International Journal of Control

ISSN: 0020-7179 (Print) 1366-5820 (Online) Journal homepage: www.tandfonline.com/journals/tcon20

Regularised neural network-based nonlinear system identification with prior system knowledge

Daniel Frank, Tobias Holicki, Carsten W. Scherer & Steffen Staab

To cite this article: Daniel Frank, Tobias Holicki, Carsten W. Scherer & Steffen Staab (03 Jun 2026): Regularised neural network-based nonlinear system identification with prior system knowledge, International Journal of Control, DOI: 10.1080/00207179.2026.2679228

To link to this article: https://doi.org/10.1080/00207179.2026.2679228

INTERNATIONAL JOURNAL OF CONTROL
https://doi.org/10.1080/00207179.2026.2679228

Regularised neural network-based nonlinear system identification with prior system knowledge

Daniel Frank[a], Tobias Holicki[b], Carsten W. Scherer[b] and Steffen Staab[a,c]

[a]Institute for Artificial Intelligence, University of Stuttgart, Stuttgart, Germany; [b]Institute of Mathematical Methods in Engineering, Numerical Analysis and Geometric Modeling, University of Stuttgart, Stuttgart, Germany; [c]Electronics and Computer Science, University of Southampton, Southampton, UK

ABSTRACT

Neural networks can learn the behaviour of nonlinear dynamical systems and achieve high prediction accuracy on test data that is drawn from the same distribution as the training data. However, these models fail to generalise during inference when excited with unseen input trajectories. We tackle this generalisation problem in the context of nonlinear system identification with a system-theoretical approach that ensures input–output stability. We enhance a linear approximation with a recurrent neural network (RNN) that models the residual behaviour to capture complex dynamics and to increase the model's generalisation capabilities. We impose constraints on the learnable parameters to ensure dissipativity, an intrinsic property of most physical systems. This leads to improved generalisation on previously unseen inputs. We evaluate our approach using in-distribution (ID) and out-of-distribution (OOD) data from three different use cases and compare it against non-regularised approaches.

ARTICLE HISTORY
Received 21 February 2025
Accepted 29 March 2026

KEYWORDS
Nonlinear system identification; recurrent neural network; ℓ_2-stability; dissipativity

1. Introduction

When learning nonlinear dynamics from input–output measurements, models with a large number of parameters, such as neural network architectures, have shown high prediction accuracy (Andersson et al., 2019; Mohajerin & Waslander, 2019; Pillonetto et al., 2025). Despite their flexibility and success in modeling complex dynamics, neural network predictions can be unreliable when excited with input distributions not present in the training dataset.

This reliability is crucial for modeling physical systems. For example, when the trained model is used for simulation, it is required to predict unseen inputs with high accuracy (Brunton et al., 2021; Karpatne et al., 2022; Lusch et al., 2018).

Identifying dynamic models of physical systems is challenging because of the nonlinear nature of the unknown system and the requirement for reliable predictions. Linear models can efficiently be analysed and trained, but do not capture the nonlinear dynamics sufficiently well. Neural networks can learn complex dynamics, but ensuring reliability is difficult due to the nonlinear structure combined with the large number of parameters. We aim to increase reliability by ensuring input–output stability.

Different stability properties, such as incremental input–output stability (Revay et al., 2020), input-to-state stability (ISS) (Bonassi et al., 2022), or exponential stability (Baier et al., 2023), have been encoded into neural network architectures. To enforce these stability conditions, either the learnable parameters are constrained by a regularisation in the loss function or enforced through structural properties of the model itself. These models either do not achieve the same level of accuracy as their unconstrained counterpart or they do so but require more training data or training epochs. To improve the predictive accuracy of neural networks, hybrid models that use prior system knowledge in the form of a linear approximation, Forgione and Piga (2020) and Wu et al. (2020) have been introduced, but they do not provide methods to enforce stability, thus making the prediction unreliable.

