# Port-Hamiltonian Neural Networks with State Dependent Ports

Sølve Eidnes, Alexander J. Stasik[*], Camilla Sterud, Eivind Bøhn and Signe Riemer-Sørensen

Department of Mathematics and Cybernetics, SINTEF Digital, 0373 Oslo, Norway

June 6, 2022

**Abstract**

Hybrid machine learning based on Hamiltonian formulations has recently been successfully demonstrated for simple mechanical systems. In this work, we stress-test the method on both simple mass-spring systems and more complex and realistic systems with several internal and external forces, including a system with multiple connected tanks. We quantify performance under various conditions and show that imposing different assumptions greatly affect the performance during training presenting advantages and limitations of the method. We demonstrate that port-Hamiltonian neural networks can be extended to larger dimensions with state-dependent ports. We consider learning on systems with known and unknown external forces and show how it can be used to detect deviations in a system and still provide a valid model when the deviations are removed. Finally, we propose a symmetric high-order integrator for improved training on sparse and noisy data.

## 1 Introduction

Hybrid machine learning is the combination of data driven machine learning with mathematical descriptions of physical systems. By providing physical knowledge to the learning problem, we might reduce the requirements for data quantity and quality. Hybrid machine learning can be implemented in several ways, where two distinct approaches are through soft and hard constraints. Soft constraints typically add penalty terms to the loss function that penalizes violations of physical laws. This procedure is widely applicable, but does not provide any guarantees since the resulting model will be a compromise between non-violation of the constrains and predictive power measured on available data. Hard constraints on the other hand provide mathematical guarantees of compliance with specified laws of physics. This is achieved through enforcing the model structure, and is independent of the available data. Enforcing hard constraints introduces bias which will typically reduce the expressiveness of the machine learning model. Hard constraints are therefore challenging to implement, as erroneous assumptions about the underlying physical system may yield wrongly biased models with poor predictive qualities.

[*]Corresponding author: alexander.stasik@sintef.no

The Hamiltonian formulation of mechanics was originally proposed in the 1830's as a generalisation of classical Newtonian mechanics [19]. Since then it has been extended and applied to mechanics, optics, electrodynamics and quantum physics, among many other fields of physics. Any closed physical system can be described by a Hamiltonian function (or the related Lagrangian). However, the Hamiltonian formulation lacks support for external interactions such as energy losses due to friction and control of the system through external forces. Such interactions are commonly present in real-world systems and crucial for engineering applications. The port-Hamiltonian formulation [32] has been developed to overcome these limitations enabling interactions and energy exchanges via *ports* and a corresponding Dirac structure. The port-Hamiltonian formulation has been successfully applied in various domains ranging from electrical circuits to chemistry [28].

The Hamiltonian neural network (HNN), first introduced by Greydanus et al., is a hybrid machine learning framework imposing hard constraints on a data-driven model [16]. HNNs attempt to learn the Hamiltonian function with a neural network while the system dynamics are given by the classical symplectic Hamiltonian structure. This implies that the expressiveness of the model lies in the learning of the Hamiltonian during training, while the hard-constrained structure guarantees that this approximate Hamiltonian is preserved. Since the Hamiltonian framework is not designed to model non-closed physical systems, HNN models are insufficient for many practical applications. This has inspired several extensions of the HNN framework to facilitate controlled systems [36], dissipative systems [35] and port-Hamiltonian system descriptions [11, 13, 14], generalizing to situations without exact energy preservation.

Other extensions and improvements of HNNs worth mentioning in this context are generalization to Poisson systems [20], generalization to coordinate-free Hamiltonian systems [6], and embedding the system in a higher-dimensional space and constraining to a submanifold [15, 4]. However, these works only consider systems with exact energy preservation, leaving application of the proposed techniques to e.g. port-Hamiltonian system descriptions to future research.

To our knowledge, the previous work most closely related to the present paper is that of Duong and Atanasov [13, 14] where they show how to learn disturbances of a controlled system. Our approach is however more general. It is not restricted to the SE(3) manifold, but instead builds on the Hamiltonian neural networks of Greydanus et al. [16], via the SymODEN [36, 35] and port-Hamiltonian neural network (PHNN) [11] frameworks, and is defined for systems on any manifold. Moreover, in contrast to most of the aforementioned references, we do not assume any specific structure on the Hamiltonian, e.g. that the Hamiltonian must be separable.

For training of dynamical systems, the derivatives of the state variables must be available, either exactly, which would rarely be the case for sensor-based data, or through estimation. Greydanus et al. suggest that the derivatives can be approximated by finite differences when unknown [16]. This is a strong limitation, especially when considering noisy data. To handle the lack of known derivatives, several works propose performing integration at each training step and evaluating the loss using the approximated state variables [29, 7, 10, 13], e.g. with the Neural Ordinary Differential Equation method [5]. In this paper we follow an alternative approach and train on an integration scheme directly as done in [24, 21, 8], see Section 3.

The main contributions of this paper are

- Extensive exploration of the capabilities of PHNNs for use on complex real-life systems.
- Extension of the PHNNs of [11] to more complex systems and demonstration of our models' ability to learn the external port even if it depends on the state variables.

- We propose learning of external ports and show how we can have an accurate model also after this port is removed or replaced. This is similar to the learning of disturbances in [13, 14], but more general.
- We introduce a symmetric fourth order integrator for accurate training without knowing the derivatives or assuming any structure on the Hamiltonian.

## 1.1 Hamiltonian formulation

The Hamiltonian formulation describes general closed systems obeying energy conservation. A physical system can be described by a set of $2n$ generalized coordinates: $n$ generalized positions $q \in \mathbb{R}^n$ and $n$ corresponding generalized momenta $p \in \mathbb{R}^n$. Note that $q$ and $p$ only correspond to classical position and momenta for simple mechanical systems. The Hamiltonian $\mathcal{H}(q,p)$ describes the total energy of the system and is connected to the dynamics via

$$\begin{pmatrix} \dot{q} \\ \dot{p} \end{pmatrix} = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix} \begin{pmatrix} \frac{\partial \mathcal{H}}{\partial q} \\ \frac{\partial \mathcal{H}}{\partial p} \end{pmatrix} \tag{1}$$

where $I_n$ is the $n$-dimensional identity matrix and $\dot{x}$ denotes the time derivative of $x$. Given $\mathcal{H}$ and initial conditions $\{q_0, p_0\}$, the system is fully specified. Systems may have several invariants, and the term energy is often used interchangeably with invariant also in cases when it is not the energy of the system in the physical sense.

The HNNs of [16] use a neural network $\hat{\mathcal{H}}_\theta$ with weights $\theta$ to approximate the Hamiltonian $\mathcal{H}(q,p)$ of a system. Then taking the gradient, i.e. the partial derivatives, of this approximated Hamiltonian and applying the general Hamiltonian structure (1), the model is trained by minimizing the difference between the predicted and actual (approximated) derivatives for the training data.

More generally than the system (1) considered in most of the literature on HNNs, we can consider non-canonical Hamiltonian systems

$$\dot{x} = S(x)\nabla\mathcal{H}(x), \qquad x \in \mathbb{R}^d, \tag{2}$$

for some skew-symmetric matrix $S(x) = -S(x)^T$. Such a formulation (2) exists for any function $\mathcal{H} : \mathbb{R}^d \to \mathbb{R}$ that is an invariant of the first-order ordinary differential equation $\dot{x} = f(x)$, i.e.

$$\frac{d\mathcal{H}}{dt} = \nabla\mathcal{H}(x)^T f(x) = 0.$$

The matrix $S(x)$ may or may not depend on $x$, and is generally not unique if $d > 2$ [25].

## 1.2 Port-Hamiltonian formulation

A generalization of (2) that includes dissipation of $\mathcal{H}$ and external forces is given by

$$\dot{x} = (S(x) - R(x))\nabla\mathcal{H}(x) + F(x,t), \qquad x \in \mathbb{R}^d, \tag{3}$$

where $R(x)$ is a positive semi-definite matrix for all $x$. This can be viewed as a more general form of the port-Hamiltonian systems of van der Schaft [31, 32]; we assume no specific structure on $F : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$. It is also closely connected to the General Equation for Non-Equilibrium

Reversible-Irreversible Coupling (GENERIC) formalism from thermodynamics [17, 26], and the methods presented here could be extended to that setting. This would be similar to what is done by Zhang et al. [34], but they do not consider external forces. In the following, we assume that $S$ and $R$ are independent of $x$. The case of $S$ dependent on $x$ but $R = F = 0$ gives Poisson systems. A generalization of HNN to that class is treated in [20].

The key innovation of port-Hamiltonian neural networks is to model $H$ and $F$ by separate neural networks and thus learn the internal and external energy separately [11]. Even when we assume $S$ to be known, the form (2) will not generally be unique. Hence, depending on the system and our assumptions we often need some regularization to learn the simplest model.

## 2 Example systems

In the following, we introduce two simple systems and show how they can be described using the port-Hamiltonian formulation. We will use those examples throughout the paper.

