Combining Physics and Deep Learning to learn Continuous-Time Dynamics Models

Michael Lutter[1] and Jan Peters[1]

Abstract

Deep learning has been widely used within learning algorithms for robotics. One disadvantage of deep networks is that these networks are black-box representations. Therefore, the learned approximations ignore the existing knowledge of physics or robotics. Especially for learning dynamics models, these black-box models are not desirable as the underlying principles are well understood and the standard deep networks can learn dynamics that violate these principles. To learn dynamics models with deep networks that guarantee physically plausible dynamics, we introduce physics-inspired deep networks that combine first principles from physics with deep learning. We incorporate Lagrangian mechanics within the model learning such that all approximated models adhere to the laws of physics and conserve energy. Deep Lagrangian Networks (DeLaN) parametrize the system energy using two networks. The parameters are obtained by minimizing the squared residual of the Euler-Lagrange differential equation. Therefore, the resulting model does not require specific knowledge of the individual system, is interpretable, and can be used as a forward, inverse, and energy model. Previously these properties were only obtained when using system identification techniques that require knowledge of the kinematic structure. We apply DeLaN to learning dynamics models and apply these models to control simulated and physical rigid body systems. The results show that the proposed approach obtains dynamics models that can be applied to physical systems for real-time control. Compared to standard deep networks, the physics-inspired models learn better models and capture the underlying structure of the dynamics.

1 Introduction

During the last five years, deep learning has shown the potential to fundamentally change the use of learning in robotics. Currently, many robot learning approaches involve a deep network as part of their algorithm. The network either represents a policy that selects the actions, a dynamics model that predicts the next state, or a state estimator that extracts the relevant features from unstructured observations. Initially, many of these approaches were only applicable to simulated environments due to the large amounts of data required to train the networks. When using massive parallelized simulations, these methods achieved astonishing results (Heess et al. 2017). By now, these learning algorithms have been improved and start to be applied to real-world systems (Akkaya et al. 2019; Haarnoja et al. 2018). On the physical systems, the deep network approaches have not bypassed classical robotics techniques yet, but have shown very promising results achieving comparable results as classical methods.

Within many proposed algorithms deep networks have replaced analytic models and other function approximators due to their simplicity, generic applicability, scalability, high model capacity and widespread availability of GPU's enabling fast training and evaluation. The generic applicability of these black-box models combined with the high model capacity is a curse and blessing. On the one side, this combination enables the learning of arbitrary functions with high fidelity. However, this combination is also susceptible to overfit to the data without retrieving the underlying structure. Furthermore, the black-box nature of standard deep networks prevents including prior knowledge from first-order principles. This limitation is especially problematic for robotics as the overfitting to spurious data can lead to unpredictable behaviors damaging the physical system. The problem is also made unnecessarily harder as all existing knowledge of robotics and mechanics is ignored.

In this article, we propose a new approach that combines existing knowledge with deep networks. This combination enables to learn better representations for robotics and retains the advantages of deep networks. To learn physically plausible continuous-time dynamics models of rigid body systems, we combine Lagrangian mechanics with deep networks. The proposed Deep Lagrangian Networks (DeLaN) use two deep networks to parameterize the kinetic and potential energy (Lutter et al. 2019). The network parameters are learned by minimizing the squared residual of the Euler-Lagrange differential equation. The resulting dynamics models are guaranteed to evolve as a mechanical system and conserve the system energy. Therefore, these models achieve better long-term predictions and control performance than the standard black-box models. The resulting physics-inspired models share many of the characteristics of analytic models without requiring specific knowledge about the individual system. For example, DeLaN models are interpretable and enable the computation of the gravitational forces, the momentum, and the system energy. Previously, computing this decomposition was

[1]Intelligent Autonomous Systems Group at Technical University of Darmstadt, Germany

Corresponding author:
Michael Lutter, michael@robot-learning.de

only possible using the analytic models with the system parameters. DeLaN also enables the computation of the forward and inverse models with the same parameters. These characteristics are in stark contrast to standard black-box models. Such black-box models only obtain either the forward or the inverse model and cannot compute the different physical quantities as these need to be learned unsupervised. Due to these advantages of physics-inspired dynamics models, many variants have been proposed (Greydanus et al. 2019; Gupta et al. 2019; Zhong et al. 2019; Saemundsson et al. 2020; Cranmer et al. 2020).

1.1 Contribution

The contribution of this article is the presentation of a model learning framework that combines the existing knowledge of mechanics with deep networks. To highlight the possibilities of this approach for learning dynamics model, we describe Deep Lagrangian Networks (DeLaN) (Lutter et al. 2019). This model learning approach combines deep learning with Lagrangian mechanics to learn a physically plausible model by minimizing the residual of the Euler-Lagrange ordinary differential equation. In contrast to our previous papers (Lutter et al. 2019; Lutter and Peters 2019), which mainly focused on specific algorithmic ideas, this article

(1) consolidates the existing literature on physics-inspired model learning which has been introduced since the initial presentation of DeLaN. We summarize the individual contributions and merge the variants into a single big picture.

(2) extends the previous experimental evaluation and provides in-depths comparisons of the different variants of physics-inspired networks. We evaluate the control performance of the learned models on the physical system using inverse dynamics control and energy control. In addition, the performance is compared to system identification and black-box model learning.

(3) provides an elaborate discussion on the current shortcomings of physics-inspired networks and highlight possibilities to overcome these limitations.

1.2 Outline

To provide a self-contained overview about physics-inspired deep networks for learning dynamics models, we briefly summarize the related work (Section 2), prior approaches for learning dynamics models of rigid body systems as well as the basics of Lagrangian and Hamiltonian mechanics (Section 3.2). Subsequently, we introduce physics-inspired networks derived from Lagrangian and Hamiltonian mechanics as well as the existing variants (Section 4). Section 5 presents the experimental results of applying these models to model-based control and compares the performance to system identification as well as deep network dynamics models. Finally, Section 6 discusses the experimental results, highlights the limitations of physics-inspired networks, and summarizes the contributions of this article.

2 Related Work

In the main part of this article, we focus on learning continuous-time dynamics models of mechanical systems. However, physics-inspired networks and continuous-time deep networks have been utilized for different applications areas. In this section, we want to briefly summarize the existing work on both topics outside the domain of rigid body systems and their differences.

2.1 Physics-Inspired Deep Networks

Incorporating knowledge of physics within deep networks has been approached by introducing conservation laws or symmetries within the network architecture. Both approaches are tightly coupled due to Noether's theorem showing that symmetries induce conservation laws. In the case of conservation laws, these laws can be incorporated by minimizing the residual of the corresponding differential equation to obtain the optimal network parameters. The combination of deep learning and differential equations has been well known for a long time and investigated in more abstract forms (Lee and Kang 1990; Meade Jr and Fernandez 1994; Lagaris et al. 1998, 2000). Using this approach, various authors proposed to use the Navier-Stokes equation (Raissi et al. 2017a; Chu et al. 2021), Schroedinger equation (Raissi et al. 2017b), Burgers Equation (Holl et al. 2020), Hamilton's equation (Greydanus et al. 2019; Zhong et al. 2019; Chen et al. 2020; Toth et al. 2019) or the Euler-Lagrange equation (Lutter et al. 2019; Qin 2020; Cranmer et al. 2020; Gupta et al. 2019).

Symmetries can be integrated within the network architecture by selecting a non-linear transformation that is either equivariant, i.e., preserves the symmetry, or is invariant to specific transformations of the input. Using this approach, one can derive layers that are translational-, rotational-, scale- and gauge equivariant (Cohen and Welling 2016; Bekkers 2019; Wang et al. 2020b; Cohen et al. 2019). These architectures are frequently used for computer vision as image classification is translational and rotational invariant (Cohen et al. 2019; Weiler and Cesa 2019; Lenc and Vedaldi 2015). Up to now, only very few papers have applied this approach to model physical phenomena (Wang et al. 2020b; Anderson et al. 2019). A different approach to symmetries was proposed by Huh et al. (2020). To obtain time translation invariance, which is equivalent to conservation of energy, this work optimized time-reversibility. Therefore, the symmetry is not incorporated in the network architecture but the optimization loss.

Besides these generic approaches utilizing symmetries and conservation laws, various authors also proposed specific architectures for individual problems. In this case, the known spatial structure of the problem is embedded within the network architecture. For example, Wang et al. (2020a) proposed a network architecture for turbulent flow predictions that incorporates multiple spatial and temporal scales. Sanchez-Gonzalez et al. (2018) used a graph network to encode the known kinematic structure and the local interactions between two links within the network structure. Similarly, Schütt et al. (2017) incorporates the local structure of molecules within the network architecture.

2.2 Continuous-Time Models & Neural ODEs

The work on neural ordinary differential equation (ODE) by Chen et al. (2018) initiated a large surge of research on continuous-time models. The original work on neural ODE proposed a deep network with infinite depth to improve classification and density estimation. While these algorithms were not meant for modeling dynamical systems, the explicit integration step within the neural ODE led to rediscover continuous-time models for dynamical systems. Since then, neural ode's have been frequently mentioned as inspiration to learn continuous-time models (Saemundsson et al. 2020; Huh et al. 2020; Botev et al. 2021; Hochlehnert et al. 2021). Frequently the term neural ODE is used interchangeably for a continuous-time model with a deep network. In this work, we will only use the term continuous-time model. One technical difference between the original neural ODE and continuous-time models is that the neural ODE uses a variable time step integrator, most commonly the Dormand–Prince method. The continuous-time models use a fixed time step integrator. For the fixed time step integrator, different authors have used the explicit Euler, the Runge Kutta method, or symplectic integrators. For dynamics models, the fixed time step is convenient as the data is observed at a fixed time step determined by the sampling rate of digital sensors.

3 Preliminaries

We want to introduce the standard model learning techniques for dynamical systems and briefly summarize the relevant theory of Lagrangian and Hamiltonian Mechanics.

3.1 Learning Dynamics Models

Models describing system dynamics, i.e. the coupling of the system input $\boldsymbol{u}$ and system state $\boldsymbol{x}$, are essential for model-based control approaches (Ioannou and Sun 1996) and model based planning. Depending on the approach, one either relies on the forward or inverse model. For example, inverse dynamics control (de Wit et al. 2012) uses the inverse model to compensate system dynamics, while model-predictive control (Camacho and Alba 2013) and optimal control (Zhou et al. 1996) use the forward model to compute the future states given an action sequence. For discrete time models, the forward model f maps from the system state $\boldsymbol{x}_t$ and input $\boldsymbol{u}_t$ to the next state $\boldsymbol{x}_{t+1}$. The inverse model f^{-1} maps from system state and the next state to the system input. Mathematically this is described by

$$f(\boldsymbol{x}_t, \boldsymbol{u}_t;\, \boldsymbol{\theta}) = \boldsymbol{x}_{t+1}, \qquad f^{-1}(\boldsymbol{x}_t, \boldsymbol{x}_{t+1};\, \boldsymbol{\theta}) = \boldsymbol{u}_t, \tag{1}$$

with the model parameters $\boldsymbol{\theta}$. In the continuous time setting the next state $\boldsymbol{x}_{t+1}$ is replaced with the change of the state $\dot{\boldsymbol{x}}$, i.e.,

$$f(\boldsymbol{x}_t, \boldsymbol{u}_t;\, \boldsymbol{\theta}) = \dot{\boldsymbol{x}}_t, \qquad f^{-1}(\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t;\, \boldsymbol{\theta}) = \boldsymbol{u}_t. \tag{2}$$

The continuous-time system can be combined with an integrator, e.g., explicit Euler, Runge-Kutta method, or symplectic integrators, to predict the next state instead of the change of the system state. Therefore, the continuous-time model is independent of the time discretization. Depending on the chosen representation, the transfer function f and parameters $\boldsymbol{\theta}$ are obtained using different approaches. In the following, we will differentiate the different approaches, (1) model engineering, (2) data-driven system identification, and (3) black-box model learning. In Section 4, we will extend these existing categories to physics-inspired models.

3.1.1 ***Model Engineering.***

The most classical approach is model engineering, which is predominantly used within the industry. In this case, the transfer function f is the equations of motion and the model parameters are the physical parameters of the robot consisting of the masses, center of gravity, length, and inertia. The equations of motion have to be manually derived for each system. Frequently, one assumes perfect rigid bodies connected by ideal joints and uses Newtonian, Lagrangian or Hamiltonian mechanics and the known structure of the systems to derive the equations. The model parameters can be either inferred using the CAD software or measured by disassembling the individual system. The latter is more precise as it incorporates the deviations due to the manufacturing process (Albu-Schäffer 2002). Furthermore, the parameters are identical for the forward and inverse model. Therefore, this approach yields the forward and inverse model simultaneously. To summarize, this approach can yield very precise forward and inverse models for rigid body systems but is labor-intensive as the parameters need to be manually inferred.

3.1.2 ***Data-Driven System Identification.***

Similar to model engineering, data-driven system identification uses the analytic equations of motions as the transfer function. However, the model parameters are learned from observed data rather than measured. Therefore, the equations of motions need to be manually derived but the model parameters are learned. In 1985 four different groups showed concurrently that the dynamics parameters can be obtained by linear regression using hand-designed features for rigid body kinematic chains (Khosla and Kanade 1985; Mukerjee and Ballard 1985; Atkeson et al. 1986; Gautier 1986). This approach is commonly referenced as the standard system identification technique for robot manipulators by the textbooks (Siciliano and Khatib 2016). However, this approach cannot guarantee physically plausible parameters as the dynamics parameters have additional constraints. For example, this approach can yield negative masses, an inertia matrix that is not positive definite, or violate the parallel axis theorem (Ting et al. 2006). The disadvantages of this approach are that one can only infer linear combinations of the dynamics parameters, cannot apply it to close-loop kinematics (Siciliano and Khatib 2016) and can only be applied inverse dynamics. The inverse dynamic formulation is problematic as the inverse dynamics do not necessarily have a unique solution due to friction (Ratliff et al. 2016). To overcome these shortcomings, Ting et al. (2006) proposed a projection-based approach while many others (Traversaro et al. 2016; Wensing et al. 2017; Ledezma and Haddadin 2017; Sutanto et al. 2020; Lutter et al. 2020, 2021b; Geist and Trimpe 2021) used virtual parametrizations that guarantee physical plausible parameters. For the latter, the optimization does not simplify to linear regression but can be solved using gradient descent. To summarize, this approach only requires the equations of motions analytically and can learn the dynamical parameters from data. Therefore, this approach is

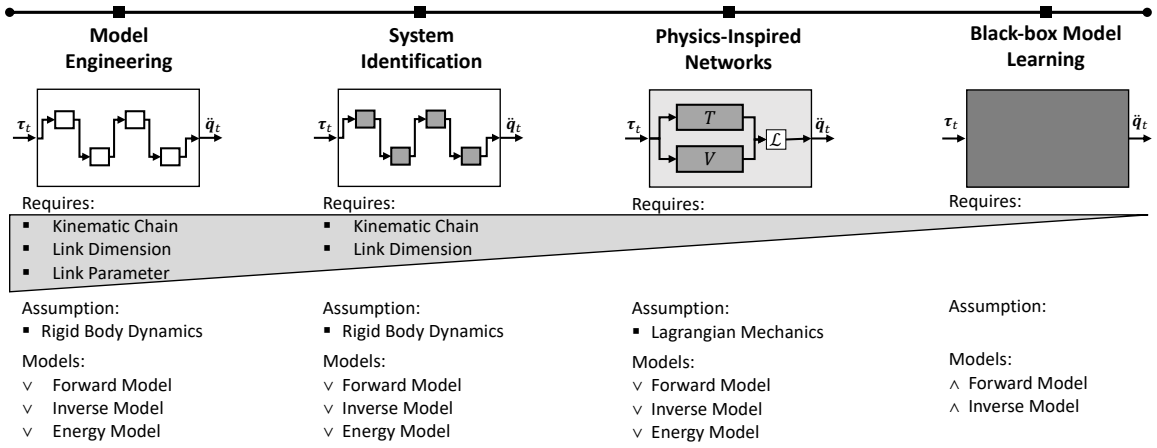

Figure 1. The requirements and assumptions of the different approaches to obtain the dynamics model of mechanical systems. The physics-inspired networks bridge the gap between classical system identification and black-box model learning. While system identification requires knowledge of the kinematic chain, the physics-inspired networks do not require any knowledge of the specific system but obtain comparable characteristics as system identification. Physics-inspired networks also guarantee energy-conserving models and obtain the forward, inverse, and energy model simultaneously.

not as labor-intensive as model engineering but one needs to ensure to collect 'good' data for the learning.

