Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 56-2 (2023) 3203–3210

Discrete Lagrangian Neural Networks with Automatic Symmetry Discovery

Yana Lishkova[†] * **Paul Scherer**[†] ** **Steffen Ridderbusch** * **Mateja Jamnik** ** **Pietro Liò** ** **Sina Ober-Blöbaum** *** **Christian Offen** ***

* *University of Oxford, Oxford, UK*
** *University of Cambridge, Cambridge CB3 0FD, UK*
*** *Paderborn University, D-33098 Paderborn, Germany*

Abstract: By one of the most fundamental principles in physics, a dynamical system will exhibit those motions which extremise an action functional. This leads to the formation of the Euler-Lagrange equations, which serve as a model of how the system will behave in time. If the dynamics exhibit additional symmetries, then the motion fulfils additional conservation laws, such as conservation of energy (time invariance), momentum (translation invariance), or angular momentum (rotational invariance). To learn a system representation, one could learn the discrete Euler-Lagrange equations, or alternatively, learn the discrete Lagrangian function $\mathcal{L}_d$ which defines them. Based on ideas from Lie group theory, we introduce a framework to learn a discrete Lagrangian along with its symmetry group from discrete observations of motions and, therefore, identify conserved quantities. The learning process does not restrict the form of the Lagrangian, does not require velocity or momentum observations or predictions and incorporates a cost term which safeguards against unwanted solutions and against potential numerical issues in forward simulations. The learnt discrete quantities are related to their continuous analogues using variational backward error analysis and numerical results demonstrate the improvement such models can have both qualitatively and quantitatively even in the presence of noise.

Copyright © 2023 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Variational integrators, Neural networks, Symmetry, Nonlinear system identification

It is a well-established theory from the field of variational calculus that the behaviour of an unforced system can be derived from the Lagrange-d'Alembert principle using only knowledge of the system's Lagrangian. The resulting Euler-Lagrange equations of motion act as a model describing how the system's configuration and velocity evolve as time progresses from an initial displacement and thus serve as a state space system description. Equivalently, one could describe the system in the phase space through the canonical equations of motion based only on knowledge of the system's Hamiltonian. To learn a system representation, one could identify the equations of motion directly, or alternatively, identify the Hamiltonian or the Lagrangian which defines them. Using these approaches Greydanus et al. (2019) proposed the Hamiltonian Neural Network (HNN), closely followed by the Lagrangian Neural Networks (LNN) by Cranmer et al. (2020). In combination with model reduction the Lagrangian learning technique has been applied to different real-life high dimensional problems such as video prediction (Allen-Blanchette et al., 2020).

Drawing from the field of mathematical physics we know that Hamiltonian and Lagrangian models encode additional information about the system such as symmetries. Through Noether's theorem, these symmetries relate to conservation laws which further restrict the motion of the system. The use of specialised integrators, known as geometric integrators, allows the preservation of these characteristics from the continuous into the discrete domain. This results in a better qualitative representation of the system and a preservation of the symmetries and energy accurately or up to small bounded oscillations for exponentially long times (Hairer et al., 2006).

Variational integrators are geometric integrators that are based on the theory of discrete mechanics (Marsden and West, 2001). With standard integrators, the equations of motion are first derived and then discretised. With variational integrators a discrete approximation of the Lagrangian is created and then used in a discrete Lagrange-d'Alembert principle leading to equations of motion which are structure-preserving. With this in mind Saemundsson et al. (2020) proposed learning of the Lagrangian with forward simulations using the variational integrator and demonstrated accurate long-term predictions even when learning from noisy data. This was extended for systems with external forcing by Havens and Chowdhary (2021). However, both works rely on a phase space formulation of the variational integrator, which requires data and prediction of both the configuration and the momentum.

When using a variational integrator, we can use a second formulation which removes the need to store observations or calculate predictions of the canonical momentum or the velocity in both the learning and testing stages. In this formulation an initial momentum value and the observations and predictions of the configuration are sufficient to

[†] These authors contributed equally to this work.

2405-8963 Copyright © 2023 The Authors. This is an open access article under the CC BY-NC-ND license.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2023.10.1457

model the system behaviour in time. This is an advantage compared to the use of standard integrators as noted by Aoshima et al. (2021) for an alternative energy-preserving method in the absence of external forcing. Another important competing approach are the symplectic recurrent neural networks (SRNN) by Chen et al. (2019), which learn the Hamiltonian and rely on a different symplectic integrator that is performant even in noisy systems. However, it again relies on observations from both the configuration and the momentum.

The approaches with variational integrators mentioned so far learn a continuous Lagrangian expression from discrete data and use a discrete approximation of this expression in the forward variational simulations. These approximations introduce discretisation errors in the results which could amplify the modelling error of a learnt Lagrangian (Ober-Blöbaum and Offen (2023)). To prevent these errors one can instead learn the discrete Lagrangian and use it for variational simulations in testing and training as demonstrated by Qin (2020) with numerical experiments on nonlinear oscillations and the Kepler problem, by Santos et al. (2022) for an assumed structure of a potential and a kinetic energy term learned with two separate neural networks, and in a PDE setting by Offen and Ober-Blöbaum (2023) for discrete Lagrangian densities.

Discrete Hamiltonians and discrete Lagrangians can be related to their continuous counterparts by backward error analysis (Hairer et al., 2006; Vermeeren, 2017). Ober-Blöbaum and Offen (2023) and Offen and Ober-Blöbaum (2022) exploit backward error analysis in a machine learning context in for analysis purposes as well as to compensate for discretisation errors when the learned discrete Lagrangian is used in numerical simulations to compute motions to given initial values. Using Gaussian Processes (GPs) they learn a so called inverse modified (continuous) Lagrangian that, after discretisation with a variational integrator, is consistent with the motion data of the true system (see Figure 1). Their regularisation terms are, however, tailored to GPs and are not suitable for neural networks.

Our proposed methodology aims to learn an inverse modified discrete Lagrangian without prior restriction on its form, incorporating additional steps to build upon the strengths of previous approaches and subvert their limitations. For example, learning the inverse modified discrete form avoids discretisation error and requires observations only of the configuration alone, without the need to obtain or simulate the velocity. Furthermore, we propose an additional *degeneracy* cost term which inductively biases the network to avoid learning discrete Lagrangians whose equations of motion have degenerate roots and pose problems for forward simulation incorporating numerical root-solving techniques. The additional term also inductively biases the neural network to avoid learning non-desirable constant Lagrangian solutions. With these improvements, the first of our two proposed methods, the *Discrete Lagrangian Neural Network* (DLNN) successfully incorporates the use of variational integrators and variational backward error analysis in a novel neural network based model, which uses the underlying structure of the problem to improve the representation qualitatively and explicitly guard against numerical issues in forward integration.