### 2.1 Damped and forced mass-spring system

Consider a mass-spring system with damping and external force,

$$m\ddot{x} + c\dot{x} + kx = f(x, t), \tag{4}$$

where $m$ is the mass, $c$ is the damping coefficient, $k$ is the stiffness coefficient, and $f$ is the force term. To describe the system in the port-Hamiltonian framework, we set $q = x$ and $p = m\dot{x}$, such that $q$ is position and $p$ is momentum. Then

$$\begin{pmatrix} \dot{q} \\ \dot{p} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & -c \end{pmatrix} \begin{pmatrix} \frac{d\mathcal{H}}{dq} \\ \frac{d\mathcal{H}}{dp} \end{pmatrix} + \begin{pmatrix} 0 \\ f(q,p,t) \end{pmatrix} \tag{5}$$

for $\mathcal{H}(q,p) = \frac{1}{2}kq^2 + \frac{1}{2m}p^2$. This is on the form of (3) with $x := (q,p)$.

In [11] a similar system is discussed but with the assumption that the external force $f$ depends only on time, and not on the state variables. Thus $f(t)$ could be modelled by a separate neural network also without using the Hamiltonian structure. If the external forces cannot be assumed to be strictly time-dependent, there is not uniqueness in the separation between the change in energy stemming from damping and external ports. This makes the learning considerably more challenging as will be discussed in Section 4.1. However, in this paper we show that we are able to separate damping from external ports even when we consider larger systems and uniqueness in the solution is not guaranteed.

### 2.2 Connected tanks in a port-Hamiltonian formulation

We construct a system of $N$ tanks connected by $M$ pipes, as described in [33]. The system can be regarded as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the vertices $\mathcal{V}$ representing the tanks and the edges $\mathcal{E}$ representing the pipes. With $\nu_i$ the flow through pipe $i$, $\mu_j$ as the volume of the fluid stored in tank $j$, and $B$ as the incidence matrix of the graph, conservation of volume gives

$$\dot{\mu} = -B\nu. \tag{6}$$

We assume the flow through a pipe is given as

$$J_i \dot{\nu}_i = P_k - P_l - \lambda_i(\nu_i) \tag{7}$$

where the constants $J_i$ depend on the density of the fluid and the pipe dimensions, $P_k$ and $P_l$ are the pressures at the two ends of the pipe, and $\lambda_i(\nu_i)$ is the friction term [9]. In our numerical experiments we use $\lambda_i(\nu_i) = r_i \nu_i$ and assume that the $R$ in (3) is independent of the state variables. A generalization where we learn a more expressive friction term is entirely feasible within our framework, but left for a future study. The kinetic energy stored in the flow within a pipe is given by $E_i^{\text{pipe}} = \frac{1}{2} J_i \nu_i^2$. The gravitational pressure on the bottom of a tank $j$ is given by $P_j = \rho g \frac{\mu_j}{A_j}$ with $A_j$ as the tank footprint, $\rho$ as the density of the fluid and $g$ as the gravitational constant, and the associated potential energy in the tank is given by $E_j^{\text{tank}} = \frac{g\rho}{2A_j} \mu_j^2$. We use the energy variable $\phi_i := J_i \nu_i$ in the Hamiltonian:

$$\mathcal{H}(\phi, \mu) = \sum_i^M \frac{1}{2J_i} \phi_i^2 + \sum_j^N \frac{g\rho}{2A_j} \mu_j^2. \tag{8}$$

In the absence of friction or external forces, we can rewrite (6) and (7) as (2) with $x := (\phi, \mu)$ and

$$S = \begin{bmatrix} 0_{M\times M} & B^T \\ -B & 0_{N \times N} \end{bmatrix}. \tag{9}$$

Including friction and external ports, we get the port-Hamiltonian system

$$\begin{bmatrix} \dot{\phi} \\ \dot{\mu} \end{bmatrix} = \begin{bmatrix} -R & B^T \\ -B & 0_{N\times N} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathcal{H}}{\partial \phi} \\ \frac{\partial \mathcal{H}}{\partial \mu} \end{bmatrix} + \begin{bmatrix} F_p \\ F_t \end{bmatrix}, \tag{10}$$

where $R$ is an $M \times M$ diagonal matrix with elements $r_i$, and $F_p$ and $F_t$ are the external forces acting on the pipes and the tanks, respectively.

Throughout the paper, we consider a tank system with four tanks and five pipes, connected as shown by the graph in Figure 1. Unless otherwise specified, we use the following parameters when simulating the system: $\rho = 1$, $J_i = 0.02$ for all $i$, $A_j = 1$ for all $j$, $R = \text{diag}(0.03, 0.03, 0.09, 0.03, 0.03)$, $F_p = (0,0,0,0,0)$ and $F_t = (0,0,0,-10\min(0.3, \max(\mu_4, -0.3)))$. That is, an external port is acting on the fourth tank. With the exception of the experiments in Section 5, we assume that we know that only the fourth tank has a port, and try to learn this together with $R$ and $H$. For convenience, no units are specified and the problem is considered as scale-free.

## 2.3 Implementation and hyperparameters

Following [16, 7, 24], we use fully connected neural networks with two hidden layers of 100 neurons each. Furthermore, we use hyperbolic tangent (TANH) and Rectified Linear Unit (ReLU) as activation function for the first and second hidden layer, respectively, while [16, 7, 24] use TANH for both. This was discovered to significantly improve performance as the modelled systems grows in size and complexity. We relate this to the fact that the true Hamiltonian is a linear combination of nonlinear functions, and note that other network architectures may be preferable when modelling systems with different dynamics. In all experiments, we use the Adam optimizer with batch size 256, learning rate $10^{-3}$, and the mean squared error (MSE) as loss function [22].

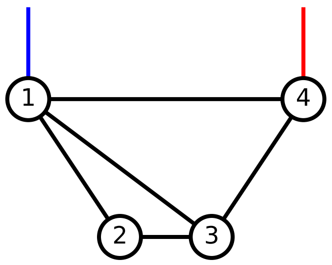

Figure 1: Graph showing how the five pipes connect the four tanks (1-4) and the external port on the fourth tank (red).

## 3 Choice of discretization method for training data

When training PHNNs, the right hand side of (3) as approximated by the model is compared to the left hand side as given or approximated by the training data. When the left hand side derivatives cannot be measured directly, these must be approximated from the training. The choice of method for approximation will affect the training. This problem is not unique to the PHNN, but is a general challenge when applying any data-driven model for estimating a dynamical system described by a differential equation.

Consider a general first-order ordinary differential equation (ODE)

$$\dot{x} = f(x,t), \quad x \in \mathbb{R}^d, \quad f : \mathbb{R}^d \to \mathbb{R}^d. \tag{11}$$

In this work, models are trained on an approximation of the ODE (11) as made by some discretization method corresponding to a numerical integration scheme, as done in [24, 21, 8]. That is, given the integrator

$$\frac{x^{n+1} - x^n}{\Delta t} = \Phi_{\Delta t}(f, x^n, x^{n+1})$$

we train the model $\hat{f}_\theta$ of $f$ using the loss function

$$\mathcal{L} = \| \frac{x^{n+1} - x^n}{\Delta t} - \Phi_{\Delta t}(\hat{f}_\theta, x^n, x^{n+1}) \|_2^2,$$

given for one data point $x^n$ and barring regularization. In contrast, works like [16, 7, 11] either assume derivatives are known or perform one or more integration steps at each training step. Different integrators have been proposed in the aforementioned references, building on established theory from the field of numerical integration, e.g. on symplectic methods for systems with invariants [18]. However, the inverse problem of learning an ODE from data points is different from integrating a known system, and might alter demands of the methods. For instance, a certain class of implicit integrators do not depend on intermediate steps and are thus explicitly given by the known data in the inverse problem. Moreover, they may be less expensive than comparable explicit integrators when used for training.

We consider two discretization methods: The second-order implicit midpoint method

$$\frac{x^{n+1} - x^n}{\Delta t} = f\left(\frac{x^n + x^{n+1}}{2}\right) \tag{12}$$

and the fourth-order method

$$\begin{aligned}\frac{x^{n+1}-x^n}{\Delta t} &= \frac{1}{2}f\left(\frac{x^n+x^{n+1}}{2} - \frac{\sqrt{3}}{6}\Delta t f\Big((\frac{1}{2}+\frac{\sqrt{3}}{6})x^n + (\frac{1}{2}-\frac{\sqrt{3}}{6})x^{n+1}\Big)\right) \\ &\quad + \frac{1}{2}f\left(\frac{x^n+x^{n+1}}{2} + \frac{\sqrt{3}}{6}\Delta t f\Big((\frac{1}{2}-\frac{\sqrt{3}}{6})x^n + (\frac{1}{2}+\frac{\sqrt{3}}{6})x^{n+1}\Big)\right),\end{aligned} \tag{13}$$

given here only for autonomous systems for reasons of brevity. The implicit midpoint method is a symmetric and symplectic integrator. We propose the scheme (13) as a method specifically designed for the inverse problem, rather than integration. It is symmetric but not symplectic. As a numerical integrator it is an implicit Runge–Kutta method, and more specifically a mono-implicit Runge–Kutta (MIRK) method [30, 3], but as a discretization of (11) it is explicitly given by $x^n$ and $x^{n+1}$. This distinguishes it from e.g. the Gauss–Legendre method of order four, which would require a system of equations to be solved by e.g. Newton's method at each training step. Moreover, (12) and (13) are applicable for non-separable Hamiltonian systems, in contrast to the second order leapfrog method used in [7] and Yoshida's fourth order method, used in [12, 10].