3.1.3 ***Black-Box Model Learning.*** While the previous approaches required knowledge about the individual kinematic chain to derive the equations of motion, the black-box approaches do not require any knowledge of the system. These approaches use any black-box function approximator as a transfer function and optimize the model parameters to fit the observed data. For example, the existing literature used Local Linear Models (Schaal et al. 2002; Haruno et al. 2001), Gaussian Mixture Models (Calinon et al. 2010; Khansari-Zadeh and Billard 2011), Gaussian Processes (Kocijan et al. 2004; Nguyen-Tuong et al. 2009; Nguyen-Tuong and Peters 2010; Romeres et al. 2019, 2016; Camoriano et al. 2016), Support Vector Machines (Choi et al. 2007; Ferreira et al. 2007), feedforward- (Jansen 1994; Lenz et al. 2015; Ledezma and Haddadin 2017; Sanchez-Gonzalez et al. 2018) or recurrent neural networks (Rueckert et al. 2017; Ha and Schmidhuber 2018; Hafner et al. 2019a,b) to learn the dynamics model. The black-box models obtain either the forward or inverse model and the learned model is only valid on the training data distribution. The previous methods based on the analytic equations of motions obtained both models simultaneously and generalize beyond the data distribution as the learned physical parameters are globally valid. However, the black-box models do not require assumptions about the systems and can learn systems including contacts. These previous approaches relied on assuming rigid body dynamics and could only learn the system dynamics of articulated bodies using reduced coordinates without contacts. Therefore, black-box models can be more accurate for real-world systems where the underlying assumption is not valid but is limited to the training domain and rarely extrapolate.

3.2 Lagrangian Mechanics

One approach to derive equations of motion is Lagrangian Mechanics. In the following, we summarize this approach as we will use it in Section 4 to propose a physics-inspired network for learning dynamics models. More specifically we use the Euler-Lagrange formulation with non-conservative forces and generalized coordinates. For more information and the formulation using Cartesian coordinates please refer to the textbooks (Greenwood 2006; de Wit et al. 2012; Featherstone 2007). Generalized coordinates $\boldsymbol{q}$ are coordinates that uniquely define the system configuration without constraints. These coordinates are often called reduced coordinates. For articulated bodies, the system state can be expressed as $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]$. The Lagrangian mechanic's formalism defines the Lagrangian $\mathcal{L}$ as a function of generalized coordinates $\boldsymbol{q}$ describing the complete dynamics of a given system. The Lagrangian is not unique and every $\mathcal{L}$ which yields the correct equations of motion is valid. The Lagrangian is generally chosen to be

$$\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = T(\boldsymbol{q}, \dot{\boldsymbol{q}}) - V(\boldsymbol{q}) = \frac{1}{2}\dot{\boldsymbol{q}}^\top \mathbf{H}(\boldsymbol{q})\dot{\boldsymbol{q}} - V(\boldsymbol{q}), \tag{3}$$

with the kinetic energy T, the potential energy V and the mass matrix $\mathbf{H}(\boldsymbol{q})$. The kinetic energy T is quadratic for any choice of generalized coordinates and any non-relativistic system. The mass matrix is the symmetric and positive definite (de Wit et al. 2012). The positive definiteness ensures that all non-zero velocities lead to positive kinetic energy. Applying the calculus of variations yields the Euler-Lagrange equation with non-conservative forces described by

$$\frac{d}{dt}\frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \dot{\boldsymbol{q}}} - \frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \boldsymbol{q}} = \boldsymbol{\tau}, \tag{4}$$

$$\frac{\partial^2 \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial^2 \dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \boldsymbol{q}\partial \dot{\boldsymbol{q}}}\dot{\boldsymbol{q}} - \frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \boldsymbol{q}} = \boldsymbol{\tau}, \tag{5}$$

where $\boldsymbol{\tau}$ are generalized forces frequently corresponding to the system input $\boldsymbol{u}$. Substituting $\mathcal{L}$ with the kinetic and potential energy into Equation (4) yields the second order ODE described by

$$\mathbf{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \underbrace{\dot{\mathbf{H}}(\boldsymbol{q})\dot{\boldsymbol{q}} - \frac{1}{2}\left(\dot{\boldsymbol{q}}^\top \frac{\partial \mathbf{H}(\boldsymbol{q})}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}}\right)^\top}_{=\mathbf{c}(\boldsymbol{q},\dot{\boldsymbol{q}})} + \underbrace{\frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}}}_{=\mathbf{g}(\boldsymbol{q})} = \boldsymbol{\tau}, \tag{6}$$

where $\mathbf{c}$ describes the forces generated by the Centripetal and Coriolis forces and $\boldsymbol{g}$ the gravitational forces (Featherstone 2007). Most robotics textbooks abbreviate this equation as $\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}$. Using this ODE, any multi-particle mechanical system with holonomic constraints can be described. Various authors used this ODE to manually derive the equations of motion for coupled pendulums (Greenwood 2006), robotic manipulators with flexible joints (Book 1984; Spong 1987), parallel robots (Miller 1992; Geng et al. 1992; Liu et al. 1993) or legged robots (Hemami and Wyman 1979; Golliday and Hemami 1977).

3.3 Hamiltonian Mechanics

A different approach to deriving the equations of motions is Hamiltonian mechanics. In this case, the system dynamics are described using the state $\boldsymbol{x} = [\boldsymbol{q}, \boldsymbol{p}]$ with generalized momentum $\boldsymbol{p}$ instead of the generalized velocity $\dot{\boldsymbol{q}}$ and the Hamiltonian $\mathcal{H}$ instead of the Lagrangian. The generalized momentum can be expressed using the Lagrangian and is described by $\boldsymbol{p} = \partial\mathcal{L}/\partial\dot{\boldsymbol{q}}$ (Fitzpatrick 2008). Given the parametrization of the Lagrangian (Equation (3)), this definition is equivalent to $\boldsymbol{p} = \boldsymbol{H}(\boldsymbol{q})\,\dot{\boldsymbol{q}}$. The Hamiltonian describes the complete energy of the system and is defined as

$$\mathcal{H}(\boldsymbol{q}, \boldsymbol{p}) = T(\boldsymbol{q}, \boldsymbol{p}) + V(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{p}^\top \boldsymbol{H}(\boldsymbol{q})^{-1}\boldsymbol{p} + V(\boldsymbol{q}). \quad (7)$$

The Hamiltonian can be computed by applying the Legendre transformation to the Lagrangian which is described by

$$\mathcal{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \dot{\boldsymbol{q}}^\top \frac{\partial\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial\dot{\boldsymbol{q}}} - \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}). \quad (8)$$

Using the generalized momentum $\boldsymbol{p}$ and the generalized coordinate $\boldsymbol{q}$, the Euler-Lagrange equation can be rewritten to yield Hamilton's equations with control (Greenwood 2006). Hamilton's equations is described by

$$\dot{\boldsymbol{q}} = \frac{\partial\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial\boldsymbol{p}}, \qquad \dot{\boldsymbol{p}} = -\frac{\partial\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial\boldsymbol{q}} + \boldsymbol{\tau}. \quad (9)$$

The Euler-Lagrange equation (Equation (6)) can be easily derived from Hamilton's equation by substituting Equation (7) into Equation (9) and using the definition of the generalized momentum, i.e.,

$$\begin{aligned}
\dot{\boldsymbol{p}} &= -\frac{\partial\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial\boldsymbol{q}} + \boldsymbol{\tau}, \\
\frac{d}{dt}\left[\boldsymbol{H}(\boldsymbol{q})\,\dot{\boldsymbol{q}}\right] &= \frac{1}{2}\left(\boldsymbol{p}^\top \boldsymbol{H}^{-1}\frac{\partial\boldsymbol{H}}{\partial\boldsymbol{q}}\boldsymbol{H}^{-1}\boldsymbol{p}\right)^\top - \frac{\partial V}{\partial\boldsymbol{q}} + \boldsymbol{\tau}, \\
\boldsymbol{H}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} &= \frac{1}{2}\left(\dot{\boldsymbol{q}}^\top \frac{\partial\boldsymbol{H}}{\partial\boldsymbol{q}}\dot{\boldsymbol{q}}\right)^\top - \frac{\partial V}{\partial\boldsymbol{q}} + \boldsymbol{\tau}, \\
\boldsymbol{H}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} - \frac{1}{2}\left(\dot{\boldsymbol{q}}^\top \frac{\partial\boldsymbol{H}}{\partial\boldsymbol{q}}\dot{\boldsymbol{q}}\right)^\top &+ \frac{\partial V}{\partial\boldsymbol{q}} = \boldsymbol{\tau}.
\end{aligned}$$

Many textbooks omit the generalized forces within Hamilton's equation but adding these generalized forces is straightforward as shown in the previous derivation.

4 Physics-Inspired Deep Networks

A different approach to black-box model learning is to combine black-box models with physics to guarantee a physically plausible dynamics model. One combination is to use deep networks to represent the system energy and use the resulting Lagrangian to derive the equations of motion using the Euler-Lagrange differential equation. This approach was initially proposed by Lutter et al. (2019) with the presentation of Deep Lagrangian Networks (DeLaN). Since then, many papers exploring variations of these approaches have been proposed including approaches that use Hamiltonian mechanics instead of Lagrangian mechanics (Greydanus et al. 2019; Cranmer et al. 2020; Gupta et al. 2019; Zhong et al. 2019; Sanchez-Gonzalez et al. 2019; Saemundsson et al. 2020; Zhong et al. 2021, 2020; Hochlehnert et al. 2021).

All of these models have in common, that the learned dynamics models conserve energy when the non-conservative forces can be modeled properly and the generalized coordinates are observed. Therefore, the learned model is guaranteed to adhere to one of the fundamental concepts of physics. This property is beneficial as it has been shown within prior research that naive deep network dynamics models frequently increase or decrease the system energy even when the energy should be conserved (Greydanus et al. 2019; Zhong et al. 2019; Hochlehnert et al. 2021; Saemundsson et al. 2020).

In the following, we present DeLaN (Section 4.1) and the combination of Hamiltonian mechanics and deep networks in Section 4.2. Afterwards, Section 4.3 describes all the proposed extensions of DeLaN and Hamiltonian Neural Networks (HNN). Therefore, this section provides the big picture of existing physics-inspired deep networks for learning dynamics models. The flow charts of the variants are shown in Figure 2

4.1 Deep Lagrangian Networks (DeLaN)

DeLaN is one instantiation of these physics-inspired deep networks. DeLaN parametrizes the mass matrix $\boldsymbol{H}$ and the potential energy V as two separate deep networks. Therefore, the approximate Lagrangian $\mathcal{L}$ described by

$$\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}};\, \psi, \phi) = \frac{1}{2}\dot{\boldsymbol{q}}^\top \boldsymbol{H}(\boldsymbol{q};\, \psi)\,\dot{\boldsymbol{q}} - V(\boldsymbol{q};\, \phi). \quad (10)$$

Using this parametrization the forward and inverse model can be derived. The forward model $\ddot{\boldsymbol{q}} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau};\, \psi, \phi)$ is described by

$$\begin{aligned}
\ddot{\boldsymbol{q}} &= \left[\frac{\partial^2\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial^2\dot{\boldsymbol{q}}}\right]^{-1}\left[\boldsymbol{\tau} - \frac{\partial\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial\boldsymbol{q}\partial\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}} + \frac{\partial\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial\boldsymbol{q}}\right], \\
&= \boldsymbol{H}^{-1}\left[\boldsymbol{\tau} - \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} + \frac{1}{2}\left(\dot{\boldsymbol{q}}^\top \frac{\partial\mathbf{H}}{\partial\boldsymbol{q}}\dot{\boldsymbol{q}}\right)^\top - \frac{\partial V}{\partial\boldsymbol{q}}\right].
\end{aligned}$$

The inverse model $\boldsymbol{\tau} = f^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}};\, \psi, \phi)$ is described by

$$\begin{aligned}
\boldsymbol{\tau} &= \frac{\partial^2\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial^2\dot{\boldsymbol{q}}}\ddot{\boldsymbol{q}} + \frac{\partial\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial\boldsymbol{q}\partial\dot{\boldsymbol{q}}}\dot{\boldsymbol{q}} - \frac{\partial\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial\boldsymbol{q}}, \\
&= \boldsymbol{H}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{H}}\dot{\boldsymbol{q}} - \frac{1}{2}\left(\dot{\boldsymbol{q}}^\top \frac{\partial\mathbf{H}}{\partial\boldsymbol{q}}\dot{\boldsymbol{q}}\right)^\top + \frac{\partial V}{\partial\boldsymbol{q}}.
\end{aligned} \quad (11)$$

The partial derivatives within the forward and inverse model can be computed using automatic differentiation or symbolic

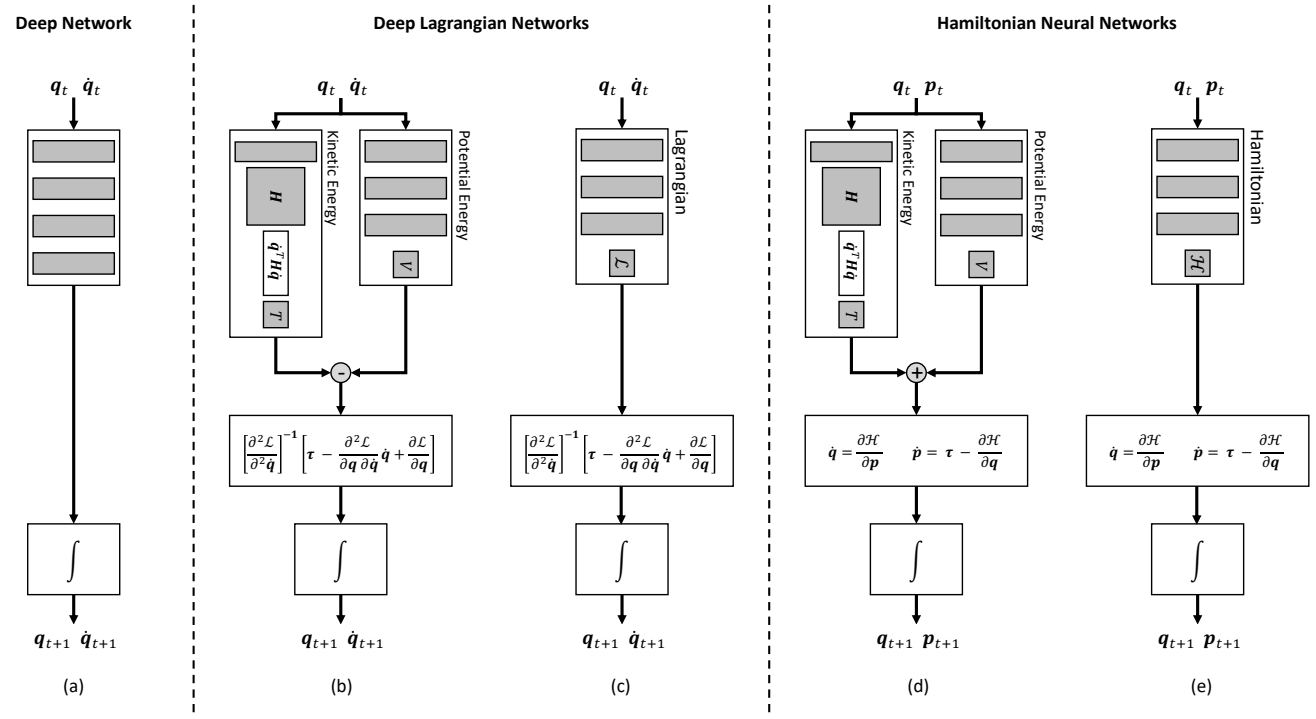

Figure 2. The flowcharts of a continuous-time forward model using a deep network and the physics-inspired networks forward models. (a) Standard deep model learning approach, where a network is used to directly predict the change in position and velocity. (b-c) Deep Lagrangian Networks which use deep networks to predict the Lagrangian $\mathcal{L}$ of the dynamical system and compute the change in position and velocity using the Euler-Lagrange differential equation. (b) shows the structured Lagrangian approach, where two networks predict the kinetic T and potential energy V and computes the Lagrangian is analytically using $\mathcal{L} = T - V$. (c) shows the black-box Lagrangian approach where a network directly predicts $\mathcal{L}$. (d-e) Hamiltonian Neural networks which use deep networks to predict the Hamiltonian $\mathcal{H}$ and computes the change in position and impulse via Hamilton's equation. Similar to the Lagrangian variants, (d) shows the structured Hamiltonian and (e) the black-box Hamiltonian. The structured Hamiltonian computes the Hamiltonian via $\mathcal{H} = T(\boldsymbol{q}, \dot{\boldsymbol{q}}) - V(\boldsymbol{q})$, where the kinectic and potential energy is predicted using two networks. The black-box Hamiltonian uses a single network to directly predict $\mathcal{H}$.

differentiation. See Lutter et al. (2019) for the symbolic differentiation of the mass matrix and the deep networks.