As another contribution, we introduce a novel method to automatically learn variational symmetries of a dynamical system along with the system's discrete Lagrangian. More precisely, we introduce a framework that for a given Lie group action on the configuration space identifies a subgroup which acts by symmetries. Through construction of a momentum map, conserved quantities are derived. While our framework considers variational symmetries of discrete actions, a framework for symplectic symmetries of Hamiltonian systems (SymHNN) was developed in Dierkes et al. (2022). Inductive biases borne out of the incorporation of symmetries into learning algorithms have previously shown desirable behaviours in the resulting models such as reduced sample complexity and improved generalisation whilst significantly reducing model complexity (Dehmamy et al., 2021).Our framework is explicitly worked out for the group of affine linear transformations, which consists of arbitrary compositions of translation, rotations, and scaling transformations. We demonstrate in numerical examples that our proposed *Symmetric Discrete Lagrangian Neural Network* (SymDLNN) successfully learns symmetries and conservation laws and that incorporation of symmetries improves the predictions compared to previous approaches.

To summarise, **our contributions** are:

- **Symmetry framework.** We introduce a novel framework based on Lie group theory to automatically discover and incorporate symmetries and conservation laws during model training.
- **Numerical analysis informed learning.** We propose a novel methodology for learning inverse modified discrete Lagrangians from snapshot observations of motions. Our new regularisation term guarantees that the learned model can be efficiently employed in the computation of trajectories.
- Use of variational backward error analysis in combination with the learnt discrete Lagrangian.

1. BACKGROUND

1.1 Continuous Lagrangian dynamics

Consider a mechanical system, whose configuration $q(t) \in \mathbb{R}^{n_q}$ evolves on a configuration manifold Q, with associated tangent bundle TQ. The behaviour of the system in time can be described in state space by the evolution of its configuration vector $q(t) \in Q$ and its associated velocity vector $\dot{q}(t)$ such that $(q(t), \dot{q}(t)) \in TQ$. The system is assumed to possess a regular Lagrangian $\mathcal{L} : TQ \to \mathbb{R}$, which is not explicitly time dependent. It is known that the motion of the system is governed by the Lagrange-d'Alembert principle, which requires that

$$\delta \int_{t_0}^{t_f} \mathcal{L}(q, \dot{q})\, dt = 0$$

is satisfied for all variations δq with $\delta q(t_0) = \delta q(t_f) = 0$ (Marsden and West, 2001). Using integration by parts this principle results in the following equations:

$$D_1\mathcal{L}(q, \dot{q}) - \frac{d}{dt} D_2\mathcal{L}(q, \dot{q}) = 0 \tag{1}$$

known as the Euler-Lagrange equations of motion (Liberzon, 2011). Here D_i is the partial derivative operator with

respect to the i-th argument and the dependencies on time have been omitted to simplify the notation. Using these equations one can model how the system evolves in the state space over time. The system's evolution can equivalently be described in phase space using the configuration vector q and conjugate momenta p defined as $p = D_2\mathcal{L}(q, \dot{q})$. Notably for a given system, the Lagrangian which satisfies Equation (1) is not unique and thus the conjugate momentum is not unique either. For example, any constant function could satisfy the equation but would not lead to any dynamics. This is important for schemes attempting to learn the Lagrangian function as one would need to safeguard against constant solutions.

1.2 Discrete Lagrangian mechanics

In the discrete time setting TQ is replaced with $Q \times Q$ and a discrete configuration path is defined as $q_d(t_k) = q_k$ with $q_k \approx q(t_k)$ for $t_k = t_0 + k\Delta T$ and $k = 0, \ldots, N$ where $N = t_f/\Delta t$. Based on this discretisation a discrete Lagrangian $\mathcal{L}_d$ is an approximation

$$\mathcal{L}_d(q_k, q_{k+1}) \approx \int_{t_k}^{t_{k+1}} \mathcal{L}(q, \dot{q})\, dt, \tag{2}$$

where q on the right hand side of (2) solves the boundary value problem (1) with $q(t_k) = q_k$, $q(t_{k+1}) = q_{k+1}$. A discrete version of the Lagrange-d' Alembert principle

$$\delta \sum_{i=0}^{N-1} \mathcal{L}_d(q_k, q_{k+1}) = 0$$

leads to the discrete Euler-Lagrange equations:

$$D_2\mathcal{L}_d(q_{k-1}, q_k) + D_1\mathcal{L}_d(q_k, q_{k+1}) = 0 \tag{3}$$

To parallel the continuous theory one can define the discrete conjugate momentum p_k as $p_k = -D_1\mathcal{L}_d(q_k, q_{k+1})$ (see Marsden and West (2001)). Given q_0 and q_1 one can use equations (3) to time-step in time to compute q_k for $k > 1$. To compute q_1 and initiate the integration one can use the expression for the discrete momentum for the initial position q_0 and conjugate momentum p_0. Similarly to the continuous section techniques learning the discrete Lagrangian would need to safeguard against solutions which would satisfy Equation (3) but would not produce meaningful dynamics.

1.3 Standard and variational backward error analysis

Assume we have a system with continuous dynamics $\dot{x}(t) = f(x(t))$ and a numerical integrator $\dot{x}_{k+1} = z(x_k)$. This discrete method is only an approximation of the actual dynamics $x(k\Delta t)$ at time $t = k\Delta t$ or in other words $x_k \approx x(k\Delta t)$. Backward error analysis focuses on discovering dynamics $\dot{x}^m = f^m(x^m(t))$ for which equation $\dot{x}_{k+1} = z(x_k)$ is in fact the exact solution or in other words $x_k = x^m(k\Delta t)$. The dynamics $\dot{x}^m = f^m(x^m(t))$ are often termed as the modified equations. By studying the difference between x_k and $x^m(k\Delta t)$ one can determine important properties of the integrator scheme (Hairer et al., 2006).