The computational cost of a method used during training is dominated by the number of evaluations of $f$. Thus, the implicit midpoint method is comparable to the forward Euler method, while (13) in turn is approximately four times as expensive as those and comparable to the classic Runge–Kutta method. The advantage of using a higher-order method is generally most prevalent when data is sparse, while symmetric methods deal well with noise. Thus, even though (13) generally performs best, the implicit midpoint method might be preferable when the training data is obtained at a sufficiently high frequency and computational cost is an issue.

### 3.1 Training models on tank system data

Consider the system of four tanks and five pipes described in Section 2.2. Four different discretization methods are used for training a PHNN modelling the tank system for four different data sets. The four discretization methods are forward Euler, the Runge–Kutta 4 method (RK4), the implicit midpoint method, and the symmetric fourth order scheme (13) (SRK4). The four data sets are made to reflect different levels of data quality, reflecting the scenarios "high sampling rate and many samples without noise", "high sampling rate and many samples with noise", "low sampling rate and few samples without noise", "low sampling rate and few samples with noise". The sampling time is 1/100 or 1/30 and the number of trajectories sampled is 300 or 100, with each trajectory lasting for one time unit. This corresponds to there being either 30000 or 3000 training points in a data set. Either there is no noise or Gaussian noise with a standard deviation $\sigma = 0.03$ is added to the measurements of the system state.

For each data set and for each discretization method, 10 PHNNs are trained for 1000 epochs. A test set consisting of 10 trajectories with random initial conditions is used to test the performance of the models. Figures 2 and 3 and Table 1 demonstrate that the symmetric methods deal well with noise, and a higher order symmetric method is required when the data is also sparse.

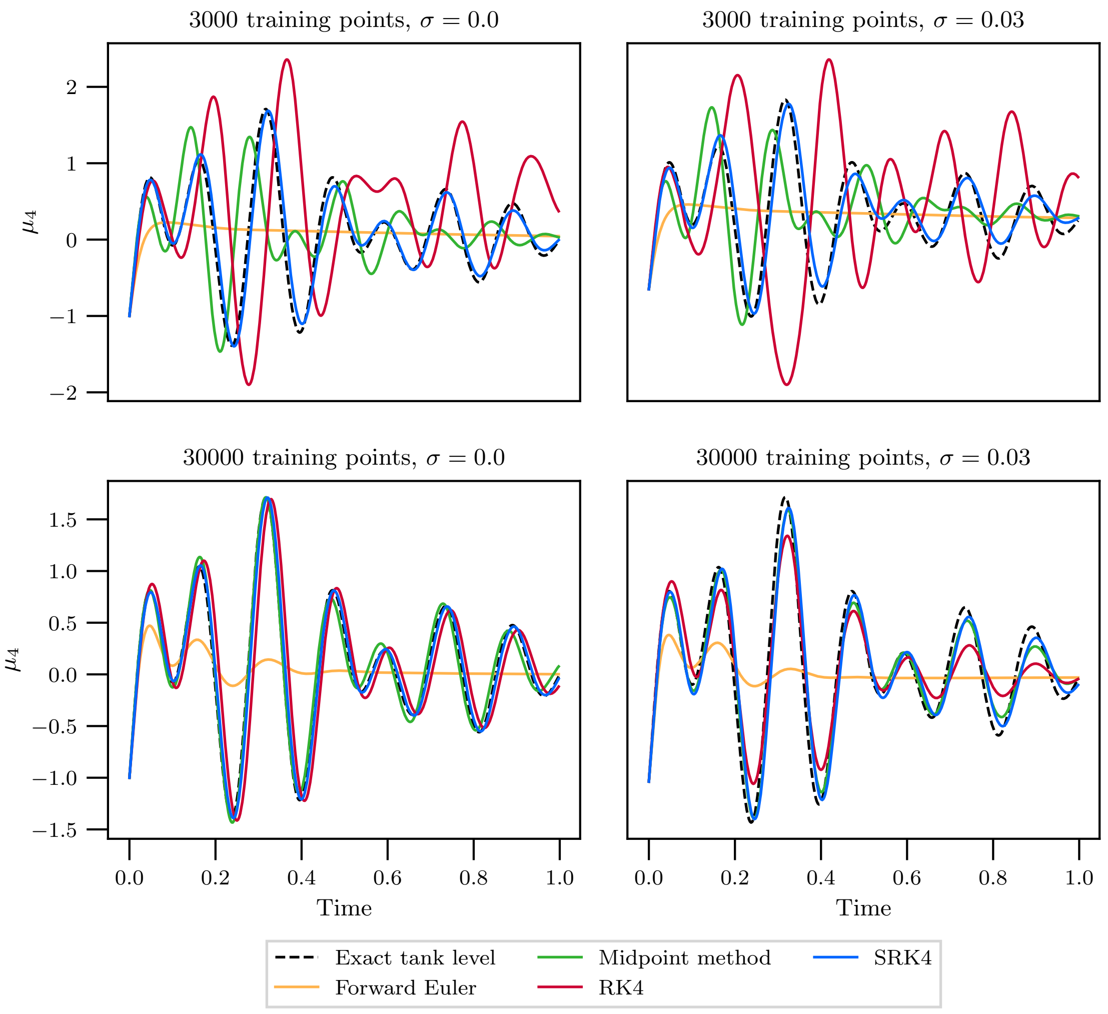

Figure 2: Volume of the fourth tank as predicted by the PHNNs with the lowest MSE on the test set. The initial condition is $\phi^0 = (-1, -1, 0, \frac{1}{2}, -1)$, $\mu^0 = (1, 1, -\frac{1}{2}, -1)$.

## 4 Performance comparison

### 4.1 Forced damped mass-spring system

Consider the damped and forced mass-spring system (4) with damping coefficient $c = 0.3$ and force term $f(t) = \sin(3t)$. We use this system to generate data sets with 1000, 2000, 5000, 10000 and 20000 samples from trajectories of length 10 with a sampling time of 1/100. We consider three model types: A baseline neural network that directly estimates the left hand side of (4) and two PHNNs that estimate the Hamiltonian $\mathcal{H}(q, p)$ and the external port $f(t)$ using two feed-forward networks, and the damping coefficient $c$ by a single learnable parameter. The first PHNN does

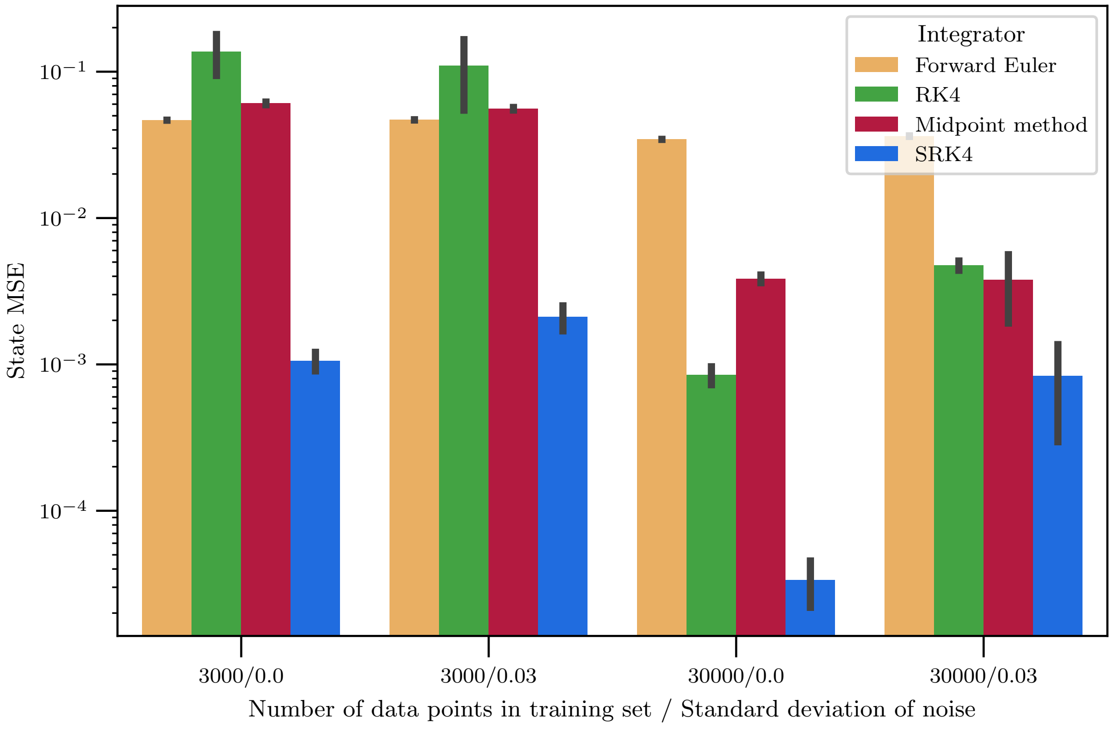

Figure 3: The mean and standard deviation of the MSE of the PHNNs trained with the different integrators on each data set. The MSE is of the predicted volume and flow in all tanks and pipes from $t = 0$ to $t = 1$ on the test set.

not assume that the external port is dependent on time alone, and thus estimates $f(t)$ using a neural network $\hat{f}_\theta(q, p, t)$, while the second PHNN assumes time dependence only and uses a neural network $\hat{f}_\theta(t)$.

