]$
- Spring Constant $k_i \in [0.5, 1]$
- Position $\mathbf{q}_i \in [-1, 1]^2$
- Velocity $\mathbf{v}_i \in [-3, 3]^2$



(a) Noiseless Setting Mean Squared Error

(b) Noisy Setting Negative Log Likelihood

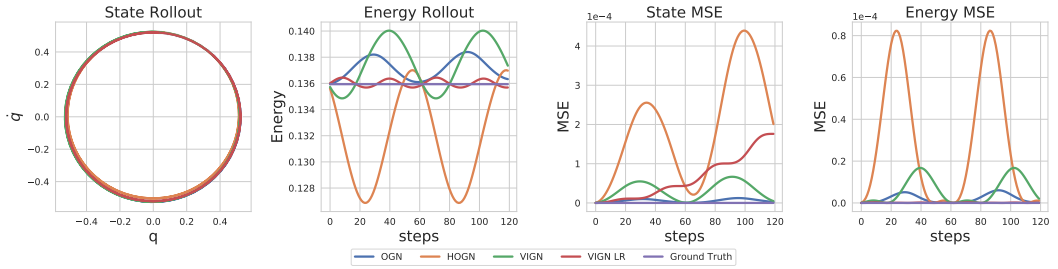
Figure 3: Training across 100,000 iterations on a 5-body noiseless (a), and noisy (b), spring force problem given optimal learning rates. 4th-order VIGN rapidly learns the underlying dynamics compared to existing methods. In (b), we observe VIGN does not overfit to noise, leading to improved inference.

Using these initial conditions, we use a Runge-Kutta-4 integrator with $r_{tol} = 10^{-9}$ and $a_{tol} = 10^{-9}$ to integrate the system to $T = 4.005$ with a $\delta t = 0.005$. We then sub-sample the results at $\delta t_{ssr} = 0.1$, which implies each initial condition results in $\frac{T}{\delta t} \times \frac{\delta t}{\delta t_{ssr}} = 41$ samples. We use 100 initial conditions for training and 20 for evaluation. We fix N to 5, although the graph networks remain flexible to learning from data which contains a mix of number of particles. To build the noisy data, we follow the approach of [6] and [11] by sampling from a Gaussian $\mathcal{N}(0, 0.1^2)$ and adding this noise to each state vector, meaning position values and momentum values are assumed to have noise. All networks are trained using AdamOptimizer for 100,000 iterations with a batch size of 40. Optimal learning rates for each method are obtained by training each network on 10 rates evenly sampled between 10^{-4} and 10^{-1}.

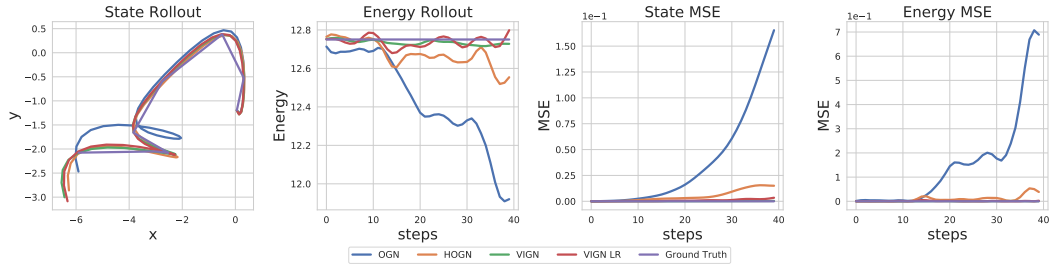
In addition, we also carry out testing on a simple 1-D mass-spring system where the mass and spring constant are set to 1. We use 100 initial conditions which satisfy the condition that their radius squared is distributed uniformly between 0.2 and 1. For each initial condition we rollout the state to $T_{max} = 6$ where $\delta t = 0.1$. We train the methods for 10,000 iterations and test them on 120 step rollouts.