Following from this idea, Vermeeren (2017) has developed a similar analysis for variational integrators, known as variational backward error analysis. As depicted in Figure 1, the true dynamics $q(t)$ of the system are governed by the continuous equations (1) when the true system Lagrangian

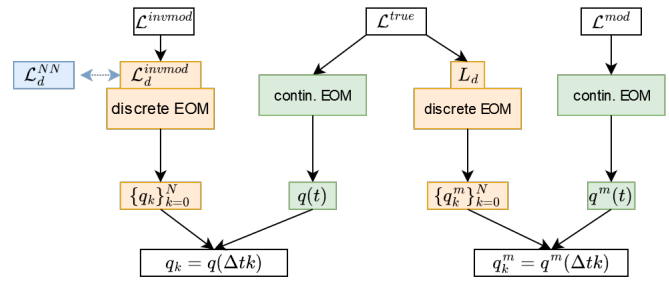

Fig. 1. Diagramatic explanation of the modified and inverse modified Lagrangian functions (contin.- continuous, EOM- equations of motion).

is used $\mathcal{L}^{true}$. The solutions of Equation (3) $\{q_k^m\}_{k=0}^N$ are only approximations of these dynamics. On the other hand Vermeeren (2017) derived a modified Lagrangian expression $\mathcal{L}^{mod}$ which when plugged in (1) would obtain a solution $q^m(t)$ such that $q^m(k\Delta t) = q_k^m$. Ober-Blöbaum and Offen (2023); Offen and Ober-Blöbaum (2022) on the other hand searched for an inverse modified Lagrangian $\mathcal{L}^{invmod}$ expression which, when plugged into the variational scheme would obtain discrete path q_k which accurately represents the true dynamics in the discrete domain i.e. $q_k = q(k\Delta t)$. He further describes how $\mathcal{L}^{invmod}$ can be used to obtain $\mathcal{L}^{true}$.

In this article, we propose to learn discrete inverse modified Lagrangians directly from data. Once it is learned, motions can be computed using Equation (3). To initialise the computation, initial data consisting of the first two positions (q_0, q_1) of the trajectory are required. For the purpose of system identification, the inverse modified Lagrangian $\mathcal{L}^{invmod}$ can be computed from its discrete learnt counterpart by a change of variables depending on the variational integrator and subsequently using variational backward error analysis formulas used to identify a function for the true Lagrangian, which will be denoted $\mathcal{L}^{VBEA}$.

For example, for a simulation using a step of size Δt and the variational midpoint rule in the one-dimensional case $\mathcal{L}^{invmod}(q, \dot{q}) = \mathcal{L}_d^{invmod}(q - \Delta t/2\dot{q}, q + \Delta t/2\dot{q})$, where $\mathcal{L}_d^{invmod}$ is the discrete Lagrangian that we learn, and

$$\mathcal{L}^{VBEA} = \mathcal{L}^{invmod} + \tfrac{1}{24}\Delta t^2 \frac{(\partial_q \mathcal{L}^{invmod} - \partial^2_{q\dot{q}}\mathcal{L}^{invmod}\dot{q})^2}{\partial^2_{\dot{q}\dot{q}}\mathcal{L}^{invmod}} - \tfrac{1}{24}\Delta t^2 \partial_{qq}\mathcal{L}^{invmod}\dot{q}^2 + O(\Delta t^4). \tag{4}$$

Formulas for higher dimensions or truncation orders can be derived as explained in Vermeeren (2017).

1.4 Variational symmetries, conserved quantities, and a framework for our symmetry learning method

Dynamical systems governed by Euler-Lagrange equations can exhibit symmetries. These are of great significance since they encode important qualitative features of motions: for instance, the idealised motion of a pendulum on a cart is described by Equation (1) with the Lagrangian

$$\mathcal{L}_{\text{cp}}(s, \phi, \dot{s}, \dot{\phi}) = \frac{1}{2}(\alpha\dot{\phi}^2 + 2\beta\cos(\phi)\dot{s}\dot{\phi} + \gamma\dot{s}^2) + D\cos(\phi), \tag{5}$$

with position variables $q = (s, \phi)$ and velocity variables $\dot{q} = (\dot{s}, \dot{\phi})$, where $\alpha = m_1 l^2$, $\beta = m_1 l$, $\gamma = m_2 + m_1$,

and $D = -m_1 gl$. Here m_2 is the mass of the cart, m_1 the mass of the pendulum, l the length of the pendulum, g a gravity acceleration constant. The Lagrangian $\mathcal{L}_{\text{cp}}$ does not depend explicitly on the variable s. In other words, it is invariant under translations of s, that is, the action $s \mapsto s + \eta$ for $\eta \in \mathbb{R}$ and $\frac{\partial \mathcal{L}_{\text{cp}}(q,\dot{q})}{\partial s} = 0$ for all $(q, \dot{q})$. By Noether's theorem, this symmetry corresponds to the conservation of the conjugate momentum

$$I_{\text{cp}} = \frac{\partial \mathcal{L}_{\text{cp}}}{\partial \dot{s}} = \beta \cos(\phi)\dot{\phi} + \gamma \dot{s} \tag{6}$$

along motions $q(t) = (s(t), \phi(t))$. More generally, if for a Lagrangian $\mathcal{L}(q, \dot{q})$ with $q \in Q = \mathbb{R}^{n_d}$ there exists a direction $w \in \mathbb{R}^{n_d}$ such that $\mathcal{L}$ is invariant under an action $(q, \dot{q}) \mapsto (q + sw, \dot{q})$ for all $s \in \mathbb{R}$ then the directional derivative

$$w^\top \nabla_q \mathcal{L}(q, \dot{q}) = \sum\nolimits_{j=1}^{n_d} w_j \frac{\partial \mathcal{L}}{\partial q_j}(q, \dot{q}) = 0 \tag{7}$$

and the quantity

$$I(q, \dot{q}) = w^\top \nabla_{\dot{q}} \mathcal{L}(q, \dot{q}) = \sum\nolimits_{j=1}^{n_d} w_j \frac{\partial \mathcal{L}}{\partial \dot{q}_j}(q, \dot{q}) \tag{8}$$

are preserved along motions $q(t)$ of Equation (1). Even more generally, an affine linear transformation is described by an invertible matrix $W \in \text{Gl}(\mathbb{R}, n_q)$ and a vector $w \in \mathbb{R}^{n_q}$ and acts on $(q, \dot{q}) \in TQ$ by $(q, \dot{q}) \mapsto (Wq + w, W\dot{q})$. Here $\text{Gl}(\mathbb{R}, n_q)$ denotes the group of invertible matrices. The matrix W encodes a rotation and scaling, and the vector w encodes a translation. We can write W and w into a matrix $\begin{pmatrix} W & w \\ 0 & 1 \end{pmatrix} \in \text{Gl}(\mathbb{R}, n_q + 1)$. Affine transformations form a group G which can be represented by the following subgroup of $(n_q+1)\times(n_q+1)$-dimensional invertible matrices

$$G_{\text{aff}} = \left\{ \begin{pmatrix} W & w \\ 0 & 1 \end{pmatrix} \,\middle|\, A \in \text{Gl}(\mathbb{R}, n_q),\ w \in \mathbb{R}^{n_q} \right\}.$$