For each data set we randomly initialize and train 10 models of each model type for 20000 epochs or until the validation loss has not decreased for 1000 epochs, keeping the model with the lowest MSE on the validation set. The baseline network and the networks estimating the Hamiltonian and the external port all have two hidden layers, where the first has TANH activation and the second has ReLU activation, and no activation on the output layer. The networks of the PHNNs have 100 hidden units in each layer, while the baseline model has 150 hidden units in each layer. The latter is chosen such that the models have a comparable number of free parameters. We train all models using the fourth-order integrator (13). A test set consisting of 10 trajectories of length 10 with random initial conditions is considered.

Figure 4 shows the mean and standard deviation of the MSE in the estimated $q$ and $p$ on the test set the across the 10 models of each type. The MSEs decrease with increased number of training data samples. The baseline and the PHNN model with a state-dependent port perform similarly, while the PHNN with a state-independent port has a different behaviour. With enough data, all

| Training data | 100 time steps, 300 samples | | 30 time steps, 100 samples | |
|---|---|---|---|---|
| | no noise | noise $\sigma = 0.03$ | no noise | noise $\sigma = 0.03$ |
| Euler | $11.38 \pm 3.24$ | $12.32 \pm 3.47$ | $44.23 \pm 13.10$ | $44.72 \pm 13.21$ |
| RK4 | $1.20 \pm 0.06$ | $1.98 \pm 0.31$ | $-0.16 \pm 0.58$ | $0.12 \pm 0.52$ |
| Midpoint | $1.07 \pm 0.02$ | $1.20 \pm 0.09$ | $1.70 \pm 0.19$ | $1.72 \pm 0.19$ |
| SRK4 | $1.04 \pm 0.02$ | $1.18 \pm 0.10$ | $1.23 \pm 0.06$ | $1.19 \pm 0.07$ |

Table 1: Mean and standard deviation of the predicted friction coefficients, relative to the ground truth $R = (0.03, 0.03, 0.09, 0.03, 0.03)$ (i.e. so that 1 would mean the correct coefficient).

models can learn the system with similar performance. Meanwhile, none of the models perform well when trained on a very limited amounts of data. For medium amounts of data, the PHNN with a state-independent port compensates for lacking data with information about the modelled system, leading to improved performance.

Consider the models trained on 2000, 5000 and 10000 samples, where the PHNN with a state-independent port outperforms the other models. Figure 5 shows example trajectories where the line indicates the mean prediction made by the 10 models of each type, and the shaded areas indicate the standard deviation. The superior performance of the most specified PHNN model is clear.

Figure 6 shows the exact Hamiltonian of the forced damped mass-spring system along with the Hamiltonians estimated by the two PHNNs. The solutions shown were selected among the random initializations based on the lowest MSE when estimating the gradient of the Hamiltonian. The MSE for estimating $\nabla\mathcal{H}$ is used in favour of the MSE of the $\mathcal{H}$ estimate, as the Hamiltonian estimate can be offset by a constant bias which does not affect the gradient. This bias term can lead to large offsets for $\mathcal{H}$, despite the model producing reasonable trajectories which depend on $\nabla\mathcal{H}$. Note that due to the initial condition samples used during training, $-4.5 \leq x_i \leq 4.5$.

Figure 7 compares the mean absolute error of the learned damping coefficients for the two types of PHNNs. Figure 8 shows how well the PHNNs estimate the external port as a function of training data sample size. Here, we have chosen the PHNN of each type with the lowest MSE on estimating the external port during testing.

The performance of the PHNN is drastically reduced when the port is not assumed to be independent of state. This introduces a non-uniqueness in the separation between the change in energy stemming from internal damping and the one from external ports, which complicates the learning.

## 4.2 Results from tank system

Consider the tank system described in 2.2. We use this system to generate data sets with 100, 250, 500, 1000, 2500, 5000, 10000 and 20000 samples from trajectories of length 1 with a sampling time of $\frac{1}{100}$. We consider two types of model: A baseline feed-forward network directly estimating the time derivatives of the tank levels and pipe flows, and a PHNN estimating the Hamiltonian function $\mathcal{H}(\phi, \mu)$ and the external port $f(\phi, \mu)$ with two feed-forward networks, and the damping coefficients with one learnable parameter per pipe. All networks take the current state of the system as input, and derivatives are estimated using the integrator (13).

For each data set we train 10 models of each model type for 20000 epochs or until the validation loss has not decreased for 1000 epochs, keeping the model with the lowest MSE on the validation set. The baseline network and the networks estimating the Hamiltonian and external port all have

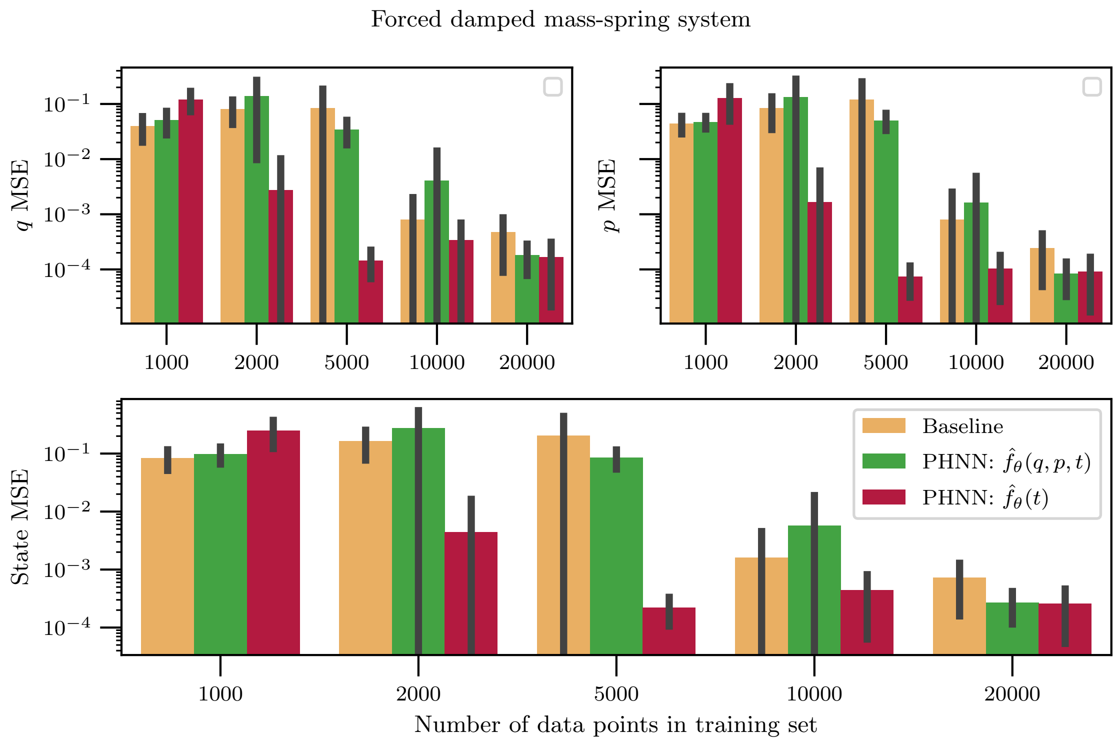

Figure 4: Mean and standard deviation of MSE of state estimates for increasing amounts of training data for the forced damped mass-spring system.

two hidden fully connected layers, where the first has TANH activation and the second has ReLU. The output layer has no activation. The networks of the PHNNs have 100 hidden units in each layer, while the baseline model has 150 hidden units in each layer. The latter is chosen such that the models have comparable numbers of free parameters.

Figure 9 shows the MSE of state estimates computed across 10 trajectories per model. The port is state-dependent and similarly to the mass-spring system, this leads to inferior performance. We attribute this to non-uniquely separable terms in 10 which do not benefit from the imposed structure.