5 Discussion

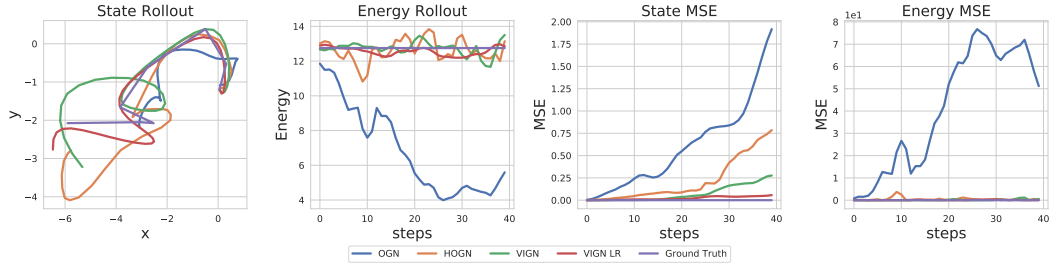
In Fig.3 we compare training across methods for both noiseless and noisy data. We find that 4th-order VIGN methods significantly outperform existing approaches in rapidly learning from data, particularly when only a limited number of training trajectories are provided. This represents a core benefit to learning in physics-based problems which are data limited. However, we do note that 1st-order OGN methods perform better than 1st-order VIGN and HOGN methods. As pointed out by [12], this is because the OGN is not constrained to obey physical laws whilst 1st-order methods which do obey physical laws cannot fully learn the dynamics. In Fig. 3, we also see that VIGN methods in the noisy data setting have a (negative) log-likelihood loss which is higher than existing methods. Although this might seem like other methods do better, they are fundamentally over-fitting to the data. As a consequence, VIGN LR methods outperform other approaches by at least an order of magnitude (Fig. 5). The reported rollout error is the mean squared error across all samples and state vectors. We highlight one of the trajectories during inference for the both noisy and noiseless settings in Fig. 2. As we can see, in the noiseless setting we have tight matching with the ground truth when using VIGN and we see that in the noisy setting, VIGN still learns to accurately capture the general dynamics. We quantify matching, in terms of position, momentum and energy in Fig. 4. In all settings we see VIGN and its long range counterpart demonstrating tight state (p, q) matching and tight energy matching.



(a) 1-body Mass Spring System



(b) 5-body Spring Force System (Noiseless)



(c) 5-body Spring Force System (Noisy)

Figure 4: State and Energy rollouts and MSE values over time. In each State rollout we simply show the trajectory of 1 particle for the 5-body problem and a phase space plot for the 1-body problem. MSE values are computed across all state vectors at the given time step. VIGN consistently outperforms existing methods.

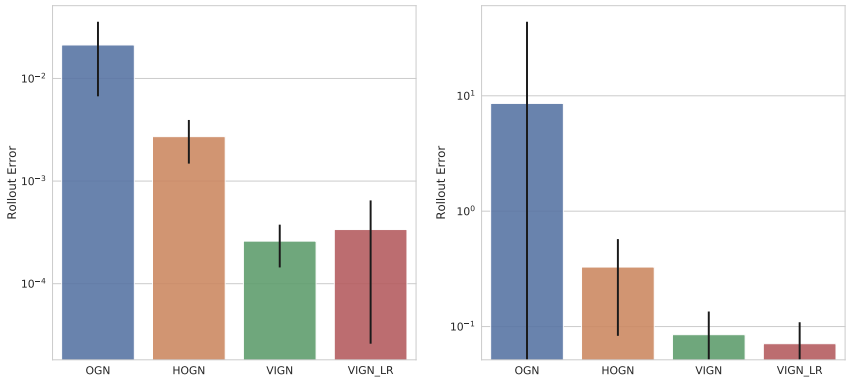


Figure 5: Best rollout error on a 40-step trajectory for noiseless (left) and noisy (right) settings, averaged over 20 initial conditions of the 5-body spring problem. One-step and multi-step VIGN RK4 perform at least an order of magnitude better than HOGN and OGN on both tasks.

We have shown that learning physics from data strongly benefits from well-chosen inductive biases. VIGN is one such method which combines existing approaches and unifies learning. In doing so, VIGN:

- unifies graph network, ODE and symplectic inductive biases,
- makes learning data-efficient,
- works profoundly well in noisy data settings through coupled learning of q and $\dot{q}$,
- maintains flexibility to learn from canonical momenta or generalized momenta,
- maintains flexibility to build higher order VIs through partitioned Runge-Kutta methods.

VIGN is therefore more realistic at solving classical physics problems where data may be scarce, for example, in materials science applications or where data is noisy, for example, in robotics. Performance gains do indeed come at a price and we note that long-range VIGN training times are roughly twice as long as those of other methods.

Acknowledgments

We would like to thank the Oxford Centre for Doctoral Training in Autonomous, Intelligent Machines and Systems (AIMS) and the Rhodes Trust for supporting this research.

References

[1] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261 [cs, stat]*, October 2018. arXiv: 1806.01261.

[2] Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction Networks for Learning about Objects, Relations and Physics. *arXiv:1612.00222 [cs]*, December 2016. arXiv: 1612.00222.

[3] Christopher P. Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. MONet: Unsupervised Scene Decomposition and Representation. *arXiv:1901.11390 [cs, stat]*, January 2019. arXiv: 1901.11390 version: 1.