Any 1-dimensional subgroup of G_{aff} can be defined by a matrix $M \in \mathbb{R}^{n_q \times n_q}$ and vector $w \in \mathbb{R}^{n_q}$ and is of the form

$$G_{(M,w)} = \left\{ \exp\left(\begin{pmatrix} \eta M & \eta w \\ 0 & 1 \end{pmatrix} \right) \,\middle|\, \eta \in \mathbb{R} \right\} \subset G,$$

where exp is the matrix exponential. If $G_{(M,w)}$ acts by symmetries, that is, if the Lagrangian $\mathcal{L}$ is invariant under actions by the elements in $G_{(M,w)}$, then

$$(Mq + w)^\top \nabla_q \mathcal{L}(q, \dot{q}) + M^\top \nabla_{\dot{q}} \mathcal{L}(q, \dot{q}) = 0 \tag{9}$$

for all $(q, \dot{q}) \in TQ$ and the quantity

$$I(q, \dot{q}) = (Mq + w)^\top \nabla_{\dot{q}} \mathcal{L}(q, \dot{q}) \tag{10}$$

is conserved along motions defined by the Euler-Lagrange Equation (1). When working with discrete Lagrangians $\mathcal{L}_d \colon Q \times Q \to \mathbb{R}$, the symmetry condition (9) is replaced by

$$(Mq_0 + w)^\top \nabla_{q_0} \mathcal{L}(q_0, q_1) + (Mq_1 + w)^\top \nabla_{q_1} \mathcal{L}(q_0, q_1) = 0 \tag{11}$$

for all $(q_0, q_1) \in Q \times Q$ and the conserved quantity for discrete motions becomes

$$I(q_k, q_{k+1}) = -(Mq_k + w)^\top \nabla_{q_k} \mathcal{L}(q_k, q_{k+1}). \tag{12}$$

For details, we refer to the book by Marsden and Ratiu (1999). Our method SymDLNN automatically identifies subgroups $G_{(M,w)}$ of the group of affine linear transformations under which a (discrete) Lagrangian is invariant while the (discrete) Lagrangian of a system is learned. As a consequence, we identify the corresponding conserved quantity as well.

Lagrangians are not uniquely determined by the system's motion and non-symmetric Lagrangians can govern highly symmetric dynamical systems. Our method will guide the learning process to a symmetric Lagrangian. This regularises the learning process and improves predictions as we will demonstrate in numerical examples. Our approach is not restricted to affine linear symmetries or 1-dimensional symmetry groups. To present the most general framework, let us briefly introduce Lie group actions and invariant vector fields. See Marsden and Ratiu (1999) for details. For a Lie group G, let $\mathfrak{g}$ denote its Lie algebra and $\exp \colon \mathfrak{g} \to G$ the exponential map. Consider a group action $a \colon G \to \text{Diff}(Q)$, $g \mapsto a_g$. Here $\text{Diff}(Q)$ denotes the group of diffeomorphisms on $Q = \mathbb{R}^{n_q}$. In the setting of continuous Lagrangians, the group action can be prolonged to an action $A \colon G \to \text{Diff}(TQ)$, $g \mapsto A_g$ by defining $A_g(q, \dot{q}) = (a_g(q), Da_g(q)\dot{q})$, where $Da_g(q)$ is the Jacobi matrix of the diffeomorphism a_g at q. Here we have identified $\mathcal{M} = TQ \cong \mathbb{R}^{n_q} \times \mathbb{R}^{n_q}$. For the setting of discrete Lagrangians, the diagonal group action $A \colon G \to \text{Diff}(Q \times Q)$, $g \mapsto A_g$ with $A_g(q_0, q_1) = (a_g(q_0), a_g(q_1))$ is considered instead of the prolonged group action and $\mathcal{M} = Q \times Q$. In both cases, for $v \in \mathfrak{g}$ the left invariant vector field $\hat{v} \in \mathfrak{X}(\mathcal{M})$ is defined by

$$\hat{v}_z = \left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} A_{\exp(tv)}(z) \in T_z\mathcal{M}, \quad z \in \mathcal{M}.$$

These vector fields can be thought of as infinitesimal actions of the Lie group G on $\mathcal{M}$.

The idea of SymDLNN is to identify a basis $v^1, \ldots, v^K$ of a subspace of $V^{(K)} \subset \mathfrak{g}$ such that $\hat{v}^j_z(\mathcal{L}) = 0$ for all $z \in \mathcal{M}$ and $j = 1, \ldots, K$. Under mild assumptions on the group action, a momentum map $J \colon M \to V^{(K)*}$ can be constructed from which K functionally independent conserved quantities can be computed as $I^j(q, \dot{q}) = \langle J(q, \dot{q}), v^j \rangle$, where $\langle \cdot, \cdot \rangle$ denotes the dual pairing. Relating this theory back to the example of affine linear symmetries, the Lie algebra for the affine linear group G_{aff} can be written as the following subspace of $\mathbb{R}^{(n_q+1)\times(n_q+1)}$:

$$\mathfrak{g} = \left\{ \begin{pmatrix} M & w \\ 0 & 0 \end{pmatrix} \,\middle|\, M \in \mathbb{R}^{n_q \times n_q},\ w \in \mathbb{R}^{n_q} \right\} \tag{13}$$

SymDLNN seeks as many elements $v^j = \begin{pmatrix} M^j & w^j \\ 0 & 0 \end{pmatrix} \in \mathfrak{g}$ as possible for which the symmetry condition (9) (or (11) in case of discrete Lagrangians) holds and such that all $v^1, \ldots, v^{(K)}$ are linearly independent. Then (under non-triviality conditions on $\mathcal{L}$) the K quantities provided by Equation (10) or Equation (12) are functionally independent and are conserved under motions.

2. PROPOSED APPROACH

Our scheme aims to obtain a model for the true Lagrangian of the system with no restriction on the structure of the Lagrangian function apart from its regularity. This is done by first learning a discrete Lagrangian function $\mathcal{L}_d^{NN}$ parameterised by a fully connected multi-layer perceptron (MLP) corresponding to the discrete inverse modified Lagrangian $\mathcal{L}_d^{invmod}$. After the learning stage is completed,

these functions are used for forward simulation with a variational integrator to obtain an accurate discretisation q_k of the true dynamics. Then the continuous inverse modified Lagrangian is obtained using $\mathcal{L}^{invmod}(q,\dot{q}) = \mathcal{L}_d^{NN}(q - \Delta t\dot{q}/2, q + \Delta t\dot{q}/2)$ based on the variational midpoint rule and employing the theory of VBEA from Section 1.3, an expression of the true Lagrangian $\mathcal{L}^{VBEA}$ is obtained and used to identify the corresponding Hamiltonian $\mathcal{H}^{VBEA}$.