Figure 10 shows the contours the exact and estimated Hamiltonian for slices of the parameter space. The estimates are based on the PHNN with the lowest MSE for $\nabla\mathcal{H}$ to eliminate the bias term. The exact Hamiltonian is spherical in the $\phi_4 - \phi_5$ and $\mu_1 - \mu_2$ planes, and ellipsoid in the remaining planes. The learned Hamiltonian is elongated along the same directions, but with a small offset in the $\phi_4$ and $\phi_5$ direction. The largest difference is for the $\mu_1 - \mu_2$ plane which is offset and slightly elongated.

"boundary conditions"

## 5 Learning of systems with external ports

The PHNN framework allows detection of unknown external ports, like a leakage or unknown inflow in one or more tanks. Moreover, in the case that external ports are altered, for instance by stopping a leak, a PHNN learned from data with a leak does not have to be retrained, as the PHNN's external ports can be altered correspondingly.

### 5.1 One leaking tank

Consider the example tank system described in Section 2.2, where we let

$$F_{t,4}(\phi, \mu) = -30 \min(0.3,\, \max(\mu_4, -0.3)) \tag{14}$$

model an undetected leak in the fourth tank. Training a PHNN on data generated with this leakage allows learning both the tank system dynamics and the leakage when the right constraints are imposed on the PHNN. First, we assume no knowledge of which tanks might be leaking. In order to not learn spurious solutions, $L_1$-regularization of the terms in $F_t$ is necessary. This is done by adding the following term to the MSE loss function:

$$\mathcal{L}_1 = \frac{\lambda}{N} \|\hat{F}_t(\frac{x^n + x^{n+1}}{2})\|_1,$$

where $\lambda$ is the regularization parameter weighting the penalty. This is similar to what is suggested in [11]. The upmost left plot in Figure 11 shows how a PHNN models the external port when trained using the integrator (12) for 600 epochs with a $\lambda = \{0.3, 0.1, 0.03, 0.01\}$ changing every 150th epoch, on 300 trajectories with a step size of 1/400 and end time $T = 1$. The predicted leakage from tanks 1-3 is negligible, while the leak from the fourth tank is accurately predicted. Re-running the training with the assumption that there is only a leak in the fourth tank we observe more efficient training; we obtain an accurate model after only 30 epochs of training, with no regularization.

The second row of Figure 11 shows how the learned model deteriorates when training on noisy data with a lower sampling rate. That is, we now have 1000 training trajectories but a step size of 1/100 and Gaussian noise with standard deviation $\sigma = 0.01$. Even using the integrator (13) and 2000 epochs with $\lambda = \{0.3, 0.1, 0.03, 0.01\}$ changing every 500th epoch we struggle to get an accurate model. However, as seen in the third row, the leak in tank four can be learned close to perfectly with noise if the PHNN is restricted to learn only the external port affecting that tank. As seen in the right column of Figure 11, the trained PHNNs can still be used for prediction after the leak is removed, as the external port model of the PHNN can be removed at will.

### 5.2 Two leaking tanks

Consider the same case as above, except now there is also leak in the first tank, given by

$$F_{t,1}(\phi, \mu) = -10 \min(0.3,\, \max(\mu_1, -0.3)), \tag{15}$$

in addition to the leak in the fourth tank (14). We have the same type of noisy and sparse data and use the same training parameters as for the above case. First training not knowing which tanks have leaks, we again struggle to learn the leaks accurately, as shown in the fourth row of Figure 11. When training with the assumption that the leakages are in the first and fourth tanks only, results improve, as seen in the last row of Figure 11.

## 6 Discussion

We have shown that the potential advantages of a PHNN model depend on the system and the available data, and also the assumptions one can impose on the model. For systems with time-dependent ports, we need to assume the ports to be strictly time-dependent to achieve improved learning over a baseline model. For larger systems with state-dependent ports, PHNN models need roughly the same data quality and quantity as the baseline models to reach the same accuracy. However, the advantage of using the port-Hamiltonian structure in this case lies in the explainability of the model and the possibility to detect and learn the external ports. Furthermore, we have shown how the integrator used during training significantly affect the resulting model and proposed a fourth-order symmetric method that is particularly suited for learning from sparse and noisy data.

### 6.1 Possible future research

The imposed PHNN structure allows for future exploitation including port-Hamiltonian system identification, control with PHNN models and expansion to infinite-dimensional systems.

System identification for port-Hamiltonian systems was recently proposed in the appendix of [23], but only for the case where the external ports are known. Similarly, [12] suggest a framework for using sparse regression to obtain an analytic expression for the Hamiltonian, but only for strictly preserving systems with a separable Hamiltonian. We plan to investigate system identification using the system set-up and the techniques presented in this paper, including the proposed symmetric fourth-order integrator (13).

Building on the encouraging performance of (13), we will further investigate numerical integrators especially suited for the inverse problem of learning dynamical systems. We will combine this with more advances techniques for dealing with noise, including taking into account more data points in the estimation of derivatives. This will be somewhat related to the symplectic recurrent neural networks of [7], but compatible with the more general system (3) and a wider class of integrators.

Furthermore, the PHNN model is well suited for control, as illustrated in Figure 12. In this scenario, a model is learned under the conditions described in Section 5.1 using 1000 data points, after which a controlled pipe that is constrained with respect to minimum and maximum flow is added to the first tank. Using the learned model in a model-predictive control (MPC) framework, the tank levels of the system can then be driven to desired reference levels through the new inflow pipe. We will include the possibility of using learned PHNN models for MPC in our code to be released soon on GitHub.

Lastly, port-Hamiltonian formulations also exist for infinite-dimensional systems [27, 2, 1], and hence PHNN could be developed for finite-dimensional approximations of such systems as well.

### Acknowledgments

This research was supported by the industry partners Borregaard, Elkem, Eramet, Hydro, Yara and the Research Council of Norway, through the projects BigDataMine (no. 309691) and TAPI: Towards Autonomy in Process Industries (no. 294544).

## References

[1] A. Brugnoli, G. Haine, A. Serhani, and X. Vasseur. Numerical approximation of port-Hamiltonian systems for hyperbolic or parabolic PDEs with boundary control. *Journal of Applied Mathematics and Physics*, 9:1278–1321, 2021.

[2] F. L. Cardoso-Ribeiro, A. Brugnoli, D. Matignon, and L. Lefèvre. Port-Hamiltonian modeling, discretization and feedback control of a circular water tank. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 6881–6886. IEEE, 2019.

[3] J. R. Cash and A. Singhal. Mono-implicit Runge–Kutta formulae for the numerical integration of stiff differential systems. *IMA J. Numer. Anal.*, 2(2):211–227, 1982.

[4] E. Celledoni, A. Leone, D. Murari, and B. Owren. Learning Hamiltonians of constrained mechanical systems. *arXiv preprint arXiv:2201.13254*, 2022.

[5] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[6] Y. Chen, T. Matsubara, and T. Yaguchi. Neural symplectic form: Learning Hamiltonian equations on general coordinate systems. *Advances in Neural Information Processing Systems*, 34, 2021.

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

[8] M. David and F. Méhats. Symplectic learning for Hamiltonian neural networks. *arXiv preprint arXiv:2106.11753*, 2021.

[9] C. De Persis and C. S. Kallesoe. Pressure regulation in nonlinear hydraulic networks by positive and quantized controls. *IEEE Transactions on Control Systems Technology*, 19(6):1371–1383, 2011.

[10] S. A. Desai, M. Mattheakis, and S. J. Roberts. Variational integrator graph networks for learning energy-conserving dynamical systems. *Physical Review E*, 104(3):035310, 2021.

[11] S. A. Desai, M. Mattheakis, D. Sondak, P. Protopapas, and S. J. Roberts. Port-Hamiltonian neural networks for learning explicit time-dependent dynamical systems. *Phys. Rev. E*, 104:034312, Sep 2021.

[12] D. DiPietro, S. Xiong, and B. Zhu. Sparse symplectically integrated neural networks. *Advances in Neural Information Processing Systems*, 33:6074–6085, 2020.

[13] T. Duong and N. Atanasov. Hamiltonian-based neural ODE networks on the SE(3) manifold for dynamics learning and control. In *Robotics: Science and Systems (RSS)*, 2021.

[14] T. Duong and N. Atanasov. Learning adaptive control for SE(3) Hamiltonian dynamics. *arXiv preprint arXiv:2109.09974*, 2021.

[15] M. Finzi, K. A. Wang, and A. G. Wilson. Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints. *Advances in neural information processing systems*, 33:13880–13889, 2020.

[16] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[17] M. Grmela and H. C. Öttinger. Dynamics and thermodynamics of complex fluids. I. Development of a general formalism. *Physical Review E*, 56(6):6620, 1997.

[18] E. Hairer, C. Lubich, and G. Wanner. *Geometric numerical integration*, volume 31 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2006. Structure-preserving algorithms for ordinary differential equations.

[19] W. R. Hamilton. On a general method in dynamics. *Phil. Trans. R. Soc.*, 124:247–308, 1834.