Our approach assumes that the unknown underlying system is dissipative and encodes this prior knowledge in the model structure through parametric constraints. When the identified model is dissipative (Willems, 1972), it cannot generate energy internally and is driven solely by the supplied energy of the input and its initial energy. This property ensures the reliability of the predictions beyond the training data. In addition, we assume to

CONTACT Daniel Frank daniel.frank@ki.uni-stuttgart.de Universitätsstraße 32 70569 Stuttgart, Germany

have access to a linear approximation model to enhance predictive accuracy. The constraints we develop are convex with respect to the learnable parameters and are shown to improve the training process compared to their non-convex counterparts. Based on the previous assumptions, we build a hybrid model architecture (`HybCRnn`) for nonlinear system identification that

(S1) interconnects a linear approximation model with an RNN and
(S2) ensures ℓ_2-stability through regularisation.

By exploiting a richer set of multipliers (Fetzer & Scherer, 2017), our method yields tighter constraints than the multipliers typically used in the machine-learning literature. The multipliers are introduced when bounding the nonlinearities of the neural network and allow for additional degrees of freedom during optimisation.

We compare `HybCRnn` against four approaches: Two regularised baselines, a RNN that ensures ℓ_2-stability, which is based on Revay et al. (2020), and a constrained long-short term memory (LSTM) that ensures ISS (Bonassi et al., 2020). Furthermore, we compare against a linear switching system Recurrent Linear Parameter Varying Network (ReLiNet) that is exponentially stable (Baier et al., 2023) and an unconstrained LSTM (Mohajerin & Waslander, 2019). In our experiments, we find that adding a linear approximation model leads to models that generalise better to inputs not seen during training. Our `HybCRnn` outperforms all baseline models on OOD datasets and reaches comparable prediction accuracy on ID test data for single-input-single-output (SISO) systems. Additionally, we show an improved upper bound on the ℓ_2-gain when analysing the stability of the trained models. We also see that by using convex constraints, the training converges, while this is not the case for non-convex constraints. Our contributions can be summarised as follows:

- We develop a hybrid architecture for nonlinear system identification with convex constraints that ensure dissipativity through regularisation.
- We introduce tighter constraints by using a larger class of multipliers.
- We present comprehensive experiments comparing our method with four baselines on three datasets.

In Section 2, we position our work within the current literature. Section 3 introduces our hybrid network structure and defines the problem. Section 4 relates various stability properties and provides an overview of the key dissipativity results relevant to this work. The analysis of our hybrid architecture is detailed in Section 5. Section 6 outlines our training and initialisation process. In Section 7, we evaluate our approach in terms of accuracy on an ID test set and various OOD datasets, as well as assess the empirical upper bound of the ℓ_2-gain. In Section 8, we empirically evaluate the effect of the partial convexification for the learning problem from Section 5. The paper concludes in Section 9.

2. Related work

This work lies at the intersection between machine learning and system theory. We will apply established tools from robust control on recurrent neural network architectures. In this section, our goal is to review the related work from both domains. While the success of neural networks is only about a decade old, the system identification literature has a long history in the field of system theory.

2.1 Constrained neural networks

The connection between discrete linear, time-invariant (LTI) systems with static nonlinearities and neural networks was drawn by an early work of Suykens et al. (1995). It allows us to use the concept of dissipativity (Willems, 1972) for systematically analysing the input–output behaviour of neural networks. This was recently used in numerous neural network architectures, e.g. for efficient calculation of a Lipschitz gain (Fazlyab et al., 2019; Pauli, Koch, et al., 2021), system identification with stability guarantees (Revay et al., 2020, 2021), and controller synthesis (Gu et al., 2022; Junnarkar et al., 2022, 2024). These approaches are based on formulating convex constraints on the learnable parameters, which are enforced during training. Convexity of the constraints ensures that the training remains efficient. The neural network architectures that guarantee input–output properties do not consider prior system knowledge of the true system. In Forgione and Piga (2020) and Wu et al. (2020) hybrid architectures assume a linear approximation model to be known and use neural networks to learn the residual, to improve predictive accuracy. By adding the linear approximation model, the convexity of the constraints gets lost due to the interconnection between the linear part of the neural network and the linear approximation model. For the problem of neural network controller design and closed-loop performance guarantees, the authors of Junnarkar et al. (2024) have derived convex constraints. In this work, we follow the idea of using a linear approximation model (Forgione & Piga, 2020; Wu et al., 2020) and let the neural network