The system energy cannot be learned using supervised learning as the system energy cannot be observed. Therefore, the network weights of the kinetic and potential energy are learned unsupervised using the temporal consequences of the actions and system energy. One approach to learn the network parameters is to minimize the residual of the Euler-Lagrange differential equation. This optimization problem is described by

$$\psi^*, \phi^* = \arg\min_{\psi,\phi} \left\| \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}} - \frac{\partial \mathcal{L}}{\partial \boldsymbol{q}_i} - \boldsymbol{\tau} \right\|^2_{\boldsymbol{W}_\tau}, \tag{12}$$

with the Mahalanobis norm $\|\cdot\|_{\boldsymbol{W}}$ and the diagonal covariance matrix of the generalized forces $\boldsymbol{W}_\tau$. It is beneficial to normalize the loss using the covariance matrix because magnitude of the residual might vary between different joints. This optimization can be solved using any gradient-based optimization technique. Minimizing the squared residual is equivalent to fitting the inverse model, i.e., $\psi^*, \phi^* = \arg\min_{\psi,\phi} \left\| \boldsymbol{\tau} - f^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}};\ \psi, \phi) \right\|^2_{\boldsymbol{W}_\tau}$. This loss can also be extended to include the forward prediction that fits the joint accelerations. The combined optimization problem is described by

$$\psi^*, \phi^* = \arg\min_{\psi,\phi} \left\| \boldsymbol{\tau} - f^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}};\ \psi, \phi) \right\|^2_{\boldsymbol{W}_\tau} + \left\| \ddot{\boldsymbol{q}} - f(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau};\ \psi, \phi) \right\|^2_{\boldsymbol{W}_{\ddot{q}}}, \tag{13}$$

with the diagonal covariance matrix of the generalized forces $\boldsymbol{W}_\tau$ and accelerations $\boldsymbol{W}_{\ddot{q}}$. Furthermore, its beneficial to add regularization in the form of weight decay as the Lagrangian is not unique. The Euler-Lagrange equation is invariant to linear transformation. Hence, the Lagrangian $\mathcal{L}' = \alpha L + \beta$ solves the Euler-Lagrange equation if α is non-zero and $\mathcal{L}$ is a valid Lagrangian. Therefore, adding weight regularization helps obtaining a unique solution.

4.1.1 *Positive-Definite Mass Matrix.*

To obtain a physically plausible kinetic energy, the mass matrix has to be positive definite, i.e.,

$$\boldsymbol{q}^\top \boldsymbol{H}(\boldsymbol{q})\, \boldsymbol{q} > 0 \quad \forall \quad \boldsymbol{q} \in \mathbb{R}^n_0. \tag{14}$$

This constraint ensures all non-zero velocities have positive kinetic energy for all joint configurations. We obtain a positive definite mass matrix by predicting the Cholesky decomposition of the mass matrix with a small positive offset ϵ on the diagonal instead of the mass matrix directly. Therefore, the mass matrix is described by

$$\boldsymbol{H}(\boldsymbol{q}) = \boldsymbol{L}(\boldsymbol{q})\boldsymbol{L}(\boldsymbol{q})^\top + \epsilon \boldsymbol{I}, \tag{15}$$

with lower triangular matrix $\boldsymbol{L}$ and identity matrix $\boldsymbol{I}$. In addition, the diagonal needs to be positive as otherwise, the mass matrix is only positive semi-definite. A positive diagonal is ensured by adding a non-negative linearity to the elements of the diagonal and the positive offset ϵ. Using this parametrization the mass matrix is ensured to be positive definite for all joint configurations. However, this parametrization is numerically not favorable because the random weights the diagonal is close to ϵ for all inputs. This small diagonal is problematic for the forward model as the small eigenvalues of the mass matrix lead to a large amplification of the control torques due to the matrix inverse. The default diagonal can be shifted by adding the positive constant α before the non-linearity. This shift is described by

$$\boldsymbol{l}_{\text{diag}} = \sigma(\boldsymbol{l}_{\text{diag}} + \alpha) + \epsilon,$$

with the vectorized diagonal $\boldsymbol{l}_{\text{diag}}$ and the softplus function σ. If $\alpha > 1$, the mass matrix damps the applied torques when $\boldsymbol{l}_{\text{diag}} \approx \boldsymbol{0}$. This transformation is not essential to obtain good results but balances the forward and inverse losses.

4.1.2 ***Advantages of DeLaN.*** In contrast to the black-box model, this parametrization of the dynamics has three advantages, (1) this approach yields a physically plausible model that conserves energy, (2) is interpretable, and (3) can be used as forward, inverse, and energy model. The DeLaN model is guaranteed to evolve like a mechanical system and is passive (Spong 1987) as the forward dynamics are derived from the physics prior and the positive definite mass matrix for all model parameters. If the system is uncontrolled, i.e., $\boldsymbol{\tau} = \boldsymbol{0}$, the system energy is conserved as the change in energy is described by $\dot{\mathcal{H}} = \dot{\boldsymbol{q}}^\top \boldsymbol{\tau} = \boldsymbol{0}$. In contrast, black-box models can generate additional energy without system inputs. It is important to note that the conservation of energy of DeLaN does not guarantee to prevent the divergence of the model rollouts. The potential energy is not bounded and can accelerate the system indefinitely. Especially outside the training domain, the potential energy is random.

The model is interpretable as one can disambiguate between the different forces, e.g., inertial-, centrifugal-, Coriolis, and gravitational force. This decomposition is beneficial as some model-based control approaches require the explicit computation of the mass matrix, the gravitational force, or the system energy. Furthermore, the same model parameters can be used for the forward, inverse, and energy model. Therefore, the forward and inverse model are consistent. In contrast, black-box models need to learn separate parameters for the inverse and forward model that might not be consistent and cannot obtain the system energy as these cannot be observed.

4.2 Hamiltonian Neural Networks (HNN)

Instead of using Lagrangian Mechanics as model prior for deep networks, Greydanus et al. (2019) proposed to use Hamiltonian mechanics. In this case, the HNN parametrize the Hamiltonian with two deep networks described by

$$\mathcal{H}(\boldsymbol{q}, \boldsymbol{p};\ \psi, \phi) = \frac{1}{2} \boldsymbol{p}^\top \boldsymbol{H}(\boldsymbol{q};\ \psi)^{-1}\, \boldsymbol{p} - V(\boldsymbol{q};\ \phi). \tag{16}$$

It is important to note that HNN predict the inverse of the mass matrix instead of the mass matrix as in DeLaN. Similar to DeLaN, the forward and inverse model can be derived. The forward model $[\dot{\boldsymbol{q}}, \dot{\boldsymbol{p}}] = f(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{\tau};\ \psi, \phi)$ is described by

$$\begin{aligned} \dot{\boldsymbol{q}} &= \frac{\partial \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial \boldsymbol{p}} = \boldsymbol{H}^{-1} \boldsymbol{p}, \\ \dot{\boldsymbol{p}} &= -\frac{\partial \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial \boldsymbol{q}} + \boldsymbol{\tau} = -\frac{1}{2} \left(\boldsymbol{p}^\top \frac{\partial \boldsymbol{H}^{-1}}{\partial \boldsymbol{q}} \boldsymbol{p} \right)^\top - \frac{\partial V}{\partial \boldsymbol{q}} + \boldsymbol{\tau}. \end{aligned}$$

The inverse model $\boldsymbol{\tau} = f^{-1}(\boldsymbol{q}, \boldsymbol{p}, \dot{\boldsymbol{p}};\ \psi, \phi)$ is described by

$$\begin{aligned} \boldsymbol{\tau} &= \dot{\boldsymbol{p}} + \frac{\partial \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial \boldsymbol{q}}, \\ &= \dot{\boldsymbol{p}} - \frac{1}{2} \left(\boldsymbol{p}^\top \boldsymbol{H}^{-1} \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{q}} \boldsymbol{H}^{-1} \boldsymbol{p} \right)^\top + \frac{\partial V}{\partial \boldsymbol{q}}. \end{aligned}$$

The network parameters of the kinetic and potential energy can be obtained by minimizing the squared residual using the observed data consisting of $[\boldsymbol{q}, \boldsymbol{p}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{p}}, \boldsymbol{\tau}]$. This optimization is described by

$$\begin{aligned} \psi^*, \phi^* = \arg\min_{\psi, \phi} \ & \left\| \dot{\boldsymbol{p}} + \frac{\partial \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial \boldsymbol{q}} - \boldsymbol{\tau} \right\|^2_{\boldsymbol{W}_{\dot{\boldsymbol{p}}}} \\ & + \left\| \dot{\boldsymbol{q}} - \frac{\partial \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})}{\partial \boldsymbol{p}} \right\|^2_{\boldsymbol{W}_{\dot{\boldsymbol{q}}}}, \end{aligned} \tag{17}$$

with the diagonal covariance matrix $\boldsymbol{W}_{\dot{\boldsymbol{q}}}$ and $\boldsymbol{W}_{\dot{\boldsymbol{p}}}$ of $\dot{\boldsymbol{q}}$ and $\dot{\boldsymbol{p}}$. The minimization can be solved using the standard gradient based optimization toolkit and automatic differentiation.

4.2.1 ***Differences to DeLaN.*** DeLaN and HNN share the same advantages as both models are derived from the same principle. Therefore, HNN conserve energy, are interpretable, and provide a forward, inverse, and energy model. The main difference is that DeLaN uses position and velocity while HNN uses position and momentum. Depending on the observed quantities either model fits better than the other. A minor difference is that minimizing the residual of the Euler-Lagrange equation is identical to the inverse model loss while minimizing the residual of Hamilton's equations is identical to the forward model loss.

From a numerical perspective, the Hamiltonian mechanics prior is slightly beneficial as the forward and inverse model only relies on the inverse of the mass matrix. Therefore, one does not need to numerically compute the inverse of the predicted matrix. Avoiding the explicit inversion makes the learning and model rollout a bit more stable. The Lagrangian mechanics prior relies on the mass matrix as well on the inverse. Therefore, the inverse of the predicted matrix has to be computed numerically. When the eigenvalues of the mass matrix approach ϵ and $\epsilon \ll 1$, the model rollout and the optimization of the forward model can become numerically sensitive. Therefore, it is important to choose ϵ as large as possible for the corresponding system as this limits the amplification of the acceleration.

4.3 Variations of DeLaN & HNN

Since the introduction of DeLaN (Lutter et al. 2019) and HNN (Greydanus et al. 2019), many other variants and extensions have been proposed within the literature. We provide an overview of the existing work and highlight the differences.

4.3.1 ***Parametrization of $\mathcal{L}$ and $\mathcal{H}$.*** In the previous sections, the Hamiltonian $\mathcal{H}$ and Lagrangian $\mathcal{L}$ were parameterized by two networks predicting the mass matrix, or its inverse, for the kinetic energy and the potential energy. Instead of predicting these two quantities separately, one can also use a single feed-forward network for both quantities. This factorization is described by

$$\mathcal{L} = h(\boldsymbol{q}, \dot{\boldsymbol{q}};\, \psi), \qquad \mathcal{H} = h(\boldsymbol{q}, \boldsymbol{p};\, \psi),$$

with the standard feed-forward network h and the network parameters ψ. Within the literature, this approach was used by (Greydanus et al. 2019; Cranmer et al. 2020) while the (Lutter et al. 2019; Gupta et al. 2020; Zhong et al. 2019; Saemundsson et al. 2020; Finzi et al. 2020) used the representation of kinetic and potential energy. In the following we, will differentiate between both approaches by using the term structured Lagrangian/Hamiltonian and black-box Lagrangian/Hamiltonian. The structured approach, represent the mass matrix and potential energy explicitly while the black-box approach uses a single network to represent the Lagrangian or Hamiltonian. The differences in the model architecture for both approaches are depicted in Figure 2.

One benefit of using a black-box $\mathcal{L}$ and $\mathcal{H}$ is that the quadratic parametrization of the kinetic energy does not apply to relativistic systems. The disadvantages of a single network approach are that this parametrization is computationally more demanding as one needs to compute the Hessian of the network. Evaluating the Hessian of a deep network can be done using automatic differentiation, but is expensive in terms of computation and memory. When using the quadratic kinetic energy, computing the Hessian of the network is not needed. Furthermore, the Hessian may not be invertible if only a single network is used. If the Hessian is singular or nearly singular, the forward model using the Lagrangian prior becomes unstable and diverges. For structured Lagrangian, this problem does not occur as the eigenvalues of the mass matrix are lower bounded.

Most existing work uses standard feed-forward networks to model the system energy, the Hamiltonian or the Lagrangian (Lutter et al. 2019; Greydanus et al. 2019; Zhong et al. 2019; Gupta et al. 2020; Saemundsson et al. 2020; Finzi et al. 2020). Other variants have also applied the physics-inspired networks to graph neural networks (Sanchez-Gonzalez et al. 2019; Cranmer et al. 2020; Botev et al. 2021). Such graph neural networks incorporate additional structure within the network architecture when the system dynamics consist of multiple identical particles without additional constraints. Therefore, these methods exhibit improved performance for modeling N-body problems.

4.3.2 ***Loss Functions and Integrators.*** The loss functions of DeLaN (Equation (13)) and HNN (Equation (17)) express the loss in terms including the acceleration, i.e., $\ddot{\boldsymbol{q}}$ and $\dot{\boldsymbol{p}}$. These quantities are commonly not observed for real-world systems and are approximated using finite differences. The problem of this approximation is that the finite differences amplify the amplitude of high frequency noise components. Therefore, one has to use low-pass filters to obtain good acceleration estimates. A different approach that avoids approximating the accelerations is to only use the forward loss and reformulate the loss in terms of position and velocities. In this case, the loss is described by

$$\psi^*, \phi^* = \arg\min_{\psi,\phi} \parallel \boldsymbol{x}_{t+1} - \hat{\boldsymbol{x}}_{t+1}(\boldsymbol{x}_t, \boldsymbol{\tau}_t; \psi, \phi) \parallel^2, \qquad (18)$$

with the predicted next state $\hat{\boldsymbol{x}}$, the state $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]$ in the case of Lagrangian formulation and the state $\boldsymbol{x} = [\boldsymbol{q}, \boldsymbol{p}]$ in the Hamiltonian formulation. The predicted next state can be obtained by solving the differential equation

$$\begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \frac{\partial^2 \mathcal{L}(\boldsymbol{q},\dot{\boldsymbol{q}})}{\partial^2 \dot{\boldsymbol{q}}}^{-1} \left[\boldsymbol{\tau} - \frac{\partial \mathcal{L}(\boldsymbol{q},\dot{\boldsymbol{q}})}{\partial \boldsymbol{q} \partial \dot{\boldsymbol{q}}} \dot{\boldsymbol{q}} + \frac{\partial \mathcal{L}(\boldsymbol{q},\dot{\boldsymbol{q}})}{\partial \boldsymbol{q}} \right] \end{bmatrix},$$

$$\begin{bmatrix} \dot{\boldsymbol{q}} \\ \dot{\boldsymbol{p}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{p}} \\ -\frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q})\, \boldsymbol{\tau} \end{bmatrix},$$

using any numerical integration approach. In the case of the explicit Euler integration this approach is identical to the loss of Equation (13) and Equation (17). A common choice to compute the next step is the Runge Kutta 4 (RK4) fixed time step integrator (Gupta et al. 2019; Greydanus et al. 2019). This loss formulation also enables a multi-step loss which has been shown to improve the performance of model predictive control for deterministic models (Lutter et al. 2021a)

A more elaborate approach has been proposed by Saemundsson et al. (2020) that combines discrete mechanics with variational integrators. This combination guarantees that even the discrete-time system conserves momentum and energy. The RK4 integration might leak or add energy due to the discrete-time approximation. The main disadvantage of the variational integrator networks is that this approach assumes a constant mass matrix. Therefore, the Coriolis and centrifugal force disappear (Equation (6)) and the acceleration only depends on the position. Within the discrete mechanics literature, extensions exist to apply the variational integrator to multi-body systems with a non-constant mass matrix. However, these extensions are non-trivial and involve solving a root-finding problem within each integration step (Lee et al. 2020).

4.3.3 ***Feature Transformation.*** The previous sections always used generalized coordinates or momentum to describe the system dynamics. However, this formulation can be problematic as these coordinates are unknown or unsuitable for function approximation. For example, continuous revolute joints without angular limits are problematic for function approximation due to the wrapping of the angle at $\pm\pi$. This problem is commonly mitigated using sine/cosine feature transformations. Such feature transformation can be included in physics-inspired networks if the feature transforms mapping from the generalized coordinates to the features $\boldsymbol{z}$ is known and differentiable. The more general problem of only observing the features and unknown feature transformation and generalized coordinates is discussed in Section 6.1.2.