[20] P. Jin, Z. Zhang, I. G. Kevrekidis, and G. E. Karniadakis. Learning Poisson systems and trajectories of autonomous systems via Poisson neural networks. *arXiv preprint arXiv:2012.03133*, 2020.

[21] P. Jin, Z. Zhang, A. Zhu, Y. Tang, and G. E. Karniadakis. SympNets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. *Neural Networks*, 132:166–179, 2020.

[22] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv e-prints*, page arXiv:1412.6980, Dec. 2014.

[23] K. Lee, N. Trask, and P. Stinis. Structure-preserving sparse identification of nonlinear dynamics for data-driven modeling. *arXiv preprint arXiv:2109.05364*, 2021.

[24] T. Matsubara, A. Ishikawa, and T. Yaguchi. Deep energy-based modeling of discrete-time physics. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13100–13111. Curran Associates, Inc., 2020.

[25] R. I. McLachlan, G. R. W. Quispel, and N. Robidoux. Geometric integration using discrete gradients. *R. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci.*, 357(1754):1021–1045, 1999.

[26] H. C. Öttinger and M. Grmela. Dynamics and thermodynamics of complex fluids. II. Illustrations of a general formalism. *Physical Review E*, 56(6):6633, 1997.

[27] R. Pasumarthy and A. J. van der Schaft. On interconnections of infinite dimensional port-Hamiltonian systems. In *Proceedings 16th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2004)*, pages 5–9, 2004.

[28] R. Rashad, F. Califano, A. J. van der Schaft, and S. Stramigioli. Twenty years of distributed port-Hamiltonian systems: a literature review. *IMA J. Math. Control Inform.*, 37(4):1400–1422, 2020.

[29] A. Sanchez-Gonzalez, V. Bapst, K. Cranmer, and P. Battaglia. Hamiltonian graph networks with ODE integrators. *arXiv preprint arXiv:1909.12790*, 2019.

[30] W. M. G. van Bokhoven. Efficient higher order implicit one-step methods for integration of stiff differential equations. *BIT*, 20(1):34–43, 1980.

[31] A. Van Der Schaft. Port-Hamiltonian systems: an introductory survey. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1339–1365. Citeseer, 2006.

[32] A. Van Der Schaft and D. Jeltsema. Port-Hamiltonian systems theory: An introductory overview. *Foundations and Trends in Systems and Control*, 1(2-3):173–378, 2014.

[33] A. J. van der Schaft and B. M. Maschke. Port-Hamiltonian systems on graphs. *SIAM J. Control Optim.*, 51(2):906–937, 2013.

[34] Z. Zhang, Y. Shin, and G. E. Karniadakis. GFINNs: GENERIC formalism informed neural networks for deterministic and stochastic dynamical systems. *arXiv preprint arXiv:2109.00092*, 2021.

[35] Y. D. Zhong, B. Dey, and A. Chakraborty. Dissipative SymODEN: Encoding Hamiltonian dynamics with dissipation and control into deep learning. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

[36] Y. D. Zhong, B. Dey, and A. Chakraborty. Symplectic ODE-Net: Learning Hamiltonian dynamics with control. In *International Conference on Learning Representations*, 2020.

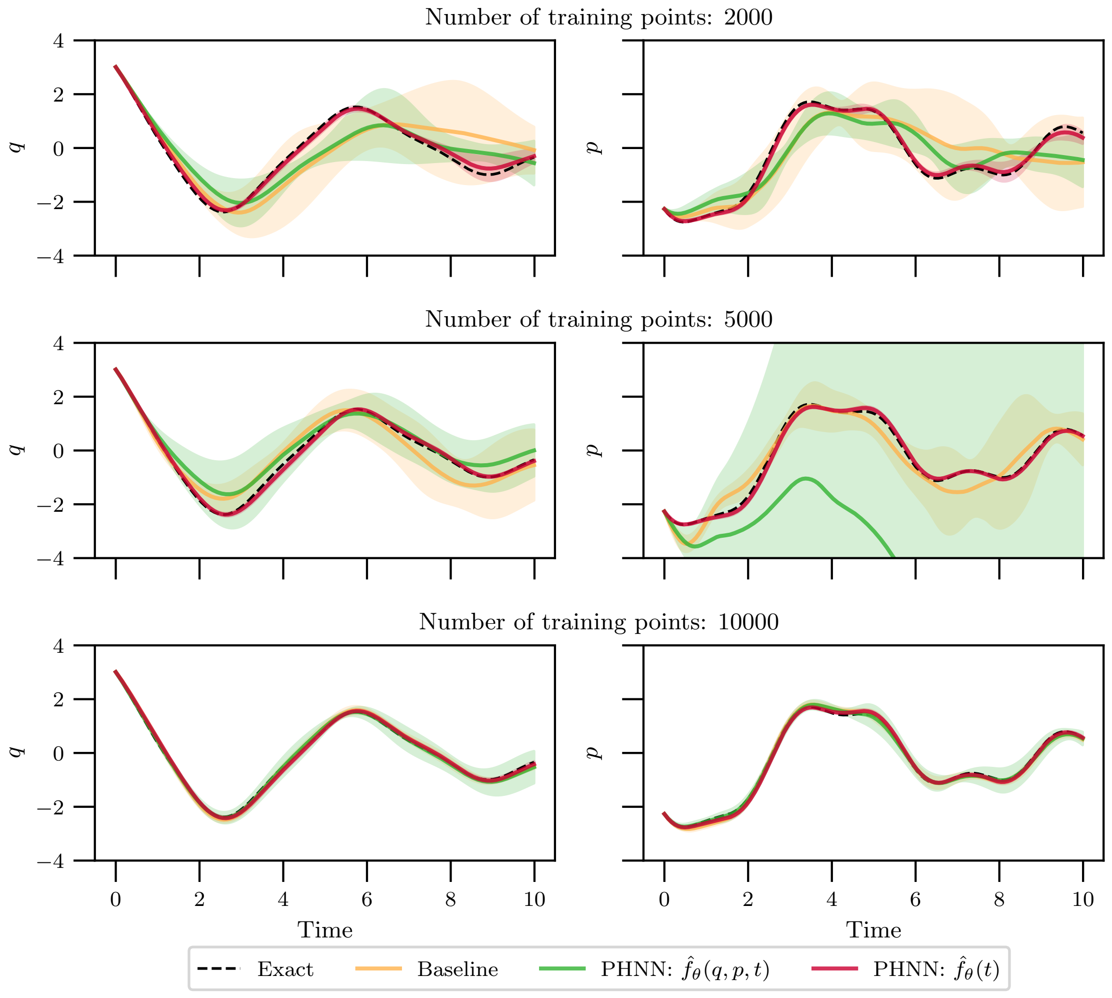

Figure 5: Example trajectory with the mean of the estimates made by the models trained on 2000, 5000 and 10000 data points for the forced damped mass-spring system. Error bands indicate standard deviation.

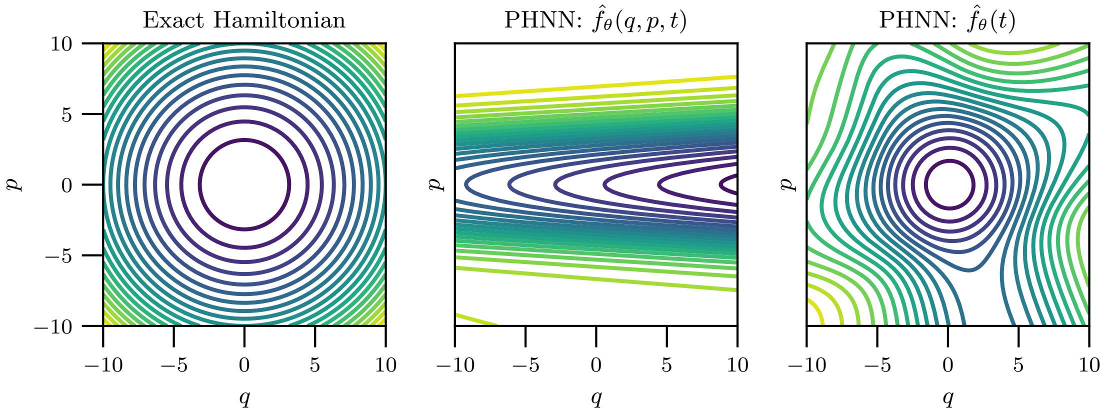

Figure 6: Contour plot of the exact Hamiltonian of the forced damped mass-spring system (left) along with the Hamiltonians estimated by two of the randomly initialised PHNNs (middle and right).