architecture learn the residual. In addition, we derive convex constraints on the learnable parameters that are based on the idea of controller synthesis (Gu et al., 2022; Junnarkar et al., 2022). The constraints are formulated as linear matrix inequalities (LMI) and can be enforced during training.

While these tools have only recently been used in machine learning, they are established in robust control (Masubuchi et al., 1998; Scherer et al., 1997). During training, we ensure the constraints by incorporating logarithmic barrier functions as penalty terms in the loss function. A direct parameterisation approach, as presented by Revay et al. (2021) and Wang and Manchester (2023), requires no constraints during training, as structural properties ensure the parametric constraints. However, we do not adopt this direct parameterisation since our goal is to conceptually show that adding prior system knowledge leads to more reliable simulation models. We will consider other parameterisation methods in future work.

The constraints rely on a multiplier relaxation of the nonlinear activation function that introduces flexibility in the optimisation problem. We use a larger class of multipliers compared to the before-mentioned approaches that only consider diagonal multipliers (except for Pauli, Gramlich, et al., 2021a). We extend this multiplier class by employing static Zames–Falb multipliers as presented by Fetzer and Scherer (2017) for discrete-time systems, which involve more degrees of freedom than the simple diagonal multipliers, and evaluate their effect on the prediction accuracy and stability.

Instead of deriving parametric constraints for internal stability or input–output stability, the work of Bonassi et al. (2020, 2022) and D'Amico et al. (2023) provides constraints that ensure ISS and incremental ISS. The networks considered include various RNN model structures such as LSTM or gated recurrent unit (GRU). The constraints derived in these models are not based on dissipativity but on linking the parameters to conditions on Lyapunov functions that are required to ensure ISS and incremental ISS.

Identification algorithms that consider neural networks as state-space models are considered in Beintema et al. (2023), J. Schoukens and Ljung (2019), and Forgione et al. (2022). In Beintema et al. (2023), a subspace encoder is learned to learn the initial state of a state-space neural network. Even though the model structure allows the analysis of its stability, this is not part of the paper. The parametric constraints we develop in this work can be applied to other state-space neural networks.

2.2 Regularised system identification

The established field of system identification in system theory and control is dominated by linear identification methods (Ljung, 1998; Pillonetto et al., 2022). Recently, Ljung et al. (2020) highlighted the importance of regularisation in enhancing the least squares regression problem. Regularisation allows the incorporation of prior system knowledge into the identification process. In Khosravi and Smith (2023), the authors introduced a kernel regularisation method that ensures a predefined ℓ_2-gain. In these identification methods, the unknown system is described by estimating the impulse response. In contrast, our neural network architecture directly learns the nonlinear system's dynamics. However, we train our neural network with the same kind of prior system knowledge (S2).

2.3 Deep learning for system identification

RNN structures have demonstrated high prediction accuracy in identifying the dynamics of nonlinear systems (Andersson et al., 2019; Hu et al., 2024; Mohajerin & Waslander, 2019). Instead of directly predicting output sequences, Baier et al. (2023) employed an LSTM to predict the parameters of a linear system at each time step, which resulted in a switched linear system. By constructing the linear system matrices to be Hurwitz and pairwise commutative, exponential stability can be guaranteed.