Let g be the feature transform mapping generalized coordinates to the features $\boldsymbol{z} = g(\boldsymbol{q})$. For continuous revolute

joints the feature transform $\boldsymbol{g}(\boldsymbol{q}) = [\cos \boldsymbol{q}_0, \sin \boldsymbol{q}_0]$ avoids the problems associated with wrapping the angle. In this case the Lagrangian is described by

$$\mathcal{L}(\boldsymbol{z}, \dot{\boldsymbol{q}};\ \psi, \phi) = \frac{1}{2} \dot{\boldsymbol{q}}^\top \boldsymbol{H}(\boldsymbol{z};\ \psi)\ \dot{\boldsymbol{q}} - V(\boldsymbol{z};\ \phi).$$

In this case, one can apply the chain rule to obtain the gradients w.r.t. the generalized coordinates, i.e.,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{q}} = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{q}}^\top \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}}.$$

This approach is identical to adding an input layer to the neural network with the hand-crafted transformations. The feature transformation was previously introduced by Zhong et al. (2019). However, the authors only manually derived the special case for continuous angle while this approach can be easily generalized to arbitrary differentiable feature transformations.

4.3.4 ***Actuator Models and Friction.*** The physics-inspired networks cannot model friction directly as the learned dynamics are conservative. Incorporating friction within this model learning approach in a non-black-box fashion is non-trivial because friction is an abstraction to combine various physical effects. For robot arms in free space, the friction of the motors dominates, for mechanical systems dragging along a surface the friction at the surface dominates while for legged locomotion the friction between the feet and floor dominates but also varies with time. Therefore, defining a general case for all types of friction in compliance with the Lagrangian and Hamiltonian Mechanics is challenging. Various approaches to incorporate friction models analytically can be found in (Lurie 2013; Wells 1967).

Most existing works on physics-inspired networks only focus on friction caused by the actuators, which dominates for robot arms (Lutter and Peters 2019; Gupta et al. 2019; Lutter et al. 2020). In this case the friction can be expressed using generalized coordinates and is a non-conservative force. Incorporating other types of friction than actuator friction is non-trivial as these cannot be easily expressed using the generalized force. In this case, one requires the contact-point and contact Jacobian to map the contact-force to the generalized force. For the actuator model, the generalized force required for DeLaN and HNN is expressed using an addition function that modulates the system input and adds friction. This function is described by $\boldsymbol{\tau} = g(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})$ with the system input $\boldsymbol{u}$. For the actuator model, one can either choose a white-box approach that uses a analytic actuator and friction models (Lutter and Peters 2019) or a black-box approach that uses a deep network (Gupta et al. 2019; Zhong et al. 2019). For example, a white-box model can add a friction torque $\boldsymbol{\tau}_f$ to motor torque $\boldsymbol{u}$. Therefore, the generalized force is described $\boldsymbol{\tau} = \boldsymbol{u} + \boldsymbol{\tau}_f$. Within the literature many friction models have been proposed (Olsson et al. 1998; Albu-Schäffer 2002; Bona and Indri 2005; Wahrburg et al. 2018). These models assume that the motor friction only depends on the joint velocity $\dot{\boldsymbol{q}}_i$ of the ith-joint and is independent of the other joints. Common choices for friction are static, viscous, stiction described by

$$\begin{aligned}
&\text{Coulomb Friction} && \boldsymbol{\tau}_f = -\boldsymbol{\tau}_c, \\
&\text{Viscous Friction} && \boldsymbol{\tau}_f = -\boldsymbol{\rho} \odot \dot{\boldsymbol{q}}, \\
&\text{Stiction} && \boldsymbol{\tau}_f = -\tau_s \odot \operatorname{sign}(\dot{\boldsymbol{q}}) \odot \exp\left(-\dot{\boldsymbol{q}}^2 / \nu\right),
\end{aligned}$$

with the elementwise multiplication $\odot$, the Coulomb friction constant $\boldsymbol{\tau}_c$, the viscous friction constant $\boldsymbol{\rho}$ and the stiction constants $\boldsymbol{\tau}_s$ and ν. These friction models can also be combined to yield the Stribeck friction described by

$$\boldsymbol{\tau}_f = -\left(\tau_c + \tau_s \odot \exp\left(-\dot{\boldsymbol{q}}^2 / \nu\right)\right) \odot \operatorname{sign}(\dot{\boldsymbol{q}}) - \boldsymbol{d} \odot \dot{\boldsymbol{q}}.$$

It is important to note that the system is not time-reversible when stiction is added to the dynamics as multiple motor-torques can generate the same joint acceleration (Ratliff et al. 2016).

In contrast to these white-box approaches, Gupta et al. (2019) and Zhong et al. (2019) proposed to add an black-box actuator model. For example, Gupta et al. (2019) proposed to use a black-box control matrix $\boldsymbol{G}(\boldsymbol{q})$ with viscous friction for DeLaN. Therefore, the actuator model is described by

$$\boldsymbol{\tau} = \boldsymbol{G}(\boldsymbol{q}, \dot{\boldsymbol{q}}; \boldsymbol{\theta})\ \boldsymbol{u} - \boldsymbol{\rho} \odot \dot{\boldsymbol{q}}, \tag{19}$$

with the positive friction coefficients $\boldsymbol{\rho}$. The control matrix $\boldsymbol{G}$ is predicted by an additional neural network. Similarly, Zhong et al. (2020) proposed to use a state-dependent control matrix $\boldsymbol{G}(\boldsymbol{q})$ and a positive definite dissipation matrix $\boldsymbol{D}(\boldsymbol{q})$ for HNN. In this case the generalized force is described by

$$\boldsymbol{\tau} = \boldsymbol{G}(\boldsymbol{q})\ \boldsymbol{u} - \boldsymbol{D}(\boldsymbol{q}) \begin{bmatrix} \frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{q}} \\ \frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{p}} \end{bmatrix}.$$

Both matrices are predicted using a deep network. The network parameters of the actuator model are optimized using gradient descent. These black-box actuator models can represent more complex actuator dynamics and even system dynamics violating the assumptions of Lagrangian and Hamiltonian Mechanics. However, this actuator model can also result that the potential and kinetic energy are ignored and only the black-box model dominates the predicted dynamics. To avoid that the actuator model predicts the complete system dynamics, it is beneficial to add penalties to the magnitude of the actuator during the optimization. The existing grey-box model learning literature (Lutter et al. 2020; Hwangbo et al. 2019; Allevato et al. 2020) has shown that these penalties improve the performance.

5 Experiments

In the experiments, we apply physics-inspired deep network models to learn the non-linear dynamics of simulated systems and physical systems. Within the simulation experiments, we want to test whether the different physics-inspired networks learn the underlying structure and highlight the empirical differences of the existing approaches. On the physical systems, we compare the model-based control performance of DeLaN with a structured Lagrangian for the fully-actuated and under-actuated system to standard system identification techniques and black-box

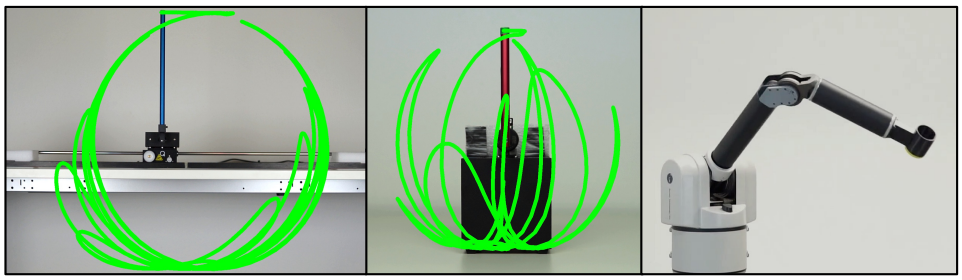

Figure 3. The (a) Cartpole, (b) Furuta pendulum and (c) Barrett WAM used for the evaluation. The Furuta pendulum and cartpole perform a swing-up using the energy controller. The Barrett WAM executes a cosine trajectory with a different frequency per joint.

model learning. We only use DeLaN for the physical systems as for these systems we do not observe the momentum. Hence, only the Lagrangian physics prior is applicable. One could treat the Hamiltonian prior as a latent space problem with the momentum being the latent representation. However, this approach would effectively boil down to the Lagrangian prior. Using these experiments, we want to answer the following questions:

Q1: Do physics-inspired networks learn the underlying representation of the dynamical system?

Q2: Do physics-inspired networks perform better than continuous-time black-box models?

Q3: Can physics-inspired networks be applied to physical systems where the physics prior does not hold?

5.1 Experimental Setup

To answer these questions, we apply the different variations of physics-inspired models to 4 different systems and compare the performance to three baselines. Within the experiments, we denote the physics-inspired networks that only use a single network to represent the Lagrangian or Hamiltonian as black-box DeLaN/HNN. When two separate networks are used to represent that mass-matrix and potential energy, we refer to this approach as structured DeLaN/HNN. The detailed differences between both approaches are described in section 4.3.1. For each of the experiments the dynamics models are learned from a fixed dataset and are trained until convergence. All evaluations are performed on a test dataset that is not contained within the training dataset.

In the following, we briefly introduce the systems and baselines. The code of Deep Lagrangian Networks (DeLaN) and Hamiltonian Neural Networks (HNN) is available at https://github.com/milutter/deep_lagrangian_networks.

5.1.1 ***Plants.***

Within the experiments, we apply the model learning techniques to a simulated two-link pendulum, the Barrett WAM, the cartpole, and the Furuta pendulum. For all physical systems, only the joint position is directly observed. The velocities and accelerations are computed using finite differences and low-pass filters. These filters are applied offline to use zero-phase shift filters that do not cause a phase shift within the observations. The hyperparameters of DeLaN for each system are described in Table 1.

Two-link Pendulum. The two-link pendulum has two continuous revolute joints, is fully actuated, and acts in the vertical x-z plane with gravity. The pendulum is simulated using Bullet (Coumans and Bai 2016).

Cartpole. The physical cartpole (Figure 3a) is an under-actuated system manufactured by Quanser (2018). The pendulum is passive and the cart is voltage controlled with up to 500Hz. The linear actuator consists of a plastic cogwheel drive with high stiction.

Furuta Pendulum. The physical Furuta pendulum (Figure 3b) is an under-actuated system manufactured by Quanser (2018). Instead of the linear actuator of the cartpole, the Furuta pendulum has an actuated revolute joint and a passive pendulum. The revolute joint is voltage controlled with up to 500Hz. The main challenge with this system is the small masses and length scale of the system. These characteristics yield a very sensitive control system.

Barrett WAM. The Barrett WAM (Figure 3c) consists of four actuated degrees of freedom controlled via torque control with 500Hz. The actuators are back-driveable and consist of cable drives with low gear ratios enabling fast accelerations. The joint angles sensing is on the motor-side. Therefore, any deviation between the motor position and joint position due to the slack of the cables cannot be observed. We only use the 4 degree of freedom version as

Table 1. The hyperparameters of DeLaN used for the different dynamical systems. All physical system have a fixed sampling time of 2.0ms. The number of training samples for the 2-DoF robot differs by dataset. See table 2 for more details about the different datasets.

	2-DoF Robot	Cartpole	Furuta Pendulum	Barrett WAM
Total DoF / Actuated DoF	2 / 2	2 / 1	2 / 1	4 / 4
Number of Training Samples	2500 & 10^5	10^5 / ≈ 200s	10^5 / ≈ 200s	10^5 / ≈ 200s
Network Dimension	[2 x 64]	[2 x 256]	[2 x 128]	[2 x 128]
Activation	Tanh	SoftPlus	SoftPlus	SoftPlus
Batch Size	512	1024	1024	1024
Learning Rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Weight Decay	10^{-5}	10^{-5}	10^{-5}	10^{-5}
Optimizer	ADAM	ADAM	ADAM	ADAM

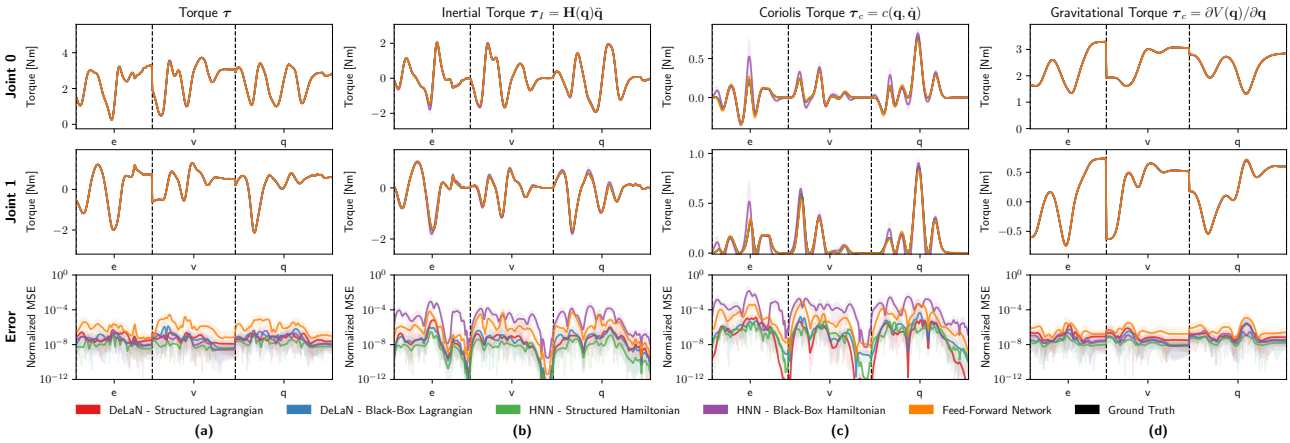

Figure 4. (a) The learned inverse model using the character dataset averaged over 10 seeds. The test character 'e', 'v', 'q' are not contained within the training set. The remaining columns show predicted force decomposition. (b) plots the inertial force $\mathbf{H}\ddot{\boldsymbol{q}}$, (c) the Coriolis and Centrifugal forces $c(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and (d) the gravitational force $g(\boldsymbol{q})$. All physics-inspired networks learn a good inverse model that obtains a lower MSE than the feed-forward network. The Lagrangian approaches learn a better force decomposition than the Hamiltonian approach. This improved performance is especially visible for the inertial, centrifugal, and Coriolis torque.

the wrist and end-effector joints cannot be excited due to the limited range of motion and acceleration.

5.1.2 ***Baselines.*** We use the analytic dynamics model, system identification, and a feed-forward deep network as baselines.

Analytic Model. The analytic model uses the equation of motion derived using rigid body dynamics and the system parameters, i.e., masses, center of gravity, and inertias, provided by the manufacturer. In addition to the rigid body dynamics, these models are augmented with a viscous friction model.

System Identification. This approach requires the knowledge of the analytic equations of motions and infers the system parameters from data. More specifically we use the technique described by Atkeson et al. (1986). This approach showed that for rigid body kinematic trees the inverse dynamics model is a linear model described by

$$\boldsymbol{\tau} = \boldsymbol{A}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\, \boldsymbol{\theta}, \tag{20}$$

with the with hand-crafted features $\boldsymbol{A}(\cdot)$ derived from the kinematics and the system parameters $\boldsymbol{\theta}$. As the inverse dynamics are a linear model, the system parameters can be obtained using linear regression. We additionally penalize deviations from the parameters nominal parameters provided by the manufacturer. In this case the optimal parameters inferred from data are obtained by

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \left(\boldsymbol{A}^\top \boldsymbol{A} + \lambda^2 \boldsymbol{I}\right)^{-1} \boldsymbol{A}^\top \left(\boldsymbol{\tau} - \boldsymbol{A}\boldsymbol{\theta}_0\right), \tag{21}$$

with the nominal parameters $\boldsymbol{\theta}_0$ and the regularization constant λ. The resulting system parameters might not be physically plausible as the individual elements of $\boldsymbol{\theta}$ have additional constraints (Ting et al. 2006). For example the masses have to be positive and the inertias have to adhere to the parallel axis theorem.

Feed-Forward Network. The deep network baseline uses two separate networks, where one describes the forward dynamics and the other the inverse dynamics. This model does not necessarily generate coherent predictions as the parameters of the forward and inverse model are decoupled. Therefore, it is not guaranteed that $f(f^{-1}(\boldsymbol{x})) = \boldsymbol{x}$ holds. The forward model is a continuous-time model and predicts the joint acceleration $\ddot{\boldsymbol{q}}$. Therefore, the deep network baseline is independent of the sampling frequency and uses an explicit integrator as the physics-inspired network. The network parameters are learned by minimizing the normalized squared error of the forward and inverse model. This optimization problem is solved by gradient descent using ADAM. This baseline cannot be applied to the energy experiments, as the system energy cannot be learned by a standard deep network.

5.2 Model Prediction Experiments

For the simulated experiments, we want to evaluate whether the physics-inspired networks can learn the underlying system dynamics and recover the structure with ideal observations. Therefore, we want to observe the data fit and as well as the long-term forward predictions. Furthermore, we want to differentiate between two separate datasets, (1) a large data set with 100k samples spanning the state domain uniformly and (2) a small dataset with only 2.5k samples which consist of trajectories of drawing characters. For the character test dataset, the test set contains different characters than the training set. This dataset only spans a small sub-domain of the state space. The character dataset was initially introduced by Williams et al. (2008) and is available in the UCI Machine Learning Repository (Dheeru and Karra Taniskidou 2017). For training, the datasets are split into a test and training set. The reported results are reported on the test set and averaged over 5 seeds.