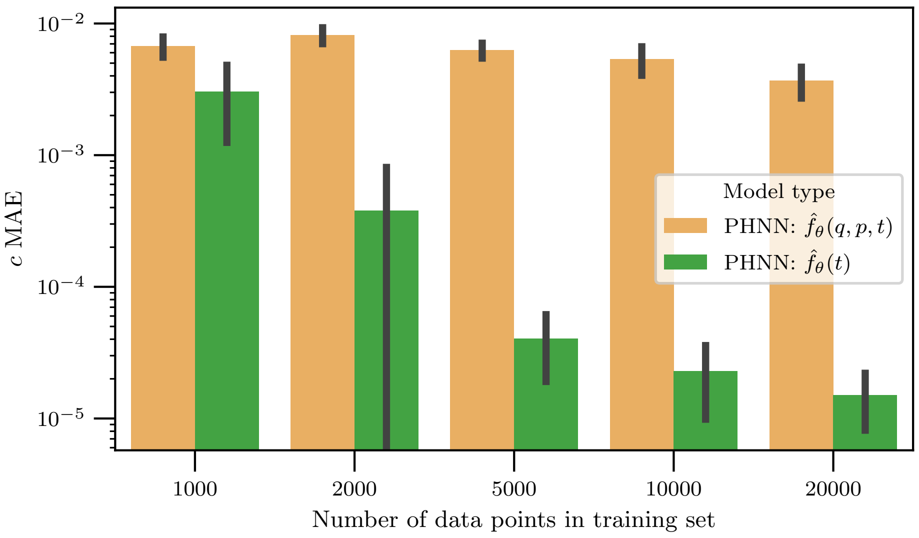

Figure 7: Mean and standard deviation of the mean absolute error in the estimates of the damping coefficient $c$ for the forced damped mass-spring system.

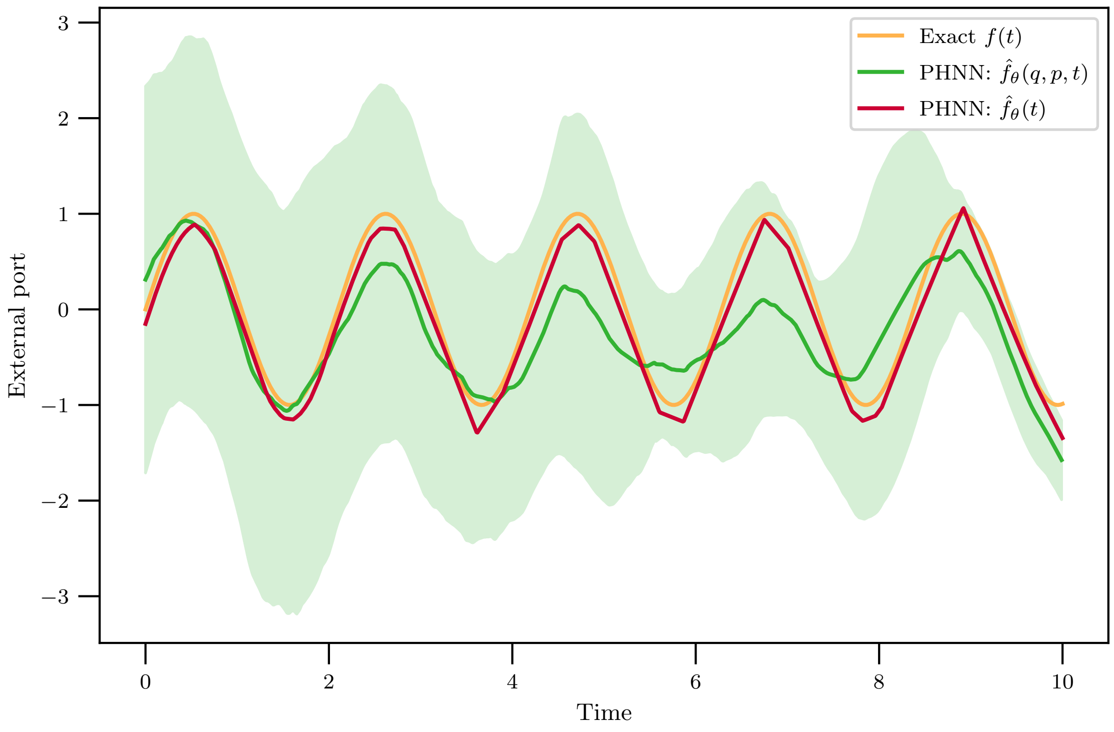

Figure 8: External port signal estimated by the two PHNN model types.

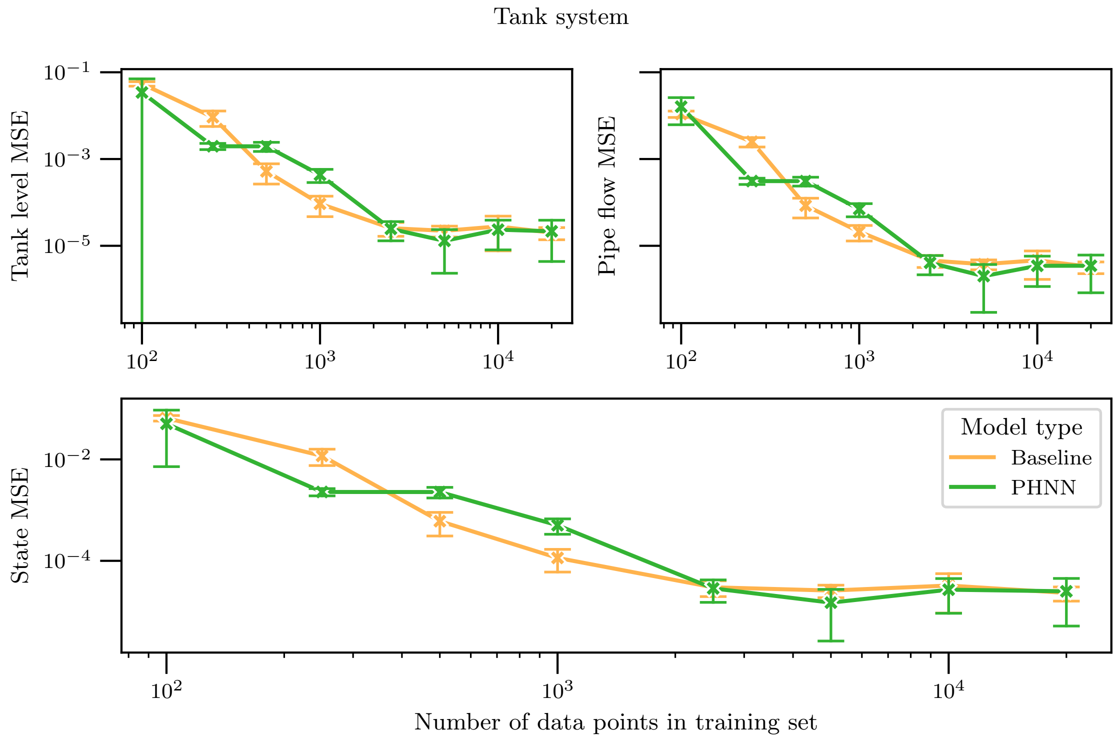

Figure 9: Mean and standard deviation of the MSE of state estimates for increasing amounts of training data for the tank system.