To learn $\mathcal{L}_d^{NN}$ we rely on observations and simulations only of the configuration in time and do not pose any restrictions on its form. If required, velocity data at step k can be computed using central difference or by solving

$$-D_1\mathcal{L}_d^{NN}(q_k, q_{k+1}) = \nabla_{\dot{q}}\mathcal{L}^{VBEA}(q,\dot{q})|_{(q_k,\,\dot{q}_k)}$$

for $\dot{q}_k$. Observations and predictions are created from configuration data using three consecutive values in discrete time (q_{k-1}, q_k, q_{k+1}) from the same test trajectory. Each triple is used to compute $D_2\mathcal{L}_d^{NN}(q_{k-1}, q_k) + D_1\mathcal{L}_d^{NN}(q_k, q_{k+1})$. As known from Equation (1) these values should be zero for all discrete points of the trajectory. As there is no requirement on the sequentiality between triple observations the neural network can be trained by minimising

$$\min \frac{1}{N}\sum_{k=0}^{N-1}\sum_{i=1}^{n_q}(D_2\mathcal{L}_d^{NN}(q_{k-1}, q_k) + D_1\mathcal{L}_d^{NN}(q_k, q_{k+1}))[i])^2 \tag{14}$$

via a (stochastic) gradient descent algorithm and backpropagation to compute the derivative of the loss function with respect to the parameters (here $[\cdot]_i$ denotes the i-th element of the vector).

However, using only Equation (14) without properly safeguarding the learning process can learn a model in which the roots of the equations of motion (the trajectory values) are degenerate, only touching the zero axis and not necessarily crossing it. Such roots present difficulties for forward simulation as they are difficult to find using common numerical methods such as Newton-Rhapson. To prevent such behaviour we propose adding the *degeneracy* term

$$\frac{1}{N}\sum_{k=0}^{N-1}\Big(1 - \frac{1}{1+e^{-0.01(d_k)^m}}\Big),$$
$$d_k = \det\Big(D_2D_1\mathcal{L}_d^{NN}(q_k, q_{k+1})\Big)$$

to the cost function (14), where $m = 1$ or 2 and d corresponds to the Jacobian of the discrete equations of motion (3). This term aims to increase the slope of the equations of motion at the root locations, safeguarding against degenerate roots (which have vanishing gradients at the root location) and rewarding steeper crossings. This term also prevents the learning of constant discrete Lagrangian functions, which satisfy the equations of motion but do not result in meaningful dynamics.

2.1 Incorporation of symmetry into the loss function

Using the proposed loss function we learn a non-degenerate discrete inverse modified Lagrangian, which can then be used to predict motions. However, since discrete Lagrangians are not uniquely determined by the system's motion, the learned discrete Lagrangian for a dynamical system which exhibits symmetries can be arbitrarily unsymmetric even if the learnt discrete Lagrangian minimises the loss function. In the following, we introduce a method which automatically detects symmetries of the dynamical system and drives the learnt discrete Lagrangian towards a symmetric representation. This is useful for system identification as symmetries inform us about conserved quantities which constitute important qualitative features of the dynamical system. Moreover, driving the learnt Lagrangian to a symmetric representation acts as an additional regulariser in the learning process and can be beneficial for numerical simulations of the learned system.

Based on the symmetry condition (11) of Section 1.4, to detect an affine linear symmetry of the dynamical system we propose adding the following term to the cost function:

$$\ell_{\text{sym}} = \frac{1}{N}\sum_{k=0}^{N-1}|(Mq_k + w)^\top\nabla_{q_k}\mathcal{L}_d^{NN}(q_k, q_{k+1}) + (Mq_{k+1} + w)^\top\nabla_{q_{k+1}}\mathcal{L}_d^{NN}(q_k, q_{k+1})|^2,$$

where $q_0, \ldots, q_N$ is a training trajectory. Additionally, the above expression is summed over all training trajectories. The n_q^2 elements in the matrix M and the n_q elements in w are now trainable parameters in addition to the parameters of $\mathcal{L}_d^{NN}$ and are included in the minimisation process. Furthermore, we add the non-triviality condition

$$|\|M\|^2 + \|w\|^2 - 1|^2 = 0 \tag{15}$$

to the loss function to encourage the process to learn a non-trivial symmetry. After learning, Equation (12) is a candidate for a conserved quantity of the discrete system and Equation (10) of the underlying continuous system.

Remark 1. The choice to test the symmetry condition (11) on points (q_k, q_{k+1}) in the training data to construct ℓ_{sym} is arbitrary. In principle, ℓ_{sym} could be any approximation (e.g. a Monte-Carlo integration) of

$$\frac{1}{\text{vol}(\mathcal{M}^\circ)}\int_{\mathcal{M}^\circ}|(Mq_0 + w)^\top\nabla_{q_0}\mathcal{L}_d^{NN}(q_0, q_1) + (Mq_1 + w)^\top\nabla_{q_1}\mathcal{L}_d^{NN}(q_0, q_1)|^2\,\mathrm{d}q_0\,\mathrm{d}q_1$$

where $\mathcal{M}^\circ \subset Q \times Q$ is a (topologically open, pre-compact) subset of the discrete phase space covering all parts of interest.

Remark 2. To learn K functionally independent integrals of motions, we simply add K instances $\ell_{\text{sym}}^{(j)}$ of ℓ_{sym} from Equation (16) to the loss function. Each $\ell_{\text{sym}}^{(j)}$ has trainable parameters $(M^{(j)}, w^{(j)})$. Further, the non-triviality condition (15) is added for each instance of $\ell_{\text{sym}}^{(j)}$. To make sure that all learned $(M^{(k)}, w^{(k)})$ yield functionally independent integrals of motions, we need to make sure that they span a basis of the Lie algebra (13). This is achieved by adding the orthogonality condition

$$\sum_{k=2}^{K}\sum_{s=1}^{k-1}(\text{vec}(M^{(s)})^\top\text{vec}(M^{(k)}) + (w^{(s)})^\top w^{(k)})$$

to the loss function. Here vec writes the columns of a matrix into a single vector for the computation of the Frobenius inner product of two matrices. This corresponds to learning a K-dimensional symmetry group.