5.2.1 ***Inverse Model.*** The results of the inverse model are summarized in Table 2 and visualized in Figure 4. All models learn a good inverse model that fits the test set. When comparing the performance across the large and small datasets one cannot observe a difference in performance. All models perform comparably for the small and large datasets. On average, the physics-inspired networks obtain a lower MSE than the black-box deep network. When comparing

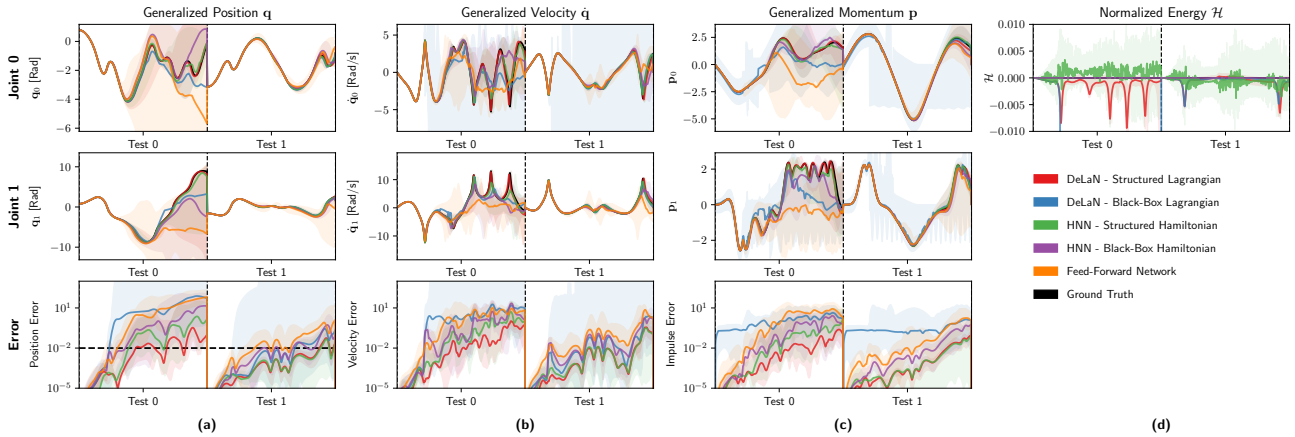

Figure 5. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the forward models for two test trajectories of the uniform dataset averaged over 10 seeds. The structured physics-inspired networks perform the best compared to the standard feed-forward network and the black-box counterparts. Especially the rollout of the black-box Lagrangian commonly diverges as the Hessian of the Lagrangian, which is required for computing the acceleration, becomes close to singular. This nearly singular Hessian causes exploding velocities and consequently also divergence of the estimated momentum computed via $\mathbf{H}\dot{\boldsymbol{q}}$. In the appendix we provide additional plots with only one model per figure to enable an in-depth comparison of the error-bounds.

the structured Lagrangian / Hamiltonian to the black-box counterparts no clear difference is observable for the inverse model.

When comparing the torque decomposition of the inertial, centrifugal, Coriolis, and gravitational forces, all models learn a good decomposition. For the unstructured models, this decomposition can be evaluated by assuming the underlying structure and evaluating the inverse model. For example, the gravitational component by evaluating $\boldsymbol{\tau}_g = f^{-1}(\boldsymbol{q}, \mathbf{0}, \mathbf{0})$. All models learn the underlying structure that fits the true decomposition. Even the black-box feed-forward network obtains a good decomposition despite having no structure. When comparing the MSE error in Table 2, the MSE for the physics-inspired networks is better than the black-box feed-forward network. This difference is especially pronounced for the inertial and Coriolis torque. The difference in the gravitational torque is not so large. When comparing the decomposition of the black-box Lagrangian / Hamiltonian to the structured counterparts, the structured approach outperforms the black-box approach on the inertial and Coriolis torque.

5.2.2 ***Forward Model.*** The results of the forward model are summarized in Table 2 and visualized in Figure 5. Also for the forward model, the physics-inspired networks obtain a better performance on the state error than the feed-forward network. All models perform better on the small character dataset than on the large dataset as the state domain is much smaller than the uniform domain of the large dataset. For the large dataset, the black-box Lagrangian / Hamiltonian approaches are much worse compared to the structured counterparts. This is especially visible for the black-box Lagrangian. The average error and variance is so large because the mass matrix becomes nearly singular for some samples. The nearly singular mass matrix amplifies small differences yielding a very large error.

To compare the long-term predictions of the models, we compare the valid prediction time (VPT) (Botev et al. 2021), which is defined as the duration until the predicted rollout has a larger error than a pre-defined threshold. We define the threshold of the MSE to be 1e−2, which corresponds to an angular error of ≈ 5 degrees. The long-term prediction of

Table 2. The normalized mean squared error (nmse) and mean valid prediction time (VPT) as well as the corresponding confidence interval averaged over 10 seeds. On average the structured Hamiltonian and Lagrangian approaches obtain better forward and inverse models than the black-box counterparts and the standard feed forward neural network. When observing the corresponding phase space coordinates, the Hamiltonian and Lagrangian approaches perform comparable.

		Inverse Model				**Forward Model**	
Uniform Data - # Samples = 100, 000		Torque - $\boldsymbol{\tau}$	Inertial Torque $\boldsymbol{\tau}_I$	Coriolis Torque $\boldsymbol{\tau}_c$	Gravitational Torque $\boldsymbol{\tau}_g$	State Error $\dot{\boldsymbol{x}}$	VPT [s]
DeLaN	Structured Lagrangian	**2.2e−7 ± 3.2e−6**	**2.9e−9 ± 4.5e−8**	**2.5e−8 ± 3.0e−7**	**1.0e−8 ± 3.2e−8**	**3.9e−5 ± 5.9e−4**	**5.64s ± 1.78s**
DeLaN	Black-Box Lagrangian	2.1e−4 ± 4.9e−3	**4.0e−9 ± 2.1e−8**	1.9e−5 ± 3.9e−4	**3.0e−8 ± 7.2e−8**	2.1e+1 ± 1.9e+3	3.59s ± 2.18s
HNN	Structured Hamiltonian	**4.6e−7 ± 1.9e−6**	**4.6e−9 ± 2.8e−8**	**8.1e−8 ± 5.3e−7**	**3.5e−8 ± 8.3e−8**	**1.1e−4 ± 4.4e−4**	**5.09s ± 1.91s**
HNN	Black-Box Hamiltonian	3.3e−5 ± 5.5e−4	9.9e−6 ± 6.0e−5	3.3e−5 ± 3.1e−4	**5.9e−8 ± 1.4e−7**	2.0e−2 ± 3.1e−1	3.80s ± 1.72s
FF-NN	Feed Forward Network	5.8e−5 ± 1.0e−3	2.3e−7 ± 1.5e−6	9.9e−6 ± 1.4e−4	2.9e−7 ± 7.4e−7	6.1e−3 ± 1.1e−1	2.52s ± 0.56s
Character Data - # Samples = 2500							
DeLaN	Structured Lagrangian	**7.7e−8 ± 2.1e−7**	2.0e−7 ± 1.7e−6	**1.1e−6 ± 5.2e−6**	2.0e−7 ± 1.0e−6	**3.5e−5 ± 1.1e−4**	**2.32s ± 0.37s**
DeLaN	Black-Box Lagrangian	**9.9e−8 ± 4.4e−7**	**9.4e−8 ± 4.7e−7**	**1.7e−6 ± 1.2e−5**	1.3e−7 ± 1.1e−6	**4.3e−5 ± 1.7e−4**	**2.76s ± 0.68s**
HNN	Structured Hamiltonian	**1.8e−8 ± 5.5e−8**	**1.5e−8 ± 9.1e−8**	**5.5e−7 ± 1.9e−6**	**2.5e−8 ± 7.0e−8**	**1.5e−5 ± 3.6e−5**	**2.90s ± 0.68s**
HNN	Black-Box Hamiltonian	**5.5e−8 ± 1.8e−7**	5.0e−5 ± 2.7e−4	7.2e−4 ± 4.3e−3	**8.4e−8 ± 2.7e−7**	6.1e−5 ± 1.6e−4	2.19s ± 0.62s
FF-NN	Feed Forward Network	2.2e−6 ± 9.1e−6	4.2e−6 ± 2.5e−5	5.5e−5 ± 3.2e−4	9.2e−7 ± 5.1e−6	6.5e−4 ± 3.0e−3	1.81s ± 0.49s

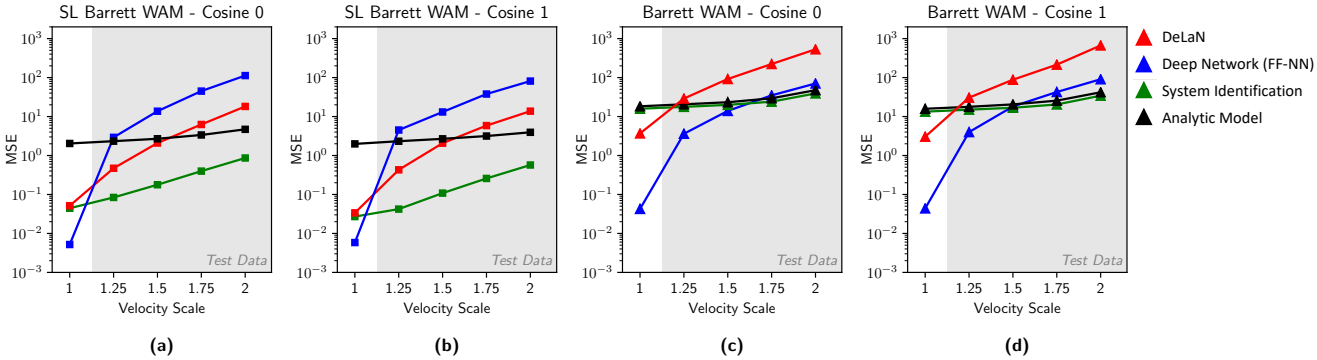

Figure 6. (a, b) The mean squared tracking error of the inverse dynamics control following cosine trajectories for the simulated Barrett WAM. (c, d) the mean squared tracking error on the physical Barrett WAM. The system identification approach, feed-forward neural network, and DeLaN are trained offline using only the trajectories at a velocity scale of 1×. Afterward, the models are tested on the same trajectories with increased velocities to evaluate the extrapolation to new velocities.

the physics-inspired networks is better than the prediction time of the feed-forward network on both datasets (Table 2). Furthermore, the structured variants of DeLaN and HNN perform better than the black-box approaches. The problem of the nearly singular mass matrix can be observed in Figure 5 for the black-box Lagrangian. For one test trajectory and some seeds, the near singular mass matrix lets the trajectory diverge. For the structured HNN and DeLaN this divergence is not observed. Furthermore, the momentum prediction of the black-box DeLaN variant shows the worse accuracy of the network Hessian corresponding to the mass matrix. The predicted momentum of this approach has a much higher variance.

5.2.3 ***Conclusion.*** The simulated experiments show that the physics-inspired networks learn the underlying structure of the dynamical system. These models can accurately predict the force decomposition, momentum, and system energy. Furthermore, the physics-inspired models can learn better forward and inverse models than a standard feed-forward deep network. The structured DeLaN and HNN perform better than the black-box counterparts. The forward and inverse model of the structured DeLaN and HNN do not show any empirical differences when the corresponding phase space coordinates are observed.

5.3 Model-Based Control Experiments

With the experiments on the physical system, we want to evaluate the control performance of the learned models with noisy real-world data. Evaluating the control performance rather than the MSE on static datasets is the more relevant performance measure as the application of the models is control. Furthermore, it has been shown that the MSE is not a good substitute to predict the control performance of a learned model and commonly overestimates the performance (Hobbs and Hepenstal 1989; Lambert et al. 2020; Lutter et al. 2021a). To evaluate the model performance for control, we apply the learned models to inverse dynamics control and energy control. We only apply DeLaN with the structured Lagrangian to the physical systems as the potentially singular mass matrix risks damaging the physical system. HNN do not apply to the system as the momentum cannot be retrieved from the position observations while the velocity can be obtained using finite differences.

5.3.1 ***Inverse Dynamics Control.*** Evaluating learned models by comparing the tracking error of a model-based control law has been a well-established benchmark for evaluating the control performance of models (Nguyen-Tuong et al. 2008, 2009). In this experiment, we use inverse dynamics control as a model-based control law. This feedback controller augments the PD-control law with an additional feed-forward torque to compensate for the non-linear dynamics of the system. Therefore, the inverse dynamics control obtains a better tracking error than the standard PD control. The resulting control law is described by

$$\boldsymbol{\tau} = \boldsymbol{K}_p \left(\boldsymbol{q}_{\text{des}} - \boldsymbol{q}\right) + \boldsymbol{K}_D \left(\dot{\boldsymbol{q}}_{\text{des}} - \dot{\boldsymbol{q}}\right) + f^{-1}(\boldsymbol{q}_{\text{des}}, \dot{\boldsymbol{q}}_{\text{des}}, \ddot{\boldsymbol{q}}_{\text{des}}),$$

with the position and derivative gains $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$. In addition, we test the generalization of the learned models by increasing the velocity of the test trajectories. One would expect that DeLaN would generalize better to scaled velocities as the predicted mass matrix and potential energy only depend on the joint position and are independent of the velocity. The training and testing sequences consist of cosine trajectories with different frequencies for each joint and include a little chirp to avoid learning the Fourier basis. The test and train data differ in the frequency of each joint. These trajectories are the standard approach to excite the system and cover a large state domain. The analytic model of the Barrett WAM is obtained from the JHU LCSR (2018).

The results for the simulated and physical Barrett WAM are summarized in Figure 6. In the simulation, DeLaN and the system identification perform equally well on the training velocity. When comparing generalization, the system identification approach generalizes better than DeLaN to higher velocities. This behavior is expected as system identification obtains the global system parameter while DeLaN only learns a local approximation of the mass matrix and potential energy. In comparison to the feed-forward deep network, DeLaN performs worse on the training velocity but generalizes better to higher velocities. Therefore, the deep network overfits to the training velocity. The analytic model and the system identification have a large performance gap in simulation as we use the same analytic model for simulation and the physical system but the analytic model is optimized for the physical system.

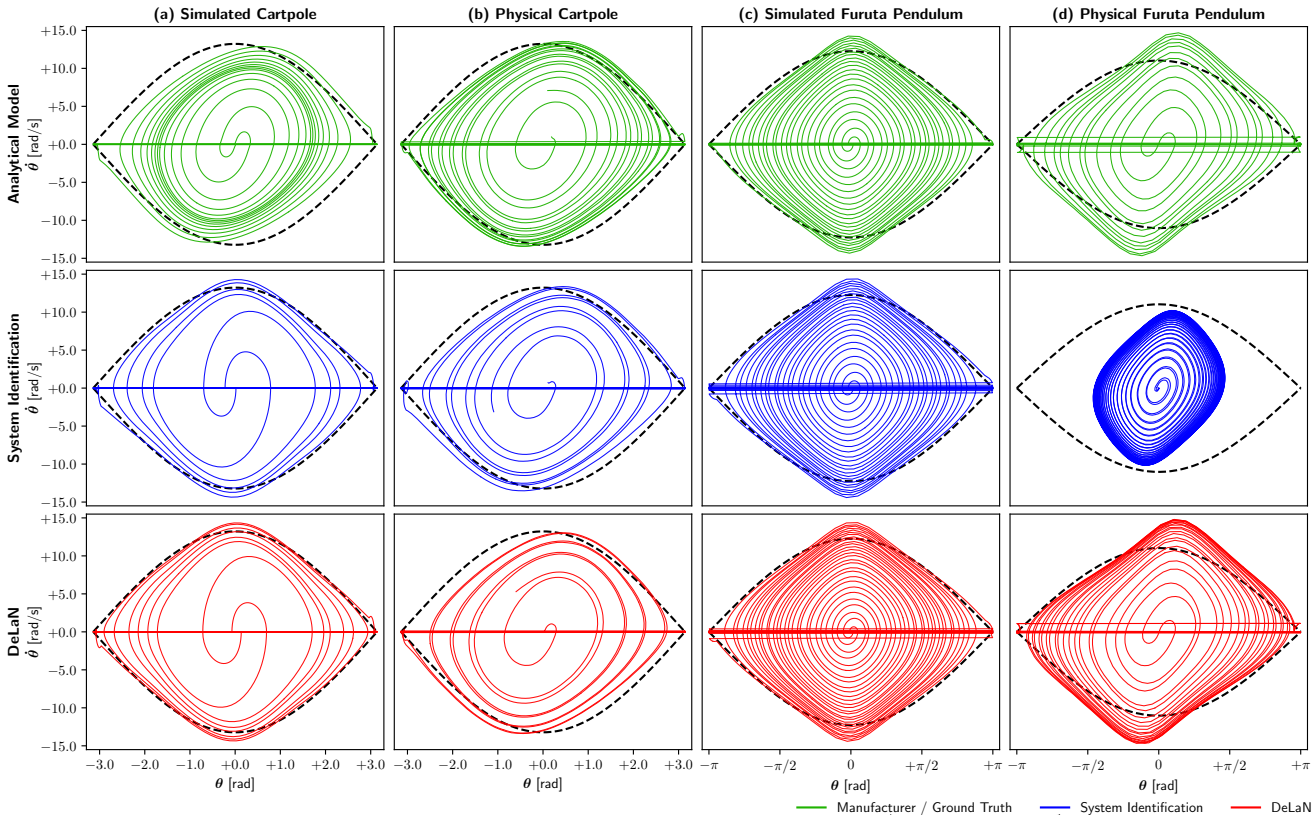

Figure 7. The position θ and velocity $\dot{\theta}$ orbits recorded using energy control to swing up the cartpole and Furuta pendulum. The rows show the different models, i.e., the analytic model, the system identification model, and the DeLaN model while the columns show the different simulated and physical systems. The dashed orbit highlights the desired energy E^*. While the learned and the analytic model can swing up the simulated system and physical Cartpole only the analytic model and DeLaN can swing up the physical Furuta pendulum, while the energy controller using the System Identification model cannot.