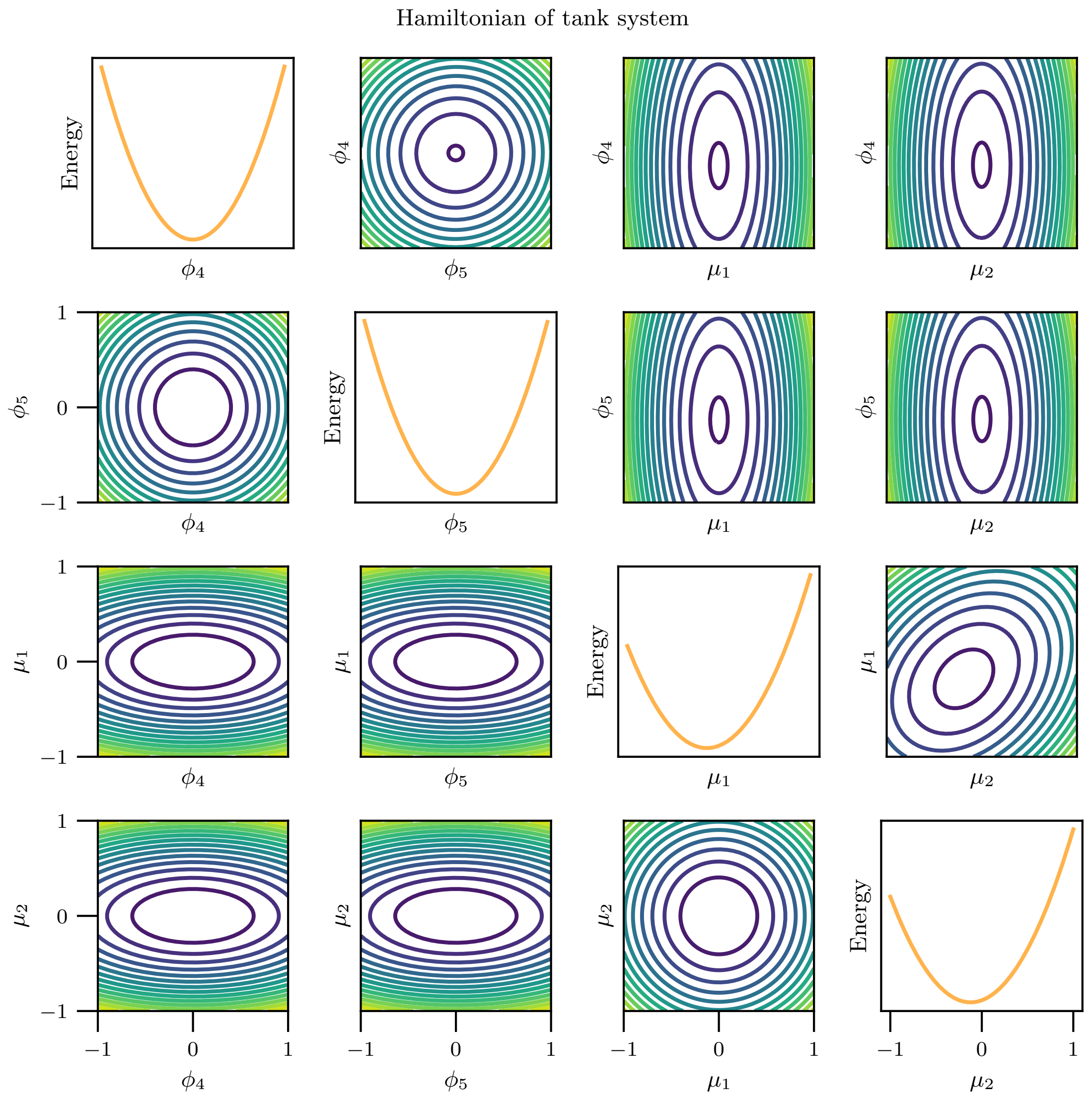

Figure 10: The lower left plots show the contour of the exact Hamiltonian when all states are set to zero except for the two displayed in each plot. The upper right shows the contour of the Hamiltonian as estimated by the PHNN with the lowest MSE in estimating $\nabla\mathcal{H}$. The plots on the diagonal show the estimated Hamiltonian when all states are set to zero except the one noted on the $x$-axis.

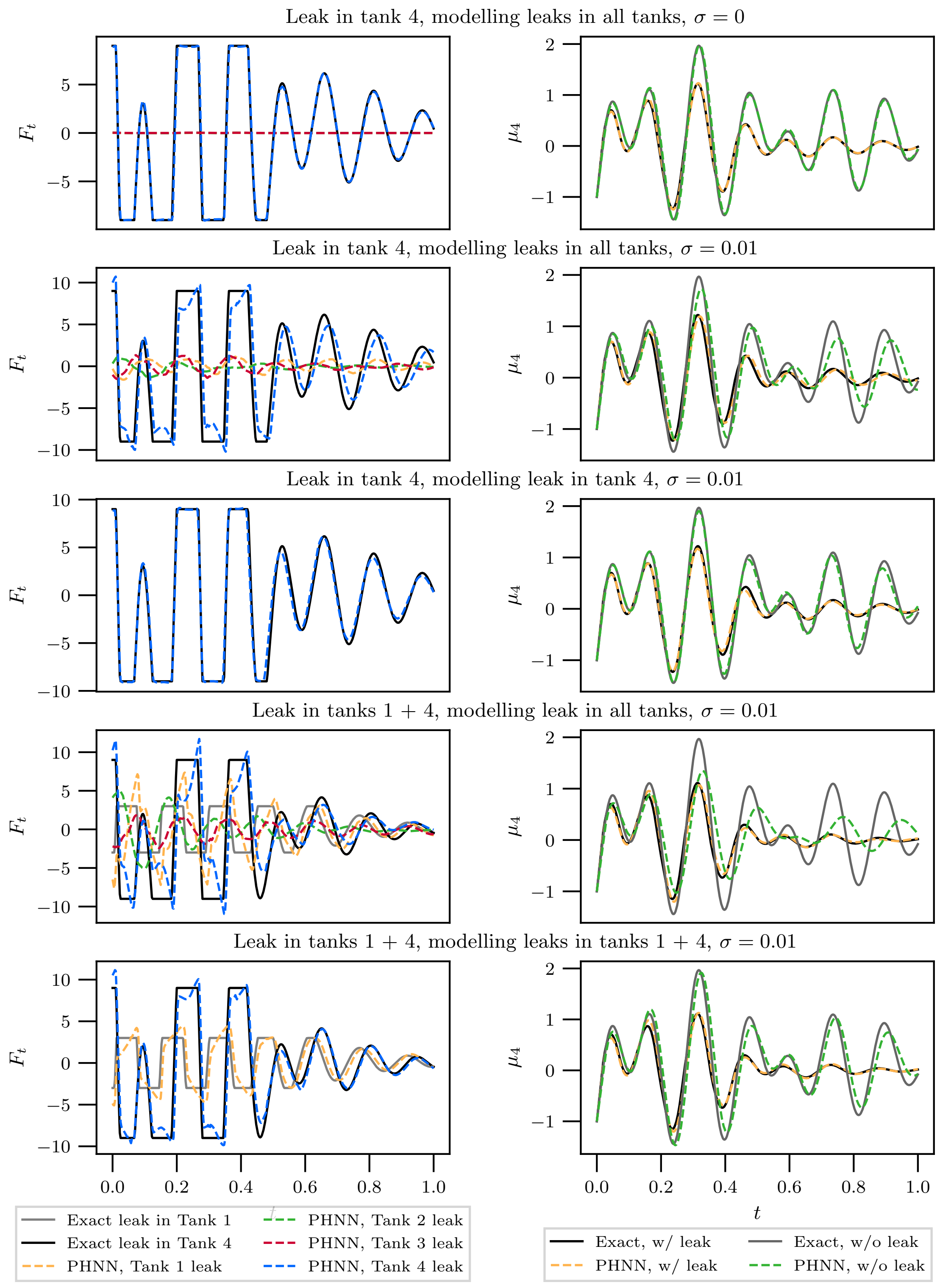

Figure 11: Left column: The external ports learned by the PHNNs trained on data gathered with one or two leakages, assuming that the location of the leakages are either known or unknown. Right column: The level in the fourth tank estimated by the PHNN before and after the leak is stopped and the PHNN external port is set to zero. Initial condition: $\phi^0 = (-1, -1, 0, \frac{1}{2}, -1)$, $\mu^0 = (1, 1, -\frac{1}{2}, -1)$.

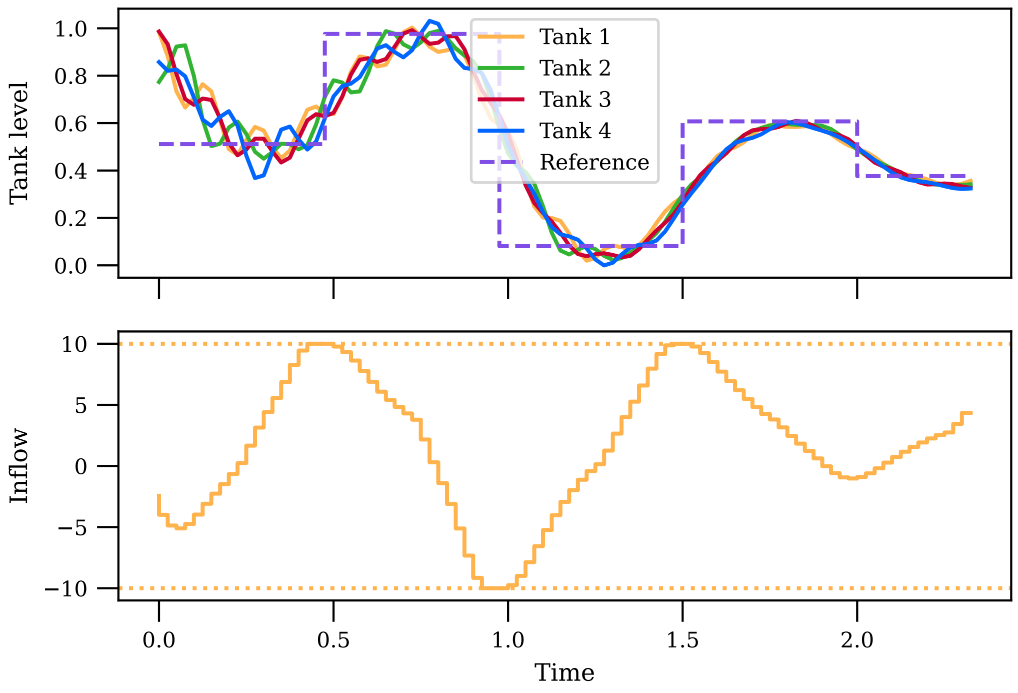

Figure 12: A learned PHNN model is used in an MPC framework, which is successfully able to drive the tank levels to the desired reference levels.