Remark 3. The approach can be generalised to arbitrary Lie group actions as considered at the end of Section 1.4 as follows. For $k = 1, \ldots, K$ define $\ell_{\text{sym}}^{(k)}$ as a numerical approximation of

$$\ell_{\text{sym}}^{(k)} \approx \frac{1}{\text{dvol}(\mathcal{M}^\circ)}\int_{\mathcal{M}^\circ}|\hat{v}^{(k)}(\mathcal{L}_d)|^2\,\mathrm{d}q_0\,\mathrm{d}q_1 \tag{16}$$

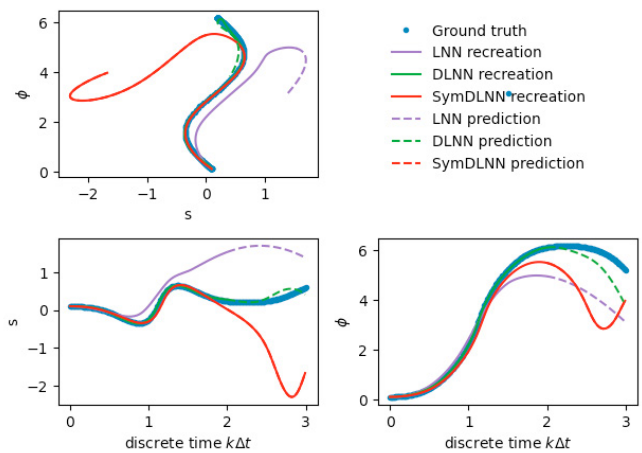

Fig. 2. Pendulum on the cart example: Recreation and prediction of the trajectory based on single trajectory observation.

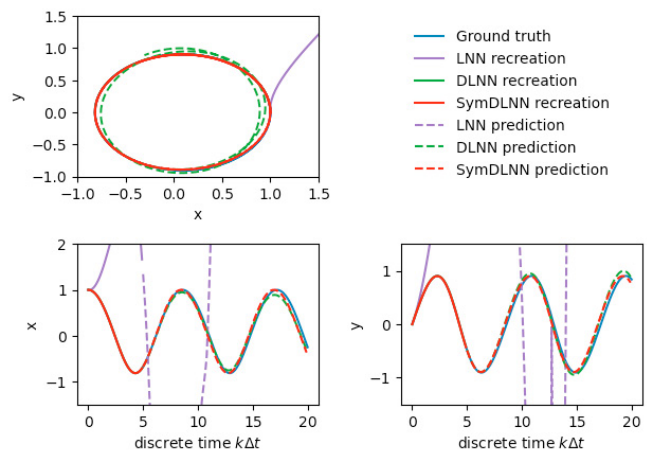

Fig. 3. Kepler example: Recreation and prediction of the trajectory based on single trajectory observation.

Here $\hat{v}^{(k)}$ denotes the invariant vector field to v. The term $\ell_{\text{sym}}^{(k)}$ measures how invariant $\mathcal{L}$ is under actions with group elements of $\exp(tv^{(k)}|t \in \mathbb{R})$. Equip $\mathfrak{g}$ with an inner product $\langle \cdot, \cdot \rangle$ and norm $\| \cdot \|$. Given weights $\alpha^{(k)}, \beta^{(k)} > 0$ define

$$\ell_{\text{sym}}^{\text{total}} = \sum_{k=1}^{K} \left(\ell_{\text{sym}}^{(k)} + \alpha^{(k)} | \|v^{(k)}\| - 1|^2 + \beta^{(k)} \sum_{s=1}^{k-1} \langle v^{(k)}, v^{(s)} \rangle \right)$$

which is added to the loss function. Here $\alpha^{(k)}$ and $\beta^{(k)}$ are fixed, non-negative weights. The last two terms of ℓ_{sym} measure the orthonormality of the spanning set $v^{(1)}, \ldots, v^{(K)}$ while the first term measures how well infinitesimal actions by elements of $\exp(V)$ preserve $\mathcal{L}$.

3. RESULTS

To evaluate the proposed DLNN and SymDLNN methods alongside LNN, each of the methods was applied to two systems: a pendulum on a cart defined by the Lagrangian in Equation (5) with $m_1 = m_2 = l = 1$, and the translational symmetry with $M = [0, 0; 0, 0]$, $w = [1, 0]$ and the Kepler problem in two dimensions with Lagrangian

$$\mathcal{L}_{\text{Kp}}(x, y, \dot{x}, \dot{y}) = \frac{1}{2}(\dot{x}^2 + \dot{y}^2) + \frac{G m_1 m_2}{\sqrt{x^2 + y^2}} \tag{17}$$

for $q = [x, y]^T$, $G = 6.673 \times 10^{-26}$, $m_1 = 6 \times 10^{24}$, $m_2 = 100$ and rotational symmetry of the form $w = [0, 0]$, $M = [0, \sqrt{2}/2; -\sqrt{2}/2, 0]$. For each system, the methods are tasked with learning the underlying dynamics defined by the Lagrangian in three evaluation scenarios: a single trajectory case, a multi-trajectory case, and learning a single trajectory with noisy measurements. After training a variational integrator is used for simulations with each of the three methods in order to assess how well the resulting system model is capable of: recreating the trajectory it was trained on, its ability to predict and extension of this trajectory and ability to preserve the energy and symmetry. Code for all implementations and experiments can be found at https://github.com/yanalish/SymDLNN.

In the first scenario a set of experiments are conducted wherein the observations used to train the neural network are obtained from a single trajectory of N consecutive configuration points spaced at a stepsize of Δt. For the pendulum on a cart $N = 200$ with $\Delta t = 0.01$ was chosen, whereas for the Kepler problem these were set to $N = 50$ and $\Delta t = 0.1$ Trajectory observations were obtained using a variational integrator with a fine grained step size $\Delta t = 10^{-4}$ for the pendulum on a cart and $\Delta t = 10^{-3}$ for the Kepler problem. Using these observations each method attempts to recreate the trajectory with the learnt Lagrangian and predict its extension for an additional $N_{extra} = 100$ steps for the pendulum and $N_{extra} = 150$ for the Kepler problem, the resulting trajectory will be denoted as q^{NN}. For a fair comparison each of LNN, DLNN and SymDLNN methods approximate the continuous and discrete Lagrangian respectively using a 3-layer multi-layer perceptron (MLP) with 128 dimensional hidden layers and SoftPlus activations at each node. The minimisation of the objective functions is performed through gradient descent. Specifically, for all methods the optimisation is performed using an Adam optimiser with an initial learning rate of 3×10^{-3}, other Adam hyperparameters were set to $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1 \times 10^{-8}$. Each network was trained for 100,000 epochs. The only notable difference in the training between the methods is in the SymDLNN method which does not employ the symmetry term for the first 5,000 epochs and then utilises them in the remaining 95,000 epochs. The rationale for this is to initially optimise the parameters of the network to find a suitable initial Lagrangian as the DLNN does and then discover the symmetries.