On the physical system, the feed-forward network performs the best on the training domain but deteriorates when the velocity is increased. DeLaN performs worse than the deep network but better than the analytic model and the system identification. The analytic model and the system identification model perform nearly identical. The system identification approach is only marginally better. Both approaches generalize better compared to DeLaN and the deep network. This better generalization is expected as the system parameters are global while the other approaches use local approximations. When increasing the velocity, DeLaN and deep network dynamics model degrade in performance. In contrast to the simulation results, where DeLaN extrapolates better than the deep network, the black-box deep network obtains the better generalization on the physical system. The worse performance and generalization of DeLaN on this physical system can be explained by the assumption of rigid body dynamics. This assumption is not fulfilled due to the cable drives and the motor-side sensing. Therefore, DeLaN cannot model every phenomenon with high fidelity. However, DeLaN learns a good approximation that is better than the system identification approach with the same rigid body assumption.

5.3.2 ***Energy Control Control.*** A different approach to test the control performance of the learned models is to apply the learned models to controlling under-actuated systems using an energy controller. More specifically we apply an energy controller to swing up the Furuta pendulum and the cartpole. This energy controller regulates the system energy rather than the position and velocities. The control law is described by

$$\boldsymbol{u} = k_E \left[E(\boldsymbol{q}, \dot{\boldsymbol{q}}) - E(\boldsymbol{q}_{\text{des}}, \dot{\boldsymbol{q}}_{\text{des}}) \right] \text{ sign} \left(\dot{\boldsymbol{q}}_1 \cos(\boldsymbol{q}_1) \right) - k_p \boldsymbol{q}_1,$$

with the energy gain k_E and position gain k_p. We use an additional position controller to prevent the system from hitting the joint limits. The control gains are tuned w.r.t. to the analytic model and fixed for all models. This control task is challenging as the control law relies on the system energy, which cannot be learned supervised. Therefore, the feed-forward network baseline cannot be applied to this task. In contrast to the feed-forward network, the physics-inspired deep network models are the first network models that can be applied to this task as these models can infer the system energy. For the training dataset, we use the energy controller to swing-up the pendulum, stabilize the pendulum at the top and let the pendulum fall down after 2s. Once the pendulum settles the process repeated until about 200s of data is collected.

The results for the simulated and physical experiments are summarized in Figure 7. Videos of the physical experiments are available at [Link]. Within the simulation, the analytic model, the system identification model, and DeLaN achieve the successful swing-up of the cartpole and Furuta pendulum. On the physical cartpole, all approaches achieve the swing-up despite the large stiction of the linear actuator. For the physical Furuta pendulum, only the

analytic model and DeLaN achieve the swing-up. The system identification model does not. The system identification model fails as the linear regression is very sensitive to the observation noise and the small condition number of the features $\boldsymbol{A}$ due to the small dimensions. Therefore, minor changes in the observation can lead to vastly different system parameters. In this specific case, the system identification approach underestimates the masses and hence, exerts too little action to swing up the pendulum and is stuck on the limit cycle.

5.3.3 ***Conclusion.*** The non-linear control experiments on the physical systems show that DeLaN with a structured Lagrangian can learn a good model despite the noisy observations. The resulting model can be used for closed-loop feedback control in real-time for fully-actuated and under-actuated systems. For both systems categories DeLaN achieves a good control performance. It is noteworthy that DeLaN is the first model learning approach utilizing no prior knowledge of the equations of motion that can be applied to energy control. The previous black-box model learning approach could not be applied as the system energy can only be learned unsupervised.

6 Conclusion

Coming back to the initial questions of the experiments. The experimental results have showed that

Q1: physics-inspired networks can learn the underlying representation of the dynamical system unsupervised. The predicted inertial, centrifugal, Coriolis and gravitational forces match the ground-truth (Fig 4). On the physical systems the learned system energy can be used for energy control. The swing-up of under-actuated Furuta pendulum and cartpole where successfully achieved.

Q2: physics-inspired networks may learn better models than continuous-time black-box models with feed-forward networks. Especially, in simulation the physics-inspired networks achieve lower mean squared error (Fig. 5), longer valid prediction times (Table 2) and better generalization (Fig. 6). On the physical system, when the assumptions of the physics-inspired networks are violated, the results are not as clear. While DeLaN performs comparable to system identification techniques, the deep network model can represent physical phenomena that are not captured in the physics prior and achieves a lower tracking error.

Q3: physics-inspired networks can be applied to physical systems. Even though the physical systems violate the physics prior due to non-ideal torque sources and unmodeled phenomena such as cable drives, motor dynamics and backslash, DeLaN was able to learn the system energy. For example, the energy controller utilizing DeLaN was able to solve the swing-up the Furuta pendulum and the cartpole, which has large backslash in the linear actuator.

Similar empirical results were also presented by (Greydanus et al. 2019; Cranmer et al. 2020; Gupta et al. 2019, 2020; Saemundsson et al. 2020; Zhong et al. 2019, 2020). When comparing the different physics prior, Hamiltonian and Lagrangian priors yield comparable models when the corresponding phase space coordinates are observed, i.e., velocities for DeLaN and impulse for HNN. When comparing structured DeLaN and HNN to the black-box DeLaN and HNN, we find that the structured approaches achieve better dynamics models (Table 2). The black-box variants struggle to obtain non-singular network Hessians for all possible system states. For the structured approaches singular Hessians can be prevented by parametrized the kinetic energy such that the eigenvalues of the Hessian are lower-bounded and do not become singular.

Despite the advantages of the physics-inspired models to standard deep networks models, the physics-inspired approaches have drawbacks that prevent the general applicability compared to feed-forward networks. In the following, we discuss these limitations.

6.1 Open Challenges

Physics-inspired deep networks have two main shortcomings, which have not been solved yet. First of all, the current approaches are only able to simulate articulated rigid-bodies *without* contact and second the current approaches rely on knowing and observing the generalized coordinates. Therefore, most of the existing work only showcased these networks for simple n-link pendulums (Zhong et al. 2021) and n-body problems (Sanchez-Gonzalez et al. 2019). For most real-world robotic tasks these assumptions are not full-filled. One frequently does not know or observe the system state and most interesting robotic tasks include contacts. In contrast to physics-inspired networks, black-box dynamics models work with any observations and contacts. These models have been extensively used for model predictive control and are sufficient for complex control tasks (Hafner et al. 2019a,b; Lutter et al. 2021a). Therefore, these challenges need to be addressed to enable the widespread use of physics-inspired methods for robot control. In the following, we highlight the challenges of both limitation and the initial step towards applying these models to contact-rich tasks with arbitrary observations.

6.1.1 ***Contacts.*** Analytically, contact forces can be incorporated by adding generalized contact forces to the Euler-Lagrange equation. In this case the differential equation is described by

$$\frac{d}{dt}\frac{\partial \mathcal{L}(\boldsymbol{q},\dot{\boldsymbol{q}})}{\partial \dot{\boldsymbol{q}}} - \frac{\partial \mathcal{L}(\boldsymbol{q},\dot{\boldsymbol{q}})}{\partial \boldsymbol{q}} = \boldsymbol{\tau} + \underbrace{\sum_{i \in \Omega} \boldsymbol{J}_i^c(\boldsymbol{q}) \boldsymbol{f}_i^c(\boldsymbol{q},\dot{\boldsymbol{q}})}_{\text{Generalized Contact Force}},$$

with the Cartesian contact forces $\boldsymbol{f}^c$, the contact Jacobian $\boldsymbol{J}^c$ connecting the Cartesian forces at the contact point to the generalized coordinates and the set of all active contacts Ω. To compute the generalized contact force, analytic simulators first use the known kinematics and meshes to find all contact points and there respective Jacobians. Afterwards the contact force is computed by solving the linear complementarity problem (LCP). A similar approach can also be used for Hamiltonian mechanics that uses contact impulses rather than forces. Within the physics-inspired deep network literature only Hochlehnert et al. (2021) and Zhong et al. (2021) have included contacts.

However, both existing works only consider special cases with strong assumptions. For example, Hochlehnert et al. (2021) only considers elastic collisions of simple geometric shapes, i.e., circles. In this case, the contact forces can be computed and the contact Jacobian is the identity matrix. Therefore, one only needs to learn an indicator function $\mathbb{1}(\boldsymbol{q})$ being 1 if the contact is active and 0 otherwise. Furthermore, the indicator function is learned supervised. Hence, the training data has to include whether the contact was active or not for each sample. The experiments only apply the proposed algorithm to a ball bouncing on a plane and the Newton cradle.

A different approach was proposed by Zhong et al. (2021). This work augments the physics-inspired network with a differentiable physics simulator to handle the contacts. In this case, a collision detection algorithm determines all active contacts and the contact Jacobians. The contact forces are computed by solving the LCP. In this case, only the coefficients of the contact model, e.g., friction and restitution, are learned from data. Therefore, this approach is similar to the white-box friction models described in Section 4.3.4. This approach also implicitly assumes that the meshes and kinematics are *known*. Without the kinematics and meshes the collision detection algorithms cannot compute the active contacts and Jacobians. If these quantities of the system are known, the analytic equations of motions can be computed and many physical parameters can be approximated from the meshes. Therefore, these assumptions are identical to the required knowledge for system identification using differentiable physics simulators (Werling et al. 2021; Degrave et al. 2019; Heiden et al. 2021). The advantage of physics-inspired networks compared to system identification with differentiable simulators are unknown. The experiments only applied the proposed approach to bouncing disks and a multi-link pendulum with a ground plane.

To summarize, no general way to add contacts to physics-inspired networks has been proposed and shown to work for multi-contact physics with complex geometries. The naive approach to add a single network to model the generalized contact forces is challenging as this reduces the physics-inspired model learning approaches to a black-box model learning technique without proper regularization. Therefore, an important open challenge for physics-inspired networks for robotics is to introduce a generic approach to include multiple contacts.

6.1.2 ***Generalized Coordinates.*** The second limiting assumption is the observation of the generalized coordinates $\boldsymbol{q}, \dot{\boldsymbol{q}}$ or the generalized momentum $\boldsymbol{p}$. For most robotic systems that do not only involve a rigid body manipulator, these coordinates are commonly not observed or known. One usually only obtains observations derived from the generalized coordinates if the system is fully observed. In many cases, the system is only partially observed and one cannot infer the system state from a single observation. For black-box models this is not a problem as these models do not require specific observations and have been shown to learn good dynamics models for complex systems using only images, e.g., Hafner et al. (2019a,b, 2020) and many others.

To overcome this limitation, existing work combined physics-inspired networks with variational autoencoders (VAE) to learn a latent space the resembles the generalized coordinates. In this case, the Lagrangian and Hamiltonian inspired networks are applied in the latent space. Using this approach, the dynamics of single-link pendulums and N-body problems have been learned from artificial images (Greydanus et al. 2019; Zhong et al. 2019; Toth et al. 2019; Saemundsson et al. 2020; Allen-Blanchette et al. 2020). However, these approaches have not been demonstrated on more complex systems and realistic rendering of systems. Botev et al. (2021) also showed that this approach does not necessarily obtain better results than using a normal deep network continuous-time model within the latent space. Therefore, it remains an important open challenge to extend physics-inspired networks to arbitrary observations. The main challenge is to learn a latent space that resembles the generalized coordinates and the naive approach to use a VAE does not seem to be sufficient.

6.2 Summary

We introduced physics-inspired networks that combine Lagrangian and Hamiltonian mechanics with deep networks. This combination obtains physically plausible dynamics models that guarantee to conserve energy. The resulting models are also interpretable and can be used as forward, inverse, or energy models using the same parameters. Previously this was not possible with standard deep network dynamics models. Furthermore, we presented all the existing extensions of physics-inspired networks which include different representations of the Hamiltonian and Lagrangian, different loss functions as well as different actuation and friction models. We elaborated on the shortcomings of the current approaches as these techniques are limited to mechanical systems without contacts and require the observation of generalized positions, velocity, momentum, and forces. Therefore, this summary provides the big picture of physics-inspired networks for learning continuous-time dynamics models of rigid body systems.

Within the experiments, we showed that Deep Lagrangian Networks (DeLaN) and Hamiltonian Neural Networks (HNN) learn the underlying structure of the dynamical system for simulated and physical systems. When the corresponding phase-space coordinates of each model are observed, both models perform nearly identical. On average the structured Hamiltonian and Lagrangian perform better than their black-box counterparts. Especially for the Lagrangian combination, the black-box approach can lead to high prediction errors due to inverting the Hessian of a deep network. Furthermore, we show that these physics-inspired techniques can be applied to the physical system despite the observation noise. The resulting DeLaN models can be used for real-time control and achieve good performance for inverse dynamics control as well as energy control. Especially the latter is noteworthy, as DeLaN is the first model learning technique that utilizes deep networks and can learn the system energy. Previously this was only possible using system identification which requires knowledge of the kinematic structure to derive the equations of motion.

Acknowledgments

This project has received funding from ABB and NVIDIA. Furthermore, we want to thank the open-source projects NumPy (Harris et al. 2020), PyTorch (Paszke et al. 2019) and JAX (Bradbury et al. 2018).

References

Akkaya I, Andrychowicz M, Chociej M, Litwin M, McGrew B, Petron A, Paino A, Plappert M, Powell G, Ribas R et al. (2019) Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113* .

Albu-Schäffer A (2002) *Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme*. PhD Thesis, Technische Universität München.

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

Allevato A, Short ES, Pryor M and Thomaz A (2020) Tunenet: One-shot residual tuning for system identification and sim-to-real robot task transfer. In: *Conference on Robot Learning (CoRL)*.

Anderson B, Hy TS and Kondor R (2019) Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems (NeuRIPS)* .

Atkeson CG, An CH and Hollerbach JM (1986) Estimation of inertial parameters of manipulator loads and links. *The International Journal of Robotics Research* .

Bekkers EJ (2019) B-spline CNNs on Lie groups. *International Conference on Learning Representations (ICLR)* .

Bona B and Indri M (2005) Friction compensation in robotics: an overview. In: *IEEE Conference on Decision and Control (CDC)*.

Book WJ (1984) Recursive lagrangian dynamics of flexible manipulator arms. *The International Journal of Robotics Research* .

Botev A, Jaegle A, Wirnsberger P, Hennes D and Higgins I (2021) Which priors matter? benchmarking models for learning latent dynamics. *Neural Information Processing Systems Track on Datasets and Benchmarks* .

Bradbury J, Frostig R, Hawkins P, Johnson MJ, Leary C, Maclaurin D, Necula G, Paszke A, VanderPlas J, Wanderman-Milne S and Zhang Q (2018) JAX: composable transformations of Python+NumPy programs. URL http://github.com/google/jax.

Calinon S, D'halluin F, Sauser EL, Caldwell DG and Billard AG (2010) Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine* .

Camacho EF and Alba CB (2013) *Model predictive control*. Springer Science & Business Media.

Camoriano R, Traversaro S, Rosasco L, Metta G and Nori F (2016) Incremental semiparametric inverse dynamics learning. In: *International Conference on Robotics and Automation (ICRA)*.

Chen RT, Rubanova Y, Bettencourt J and Duvenaud D (2018) Neural ordinary differential equations. *Advances in neural information processing systems (NeuRIPS)* .

Chen Z, Zhang J, Arjovsky M and Bottou L (2020) Symplectic recurrent neural networks. *International Conference on Learning Representations (ICLR)* .

Choi Y, Cheong SY and Schweighofer N (2007) Local online support vector regression for learning control. In: *International Symposium on Computational Intelligence in Robotics and Automation*.

Chu M, Thuerey N, Seidel HP, Theobalt C and Zayer R (2021) Learning meaningful controls for fluids. *ACM Transactions on Graphics (TOG)* .