[4] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. Reversible Architectures for Arbitrarily Deep Residual Neural Networks. *arXiv:1709.03698 [cs, stat]*, November 2017. arXiv: 1709.03698.

[5] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations. *arXiv:1806.07366 [cs, stat]*, January 2019. arXiv: 1806.07366.

[6] Sam Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian Neural Networks. *arXiv:1906.01563 [cs]*, June 2019. arXiv: 1906.01563.

[7] E. Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*. Number 31 in Springer series in computational mathematics. Springer, Berlin ; New York, 2nd ed edition, 2006. OCLC: ocm69223213.

[8] A. Lew, J. E. Marsden, M. Ortiz, and M. West. Variational time integrators. *International Journal for Numerical Methods in Engineering*, 60(1):153–212, 2004. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.958.

[9] J. E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, May 2001.

[10] A. Pukrittayakamee, M. Malshe, M. Hagan, L. M. Raff, R. Narulkar, S. Bukkapatnum, and R. Komanduri. Simultaneous fitting of a potential-energy surface and its corresponding force fields using feedforward neural networks. *The Journal of Chemical Physics*, 130(13):134101, April 2009.

[11] Steindor Saemundsson, Alexander Terenin, Katja Hofmann, and Marc Peter Deisenroth. Variational Integrator Networks for Physically Meaningful Embeddings. *arXiv:1910.09349 [cs, stat]*, October 2019. arXiv: 1910.09349.

[12] Alvaro Sanchez-Gonzalez, Victor Bapst, Kyle Cranmer, and Peter Battaglia. Hamiltonian Graph Networks with ODE Integrators. *arXiv:1909.12790 [physics]*, September 2019. arXiv: 1909.12790.

[13] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to Simulate Complex Physics with Graph Networks. *arXiv:2002.09405 [physics, stat]*, February 2020. arXiv: 2002.09405.

[14] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. *arXiv:1806.01242 [cs, stat]*, June 2018. arXiv: 1806.01242.

[15] Peter Toth, Danilo Jimenez Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian Generative Networks. *arXiv:1909.13789 [cs, stat]*, September 2019. arXiv: 1909.13789.

[16] James B. Witkoskie and Douglas J. Doren. Neural Network Models of Potential Energy Surfaces: Prototypical Examples. *Journal of Chemical Theory and Computation*, 1(1):14–23, January 2005.

Appendix

Symplectic partitioned Runge-Kutta methods

Consider the discrete Lagrangian:

$$L_h(q_0, q_1) = h \sum_{i=1}^{s} b_i L(Q_i, \dot{Q}_i) \tag{21}$$

where

$$Q_i = q_0 + h \sum_{j=1}^{s} b_i \dot{Q}_j \tag{22}$$

and $\dot{Q}_i$ extremize the above Lagrangian under the constraint:

$$q_1 = q_0 + h \sum_{i=1}^{s} b_i \dot{Q}_i \tag{23}$$

The b_i are non-zero and their sum equals 1. With a Lagrange multiplier $\lambda = (\lambda_1 ..., \lambda_d)$ for the constraint, the extremality conditions obtained by differentiating the Lagrangian with respect to $\dot{Q}$ for each s results in:

$$\sum_{i=1}^{s} b_i \frac{\partial L}{\partial q}(Q_i, \dot{Q}_i) h a_{ij} + b_j \frac{\partial L}{\partial \dot{q}}(Q_j, \dot{Q}_j) = b_j \lambda \tag{24}$$

where:

$$\dot{P}_i = \frac{\partial L}{\partial q}(Q_i, \dot{Q}_i), P_i = \frac{\partial L}{\partial \dot{q}}(Q_i, \dot{Q}_i) \tag{25}$$

By rearranging and simplifying we obtain:

$$b_j P_j = b_j \lambda - h \sum_{i=1}^{s} b_i a_{ij} \dot{P}_i \tag{26}$$

Symplectic methods imply:

$$p_0 = -\frac{\partial L_h}{\partial x}(q_0, q_1)$$

which results in:

$$p_0 = -h \sum_{i=1}^{s} b_i \dot{P}_i + \lambda$$

and

$$p_1 = \lambda$$

Combining these results gives the update equations specified in the Lagrangian method section.

$$Q_i = q_0 + h \sum_{i=1}^{s} a_{ij} \dot{Q}_j, P_i = p_0 + h \sum_{j=1}^{s} a_{ij} \dot{P}_j \tag{27}$$

$$q_1 = q_0 + h \sum_{i=1}^{s} b_i \dot{Q}_i, p_1 = p_0 + h \sum_{i=1}^{s} b_i \dot{P}_i \tag{28}$$

with $a_{ij} = b_j - b_j a_{ji}/b_i$ for symplecticity to hold.