In Figures 2 and 3 we can see for both example systems that the DLNN and SymDLNN methods better recreate the original train trajectory than the LNN approach. For the pendulum on a cart example DLNN provides the best prediction whereas for Kepler both DLNN and SymDLNN outperform LNN for the unseen extension. For the cart pendulum example the SymDLNN model identified symmetry parameters $M_{cp}^{NN} = [1.51 \times 10^{-3}, -1.26 \times 10^{-3}; -7.83 \times 10^{-6}, -3.62 \times 10^{-5}]$, $w_{cp}^{NN} = [9.97 \times 10^{-1}, 8.04 \times 10^{-2}]^T$ and for the Kepler $M_{Kp}^{NN} = [0.074, 0.813, -0.568, -0.018]$, $w_{Kp}^{NN} = [0.057, 0.076]^T$ from initialization guess $M_{cp}^{guess} = [0.1, 0.1; 0.1, 0.1]$, $w_{cp}^{guess} = [1.5, 0.5]^T$ and $M_{Kp}^{guess} = [0.1, 0.807; -0.607, 0.1]$, $w_{Kp}^{guess} =$

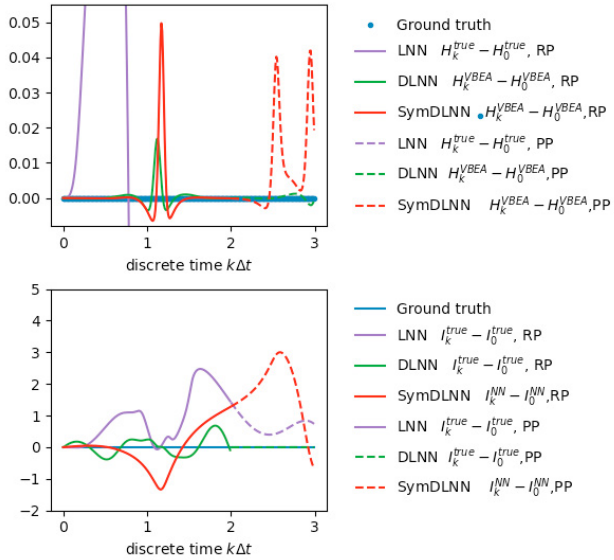

Fig. 4. Pendulum on a cart example: Symmetry and energy error test based on single trajectory observation (RP- recreation phase, PP- prediction phase).

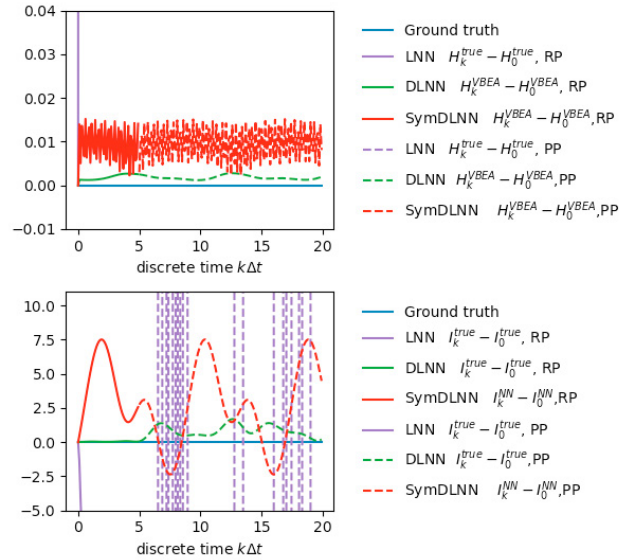

Fig. 5. Kepler example: Symmetry and energy error test based on single trajectory observation (RP- recreation phase, PP- prediction phase).

$[0.1, 0.1]^T$ respectively. We believe this can be further improved with a more rigorous hyperparameter search and more generous computing resources. Figures 4 and 5 demonstrate how accurately the symmetry and energy are preserved with each model. In these plots, we compare how well LNN and DLNN preserve the symmetry with

$$I_k^{true} = (p_k^{NN})^T (M q_k^{NN} + w)$$
$$\text{for } p_k^{NN} = \nabla_{\dot{q}} \mathcal{L}^{true}\left(q_k, \frac{q_{k+1}^{NN} - q_{k-1}^{NN}}{2\Delta t}\right)$$

and how well SymDLLN learns it by computing

$$I_k^{NN} = (p_k^{NN})^T (M^{NN} q_k^{NN} + w^{NN})$$
$$\text{for } p_k^{NN} = -D_1 \mathcal{L}_d^{NN}(q_k^{NN}, q_{k+1}^{NN})$$

For the energy plots we compare how well LNN has learnt the original Hamiltonian and how accurately a Hamiltonian has been approximated based on VBEA using the DLNN and SymDLNN model calculating respectively $H_k^{true}(q_k, \frac{q_{k+1}^{NN} - q_{k-1}^{NN}}{2\Delta t})$ and $H_k^{VBEA}(q_k, \frac{q_{k+1}^{NN} - q_{k-1}^{NN}}{2\Delta t})$. We can clearly see the improvement of using DLNN compared to the LNN approach. The SymDLNN approach performs slightly worse, however, it is important to note that for LNN and DLNN we plot based on our knowledge for the symmetry expression whereas SymDLNN has learnt the symmetry itself, providing a qualitatively better model and additional knowledge of the system's dynamics.

Both examples represent non-chaotic systems, thus learning from a single trajectory learns the behaviour of the system for a restricted portion of the phase space. For this purpose, the same experiments were conducted for all three methods when learning from data from 100 different trajectories of the same lengths and steps as in the single trajectory examples. Recreation and prediction of the trajectory as well as symmetry and energy preservation tests closely resembled the results demonstrated above for learning from a single trajectory.

In our third and last evaluation scenario we examined the performance of our approaches DLNN and SymDLNN

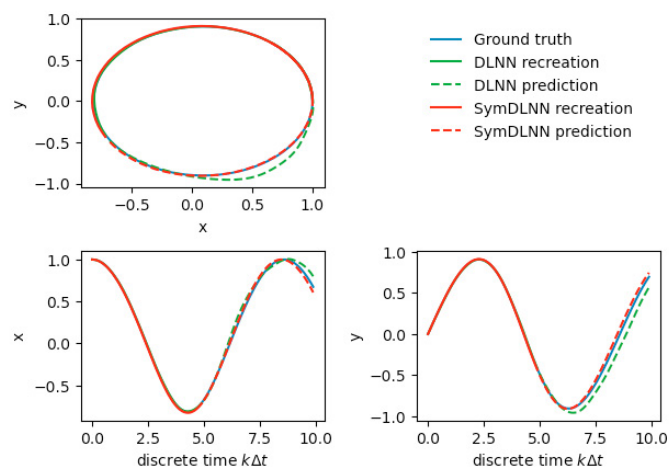

Fig. 6. Kepler example with noise: Recreation and prediction of the trajectory based on observations gathered from a single trajectory.