Cohen T, Weiler M, Kicanaoglu B and Welling M (2019) Gauge equivariant convolutional networks and the icosahedral CNN. In: *International Conference on Machine Learning (ICML)*.

Cohen T and Welling M (2016) Group equivariant convolutional networks. In: *International Conference on Machine Learning (ICML)*.

Coumans E and Bai Y (2016) Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

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

de Wit CC, Siciliano B and Bastin G (2012) *Theory of robot control*. Springer Science & Business Media.

Degrave J, Hermans M, Dambre J et al. (2019) A differentiable physics engine for deep learning in robotics. *Frontiers in neurorobotics* .

Dheeru D and Karra Taniskidou E (2017) UCI machine learning repository. URL http://archive.ics.uci.edu/ml.

Featherstone R (2007) *Rigid Body Dynamics Algorithms*. Springer-Verlag.

Ferreira JP, Crisostomo M, Coimbra AP and Ribeiro B (2007) Simulation control of a biped robot with support vector regression. In: *IEEE International Symposium on Intelligent Signal Processing*.

Finzi M, Wang KA and Wilson AG (2020) Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints. *Advances of Neural Information Processing Systems (NeuRIPS)* .

Fitzpatrick R (2008) Newtonian dynamics.

Gautier M (1986) Identification of robots dynamics. *IFAC Proceedings Volumes* .

Geist AR and Trimpe S (2021) Structured learning of rigid-body dynamics: A survey and unified view from a robotics perspective. *GAMM-Mitteilungen* .

Geng Z, Haynes LS, Lee JD and Carroll RL (1992) On the dynamic model and kinematic analysis of a class of stewart platforms. *Robotics and autonomous systems* .

Golliday CL and Hemami H (1977) An approach to analyzing biped locomotion dynamics and designing robot locomotion controls. *IEEE Transactions on Automatic Control* .

Greenwood DT (2006) *Advanced dynamics*. Cambridge University Press.

Greydanus S, Dzamba M and Yosinski J (2019) Hamiltonian neural networks. In: *Advances in Neural Information Processing Systems (NeuRIPS)*.

Gupta JK, Menda K, Manchester Z and Kochenderfer M (2020) Structured mechanical models for robot learning and control. In: *Learning for Dynamics and Control (L4DC)*.

Gupta JK, Menda K, Manchester Z and Kochenderfer MJ (2019) A general framework for structured learning of mechanical systems. *arXiv preprint arXiv:1902.08705* .

Ha D and Schmidhuber J (2018) World models. *arXiv preprint arXiv:1803.10122* .

Haarnoja T, Zhou A, Abbeel P and Levine S (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: *International Conference on Machine Learning (ICML)*.

Hafner D, Lillicrap T, Ba J and Norouzi M (2019a) Dream to control: Learning behaviors by latent imagination. *International Conference on Learning Representations (ICLR)* .

Hafner D, Lillicrap T, Fischer I, Villegas R, Ha D, Lee H and Davidson J (2019b) Learning latent dynamics for planning from pixels. In: *International Conference on Machine Learning (ICML)*.

Hafner D, Lillicrap T, Norouzi M and Ba J (2020) Mastering atari with discrete world models. *International Conference on Learning Representations (ICLR)* .

Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, Kern R, Picus M, Hoyer S, van Kerkwijk MH, Brett M, Haldane A, del Río JF, Wiebe M, Peterson P, Gérard-Marchant P, Sheppard K, Reddy T, Weckesser W, Abbasi H, Gohlke C and Oliphant TE (2020) Array programming with NumPy. *Nature* .

Haruno M, Wolpert DM and Kawato M (2001) Mosaic model for sensorimotor learning and control. *Neural computation* .

Heess N, Sriram S, Lemmon J, Merel J, Wayne G, Tassa Y, Erez T, Wang Z, Eslami S, Riedmiller M et al. (2017) Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286* .

Heiden E, Macklin M, Narang Y, Fox D, Garg A and Ramos F (2021) Disect: Differentiable simulation engine for autonomous robotic cutting. *Robotics: Science and Systems (RSS)* .

Hemami H and Wyman B (1979) Modeling and control of constrained dynamic systems with application to biped locomotion in the frontal plane. *IEEE Transactions on Automatic Control (TAC)* .

Hobbs BF and Hepenstal A (1989) Is optimization optimistically biased? *Water Resources Research* .

Hochlehnert A, Terenin A, Sæmundsson S and Deisenroth M (2021) Learning contact dynamics using physically structured neural networks. In: *International Conference on Artificial Intelligence and Statistics (Aistats)*.

Holl P, Koltun V and Thuerey N (2020) Learning to control PDEs with differentiable physics. *International Conference on Learning Representations (ICLR)* .

Huh I, Yang E, Hwang SJ and Shin J (2020) Time-reversal symmetric ode network. *Advances of Neural Information Processing Systems (NeuRIPS)* .

Hwangbo J, Lee J, Dosovitskiy A, Bellicoso D, Tsounis V, Koltun V and Hutter M (2019) Learning agile and dynamic motor skills for legged robots. *Science Robotics* .

Ioannou PA and Sun J (1996) *Robust adaptive control*. Prentice-Hall.

Jansen M (1994) Learning an accurate neural model of the dynamics of a typical industrial robot. In: *International Conference on Artificial Neural Networks (ICANN)*.

JHU LCSR JL (2018) Barrett model containing the 7-dof urdf. URL https://github.com/jhu-lcsr/barrett_model.

Khansari-Zadeh SM and Billard A (2011) Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics* .

Khosla PK and Kanade T (1985) Parameter identification of robot dynamics. In: *Conference on Decision and Control (CDC)*.

Kocijan J, Murray-Smith R, Rasmussen CE and Girard A (2004) Gaussian process model based predictive control. In: *American Control Conference (ACC)*.

Lagaris IE, Likas A and Fotiadis DI (1998) Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks* .

Lagaris IE, Likas AC and Papageorgiou DG (2000) Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks* .

Lambert N, Amos B, Yadan O and Calandra R (2020) Objective mismatch in model-based reinforcement learning. *Learning for Dynamics and Control (L4DC)* .

Ledezma FD and Haddadin S (2017) First-order-principles-based constructive network topologies: An application to robot inverse dynamics. In: *International Conference on Humanoid Robotics (Humanoids)*.

Lee H and Kang IS (1990) Neural algorithm for solving differential equations. *Journal of Computational Physics* .

Lee J, Liu CK, Park FC and Srinivasa SS (2020) A linear-time variational integrator for multibody systems. *Algorithmic Foundations of Robotics XII. Springer* .

Lenc K and Vedaldi A (2015) Understanding image representations by measuring their equivariance and equivalence. In: *Conference on Computer Vision and Pattern Recognition*.

Lenz I, Knepper RA and Saxena A (2015) Deepmpc: Learning deep latent features for model predictive control. In: *Robotics: Science and Systems (RSS)*.

Liu K, Lewis F, Lebret G and Taylor D (1993) The singularities and dynamics of a stewart platform manipulator. *Journal of Intelligent and Robotic Systems* .

Lurie AI (2013) *Analytical mechanics*. Springer Science & Business Media.

Lutter M, Hasenclever L, Byravan A, Dulac-Arnold G, Trochim P, Heess N, Merel J and Tassa Y (2021a) Learning dynamics models for model predictive agents. *arXiv preprint arXiv:2109.14311* .

Lutter M and Peters J (2019) Deep lagrangian networks for end-to-end learning of energy-based control for under-actuated systems. In: *International Conference on Intelligent Robots and Systems (IROS)*.

Lutter M, Ritter C and Peters J (2019) Deep lagrangian networks: Using physics as model prior for deep learning. In: *International Conference on Learning Representations (ICLR)*.

Lutter M, Silberbauer J, Watson J and Peters J (2020) A differentiable Newton-Euler algorithm for multi-body model learning. *arXiv preprint arXiv:2010.09802* .

Lutter M, Silberbauer J, Watson J and Peters J (2021b) Differentiable physics models for real-world offline model-based reinforcement learning. *International Conference on Robotics and Automation (ICRA)* .

Meade Jr AJ and Fernandez AA (1994) Solution of nonlinear ordinary differential equations by feedforward neural networks. *Mathematical and Computer Modelling* .

Miller K (1992) The lagrange-based model of delta-4 robot dynamics. *Robotersysteme* .

Mukerjee A and Ballard D (1985) Self-calibration in robot manipulators. In: *International Conference on Robotics and*

Automation (ICRA).

Nguyen-Tuong D and Peters J (2010) Using model knowledge for learning inverse dynamics. In: *International Conference on Robotics and Automation (ICRA)*.

Nguyen-Tuong D, Seeger M and Peters J (2008) Computed torque control with nonparametric regression models. In: *American Control Conference (ACC)*.

Nguyen-Tuong D, Seeger M and Peters J (2009) Model learning with local gaussian process regression. *Advanced Robotics* .

Olsson H, Åström KJ, De Wit CC, Gäfvert M and Lischinsky P (1998) Friction models and friction compensation. *European Journalf of Control* .

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J and Chintala S (2019) PyTorch: An imperative style, high-performance deep learning library. In: *Advances in Neural Information Processing Systems (NeuRIPS)*.

Qin H (2020) Machine learning and serving of discrete field theories. *Scientific Reports* .

Quanser (2018) Quanser courseware and resources. https://www.quanser.com/solution/control-systems/.

Raissi M, Perdikaris P and Karniadakis GE (2017a) Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561* .

Raissi M, Perdikaris P and Karniadakis GE (2017b) Physics informed deep learning (part ii): data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566* .

Ratliff N, Meier F, Kappler D and Schaal S (2016) Doomed: Direct online optimization of modeling errors in dynamics. *Big data* .

Romeres D, Zorzi M, Camoriano R and Chiuso A (2016) Online semi-parametric learning for inverse dynamics modeling. In: *Conference on Decision and Control (CDC)*.

Romeres D, Zorzi M, Camoriano R, Traversaro S and Chiuso A (2019) Derivative-free online learning of inverse dynamics models. *IEEE Transactions on Control Systems Technology* .

Rueckert E, Nakatenus M, Tosatto S and Peters J (2017) Learning inverse dynamics models in o (n) time with lstm networks. In: *International Conference on Humanoid Robotics (Humanoids)*.

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 (Aistats)*.

Sanchez-Gonzalez A, Bapst V, Cranmer K and Battaglia P (2019) Hamiltonian graph networks with ode integrators. *arXiv preprint arXiv:1909.12790* .

Sanchez-Gonzalez A, Heess N, Springenberg JT, Merel J, Riedmiller M, Hadsell R and Battaglia P (2018) Graph networks as learnable physics engines for inference and control. *International Conference on Machine Learning (ICML)* .

Schaal S, Atkeson CG and Vijayakumar S (2002) Scalable techniques from nonparametric statistics for real time robot learning. *Applied Intelligence* .

Schütt KT, Arbabzadah F, Chmiela S, Müller KR and Tkatchenko A (2017) Quantum-chemical insights from deep tensor neural networks. *Nature communications* .

Siciliano B and Khatib O (2016) *Springer handbook of robotics*. Springer.

Spong MW (1987) Modeling and control of elastic joint robots. *Journal of dynamic systems, measurement, and control* .

Sutanto G, Wang A, Lin Y, Mukadam M, Sukhatme G, Rai A and Meier F (2020) Encoding physical constraints in differentiable Newton-Euler Algorithm. *Learning for Dynamics Control (L4DC)* .

Ting JA, Mistry M, Peters J, Schaal S and Nakanishi J (2006) A bayesian approach to nonlinear parameter identification for rigid body dynamics. In: *Robotics: Science and Systems*.

Toth P, Rezende DJ, Jaegle A, Racanière S, Botev A and Higgins I (2019) Hamiltonian generative networks. *International Conference on Learning Representations (ICLR)* .

Traversaro S, Brossette S, Escande A and Nori F (2016) Identification of fully physical consistent inertial parameters using optimization on manifolds. In: *International Conference on Intelligent Robots and Systems (IROS)*.

Wahrburg A, Bös J, Listmann KD, Dai F, Matthias B and Ding H (2018) Motor-current-based estimation of cartesian contact forces and torques for robotic manipulators and its application to force control. *IEEE Transactions on Automation Science and Engineering* .

Wang R, Kashinath K, Mustafa M, Albert A and Yu R (2020a) Towards physics-informed deep learning for turbulent flow prediction. In: *International Conference on Knowledge Discovery & Data Mining*.

Wang R, Walters R and Yu R (2020b) Incorporating symmetry into deep dynamics models for improved generalization. *International Conference on Learning Representations* .

Weiler M and Cesa G (2019) General E(2)-equivariant steerable CNNs. *Advances in neural information processing systems (NeuRIPS)* .

Wells DA (1967) *Schaum's outline of theory and problems of lagrangian dynamics*. McGraw-Hill.

Wensing PM, Kim S and Slotine JJE (2017) Linear matrix inequalities for physically consistent inertial parameter identification: A statistical perspective on the mass distribution. *IEEE Robotics and Automation Letters (RAL)* .

Werling K, Omens D, Lee J, Exarchos I and Liu CK (2021) Fast and feature-complete differentiable physics for articulated rigid bodies with contact. *Robotics: Science and Systems (RSS)* .

Williams B, Toussaint M and Storkey AJ (2008) Modelling motion primitives and their timing in biologically executed movements. In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhong YD, Dey B and Chakraborty A (2019) Symplectic ode-net: Learning Hamiltonian dynamics with control. *International Conference on Learning Representations (ICLR)* .

Zhong YD, Dey B and Chakraborty A (2020) Dissipative symoden: Encoding Hamiltonian dynamics with dissipation and control into deep learning. *arXiv preprint arXiv:2002.08860* .

Zhong YD, Dey B and Chakraborty A (2021) Extending Lagrangian and Hamiltonian neural networks with differentiable contact models. *Advances of Neural Information Processing Systems (NeuRIPS)* .

Zhou K, Doyle JC, Glover K et al. (1996) *Robust and optimal control*. Prentice Hall.

7 Appendix

7.1 Extended Experimental Results

To make the differences between the different learned models shown in Figure 4 and Figure 5 clearer, we provide a figure per model in the appendix. For more details about the figures please refer to the figures in the article and section 5.

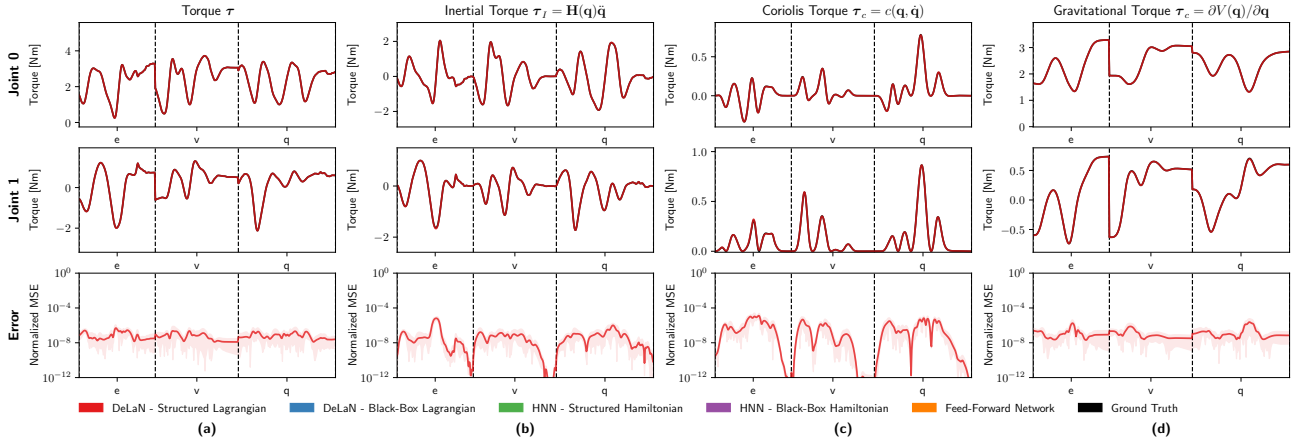

Figure 8. The learned inverse model of structured DeLaN using the character dataset averaged over 10 seeds. For more information see Fig. 4.

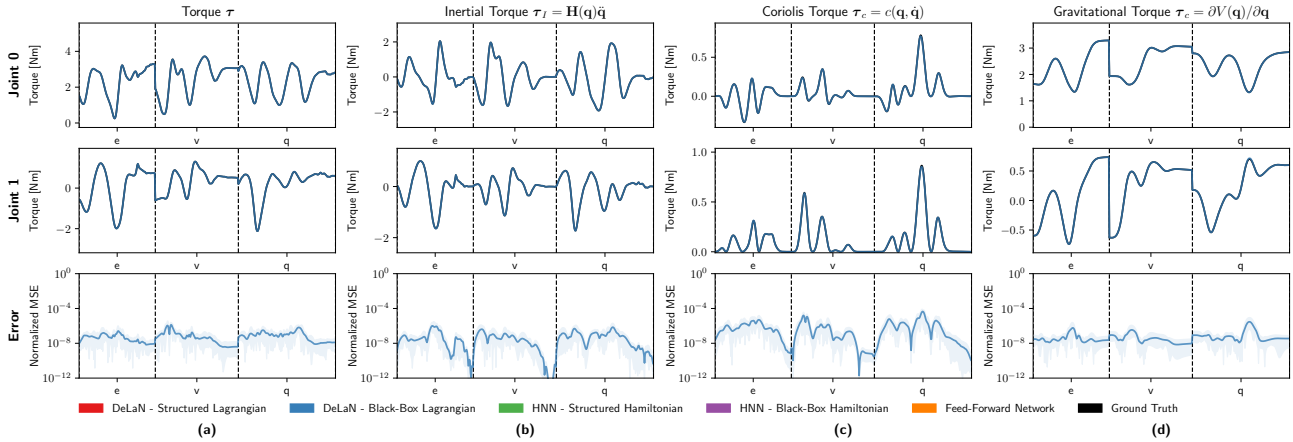

Figure 9. The learned inverse model of black-box DeLaN using the character dataset averaged over 10 seeds. For more information see Fig. 4.

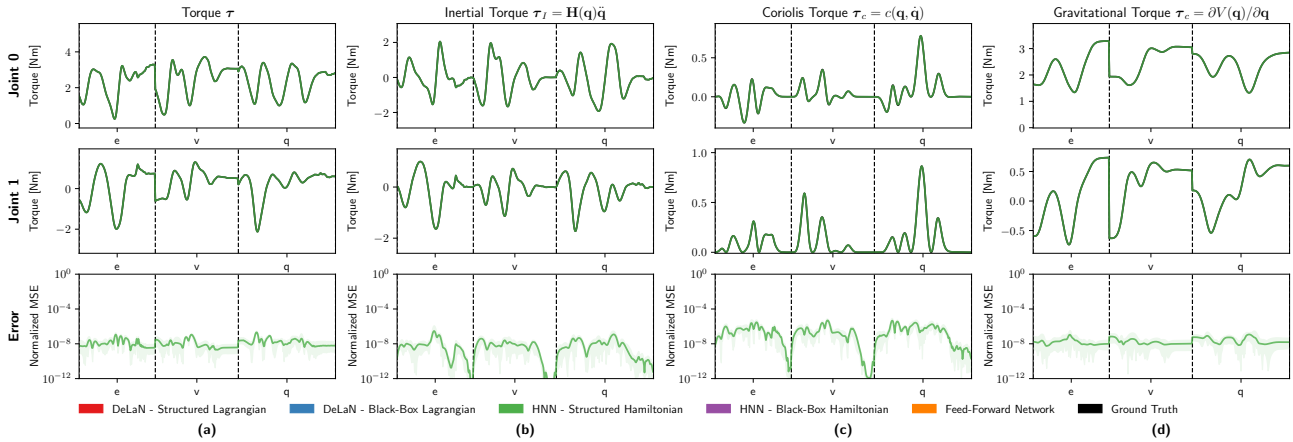

Figure 10. The learned inverse model of structured HNN using the character dataset averaged over 10 seeds. For more information see Fig. 4.

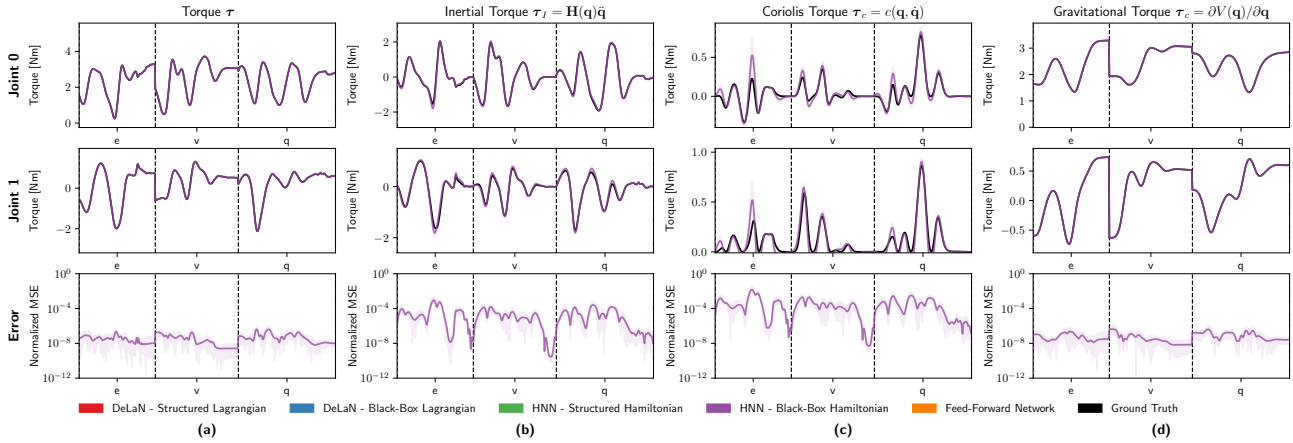

Figure 11. The learned inverse model of black-box HNN using the character dataset averaged over 10 seeds. For more information see Fig. 4.

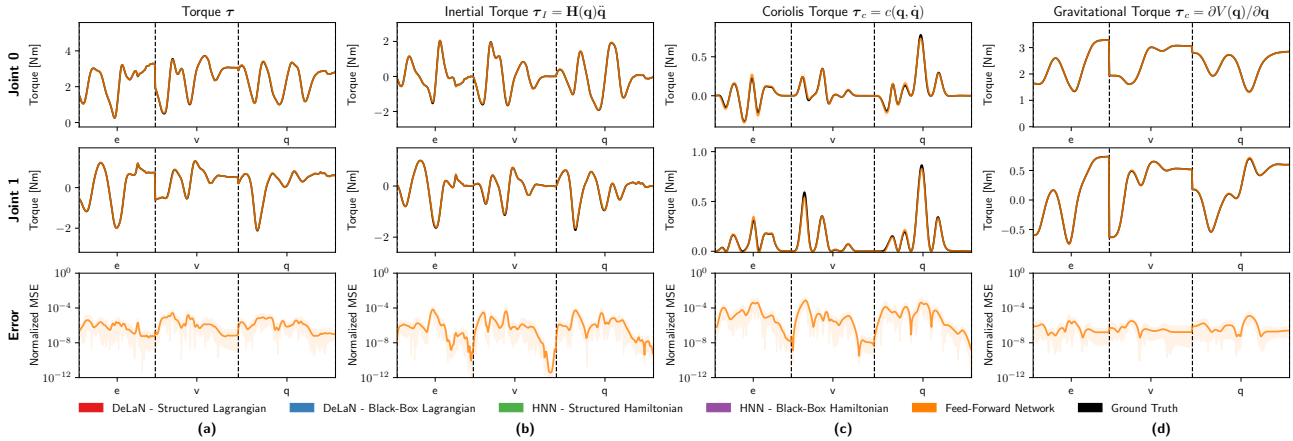

Figure 12. The learned inverse model of the feed-forward network using the character dataset averaged over 10 seeds. For more information see Fig. 4.

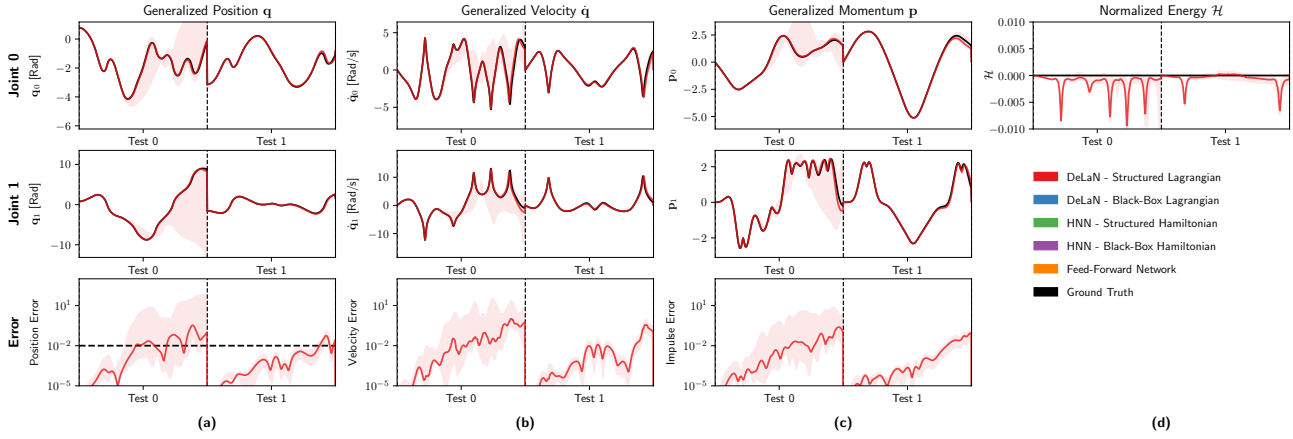

Figure 13. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the structured DeLaN model trained on the uniform dataset averaged over 10 seeds. For more information see Fig. 5

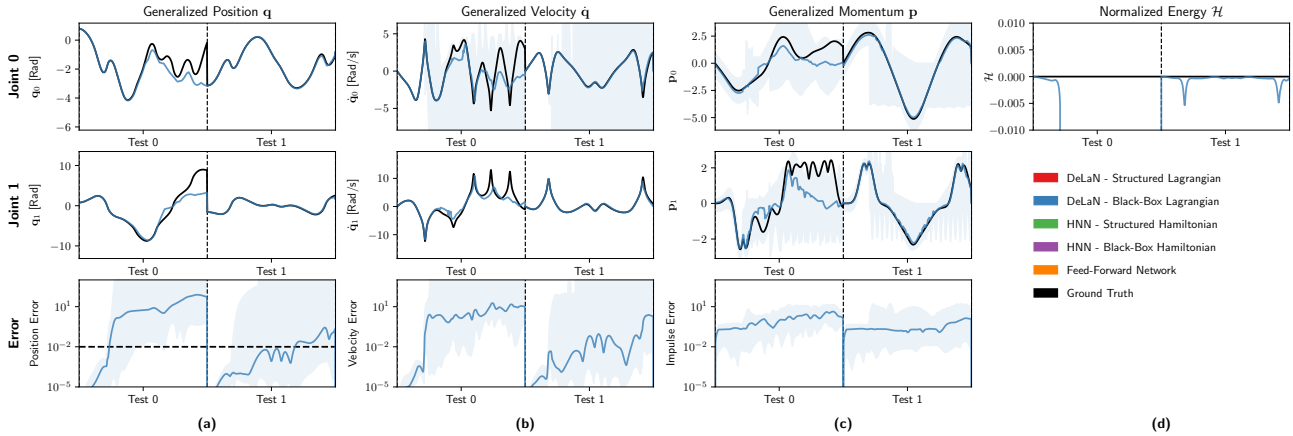

Figure 14. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the black-box DeLaN model trained on the uniform dataset averaged over 10 seeds. For more information see Fig. 5

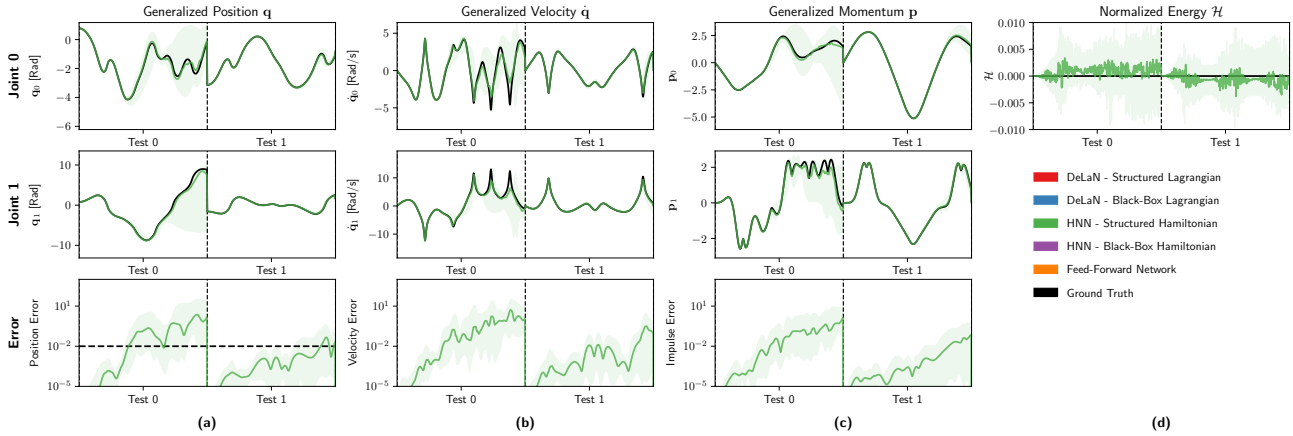

Figure 15. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the structured HNN model trained on the uniform dataset averaged over 10 seeds. For more information see Fig. 5

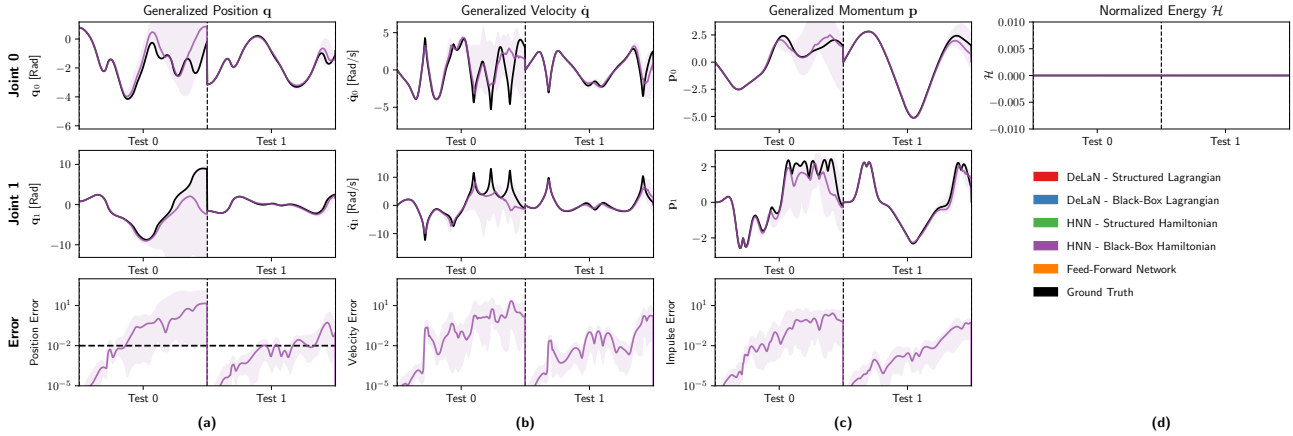

Figure 16. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the black-box HNN model trained on the uniform dataset averaged over 10 seeds. For more information see Fig. 5

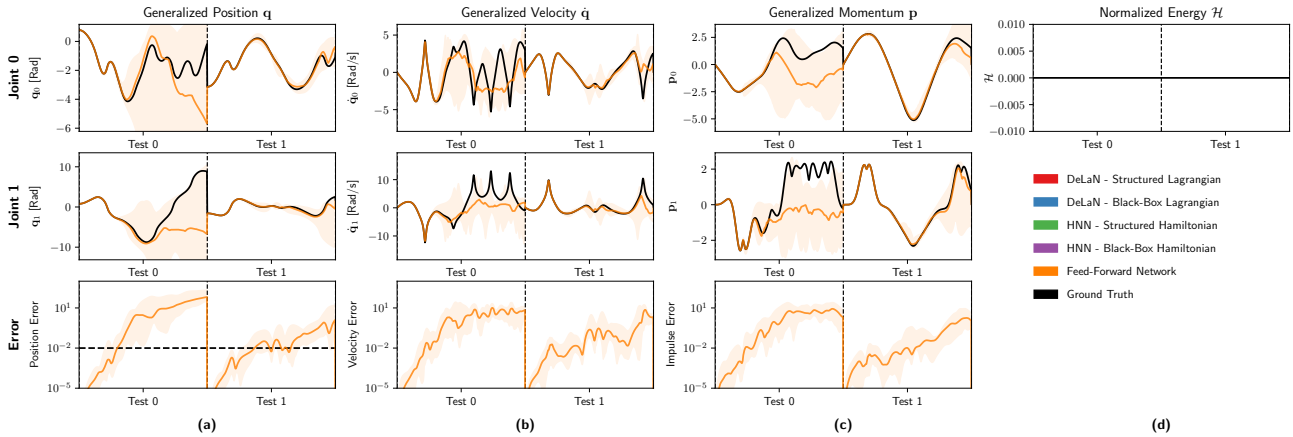

Figure 17. The model rollouts of (a) the position, (b) velocity and (c) momentum, and (d) energy of the feed-forward network model trained on the uniform dataset averaged over 10 seeds. For more information see Fig. 5