in the presence of noise. As we can see below for the Kepler problem the proposed model was capable of accurately recreating the observed trajectory and extending it beyond for $N = 50$ and $N_{extra} = 200$ learning from a single trajectory observation for which measurement noise was added with a standard Gaussian distribution with $\sigma^2 = 0.001$. For DLNN, the H^{VBEA} energy and the true symmetry are quite well preserved both during the recreation and the prediction of the trajectory. Despite the noise, SymDLNN is capable of preserving energy and although the symmetry oscillation is slightly larger, it is based on a symmetry which the SymDLNN approach itself identified with values $w^{NN} = [0.056, 0.077]^T, M^{NN} = [0.090, 0.812, -0.568, -0.016]$ with the same initialization as before M_{Kp}^{guess}, w_{Kp}^{guess}. Moreover, we can see that the additional symmetry term helps SymDLNN better extend the trajectory than DLNN.

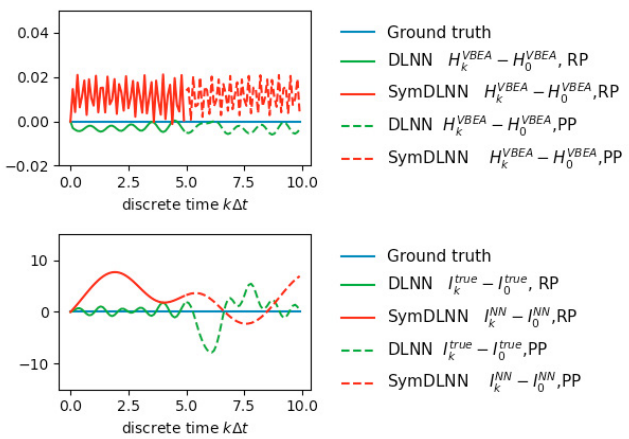

Fig. 7. Kepler example with noise: Symmetry and energy error test based on single trajectory observation (RP- recregation phase, PP- prediction phase).

4. CONCLUSION

We have presented two novel methods, DLNN and SymDLNN, which successfully incorporate the use of variational integrators and variational backward error analysis for learning discrete Lagrangians, utilising the underlying structure of the problem to improve the learned system representation qualitatively and guard against numerical issues in forward simulations. This was achieved through the proposal of a novel regularisation degeneracy term that inductively biases the network to avoid learning Lagrangians whose equations of motion have degenerate roots as well as constant Lagrangians. SymDLNN further extends the DLNN in its ability to automatically identify symmetries during the learning process through a framework that identifies a subgroup which acts by symmetries upon a given Lie group on the configuration space. Numerical experiments demonstrate the qualitative improvements of our proposed methods to previous approaches.

ACKNOWLEDGEMENTS

Y.L. acknowledges funding from the EPSRC Doctoral Training Partnership EP/R513295/1, project reference 2280382. P.S. acknowledges funding by the W.D. Armstrong Fund from the School of Technology at the University of Cambridge. S.R. acknowledges funding by the EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines & Systems EP/L015897/1. C.O. acknowledges funding by the Ministry of Culture and Science of the State of North Rhine-Westphalia.

REFERENCES

Allen-Blanchette, C., Veer, S., Majumdar, A., and Leonard, N.E. (2020). LagNetViP: A Lagrangian neural network for video prediction. *arXiv preprint, arXiv:2010.12932*.

Aoshima, T., Matsubara, T., and Yaguchi, T. (2021). Deep discrete-time lagrangian mechanics. In *ICLR2021 Workshop on Deep Learning for Simulation (SimDL)*, volume 5.

Chen, Z., Zhang, J., Arjovsky, M., and Bottou, L. (2019). Symplectic recurrent neural networks. *arXiv preprint arXiv:1909.13334*.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., and Ho, S. (2020). Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*.

Dehmamy, N., Walters, R., Liu, Y., Wang, D., and Yu, R. (2021). Automatic symmetry discovery with lie algebra convolutional network. *Advances in Neural Information Processing Systems*, 34, 2503–2515.

Dierkes, E., Offen, C., Ober-Blöbaum, S., and Flaßkamp, K. (2022). Learning Hamiltonian systems and symmetries. *arXiv preprint, arXiv:2301.07928*.

Greydanus, S., Dzamba, M., and Yosinski, J. (2019). Hamiltonian neural networks. *Advances in neural information processing systems*, 32.

Hairer, E., Lubich, C., and Wanner, G. (2006). *Geometric numerical integration. Structure-preserving algorithms for ordinary differential equations*, volume 31. vol. 31. Springer Science & Business Media, 2 edition.

Havens, A. and Chowdhary, G. (2021). Forced variational integrator networks for prediction and control of mechanical systems. In *Learning for Dynamics and Control*, 1142–1153. PMLR.

Liberzon, D. (2011). *Calculus of variations and optimal control theory: a concise introduction*. Princeton university press.

Marsden, J.E. and Ratiu, T.S. (1999). *Introduction to mechanics and symmetry*, volume 17 of *Texts in Applied Mathematics*. Springer, 2nd edition.

Marsden, J.E. and West, M. (2001). Discrete mechanics and variational integrators. *Acta Numerica*, 10(1), 357–514.

Ober-Blöbaum, S. and Offen, C. (2023). Variational learning of Euler–Lagrange dynamics from data. *Journal of Computational and Applied Mathematics*, 421, 114780. doi:10.1016/j.cam.2022.114780.

Offen, C. and Ober-Blöbaum, S. (2022). Symplectic integration of learned Hamiltonian systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(1), 013122. doi:10.1063/5.0065913.

Offen, C. and Ober-Blöbaum, S. (2023). Learning discrete Lagrangians for variational PDEs from data and detection of travelling waves. *arXiv preprint, arXiv:2302.08232*.

Qin, H. (2020). Machine learning and serving of discrete field theories. *Scientific Reports*, 10(1), 1–15.

Saemundsson, S., Terenin, A., Hofmann, K., and Deisenroth, M. (2020). Variational integrator networks for physically structured embeddings. In *International Conference on Artificial Intelligence and Statistics*, 3078–3087. PMLR.

Santos, S., Ekal, M., and Ventura, R. (2022). Symplectic momentum neural networks-using discrete variational mechanics as a prior in deep learning. In *Learning for Dynamics and Control Conference*, 584–595. PMLR.

Vermeeren, M. (2017). Modified equations for variational integrators. *Numerische Mathematik*, 137(4), 1001–1037.