Due to the absence of state information and the unknown relationship between the physical state and the neural network's hidden state, initialisation becomes an integral part of system identification with neural networks (Mohajerin & Waslander, 2019; M. Schoukens, 2021). In our experiments, we will use *washout* initialisation for the constrained methods and *NN-based* initialisation for the non-constrained models.

2.4 Notation

In this work, we consider square summable sequences in the space $\ell_2^n := \{(w^k)_{k\in\mathbb{N}_0} \mid w^k \in \mathbb{R}^n \text{and} \|w\|^2 = \sum_{k=0}^{\infty}(w^k)^\top w^k < \infty\}$. The set of diagonal positive definite $n \times n$ matrices is denoted by $\mathbb{D}_+^n$, the set of symmetric, positive definite $n \times n$ matrices is denoted by $\mathbb{S}_+^n$ and the set of symmetric $n \times n$ matrices by $\mathbb{S}^n$. The notation $A \succ 0$ ($A \prec 0$) means that the matrix A is positive definite (negative definite). We use $(\star)$ for matrices or vectors that can be obtained from symmetry. In data preprocessing we use the floor function which is defined as $\lfloor x \rfloor := \max\{m \in \mathbb{Z} \mid m \leq x\}$. We denote the maximum of the ℓ_2 norm as $\|w\|_{2,\infty} = \max_{k\in\mathbb{N}_0} \|w^k\|_2$.

3. Problem statement

We consider an unknown nonlinear, time-invariant dynamical system driven by an external input u, where $u^k \in \mathbb{R}^{n_u}$ and with the description

$$\begin{aligned} x^{k+1} &= f(x_u^k, x^k, u^k) \\ x_u^{k+1} &= g(x_u^k, x^k, u^k) \\ y^k &= h(x_u^k, x^k, u^k) \end{aligned} \tag{1}$$

for $k \in \mathbb{N}_0$, internal states $x^k \in \mathbb{R}^{n_x}$, $x_u^k \in \mathbb{R}^{n_{x_u}}$, and some output y, with $y^k \in \mathbb{R}^{n_y}$.

We aim to find a precise approximation of such a system based on a set of available input–output measurements in order to simulate its response to new inputs or to design controllers.

We assume that prior knowledge (S1) about the system (1) is available in terms of a linearisation of f and h. For simplicity, we linearise around the origin and suppose that $f(0,0,0) = 0$ and $h(0,0,0) = 0$. Then, the system (1) can be written as

$$\begin{aligned} x^{k+1} &= A_{\text{lin}} x^k + B_{\text{lin}} u^k + \tilde{B}_{\text{lin},2} x_u^k + \tilde{B}_{\text{lin},3} N_f(x_u^k, x^k, u^k) \\ x_u^{k+1} &= g(x_u^k, x^k, u^k) \\ y^k &= C_{\text{lin}} x^k + D_{\text{lin}} u^k + \tilde{D}_{\text{lin},2} x_u^k + \tilde{D}_{\text{lin},3} N_h(x_u^k, x^k, u^k) \end{aligned} \tag{2}$$

for known matrices $A_{\text{lin}}, B_{\text{lin}}, \tilde{B}_{\text{lin},2}, \tilde{B}_{\text{lin},3}, C_{\text{lin}}, D_{\text{lin}}, \tilde{D}_{\text{lin},2}, \tilde{D}_{\text{lin},3}$ and unknown nonlinear maps N_f, N_h and g.

By introducing auxiliary signals

$$u_r^k := \begin{pmatrix} x_u^k \\ N_f(x_u^k, y^k) \\ N_h(x_u^k, y^k) \end{pmatrix} \quad \text{and} \quad y_r^k := \begin{pmatrix} y_1^k \\ y_2^k \end{pmatrix} := \begin{pmatrix} x^k \\ u^k \end{pmatrix}, \tag{3}$$

the system (2) can be expressed as the feedback interconnection of the known linear system

$$\mathcal{H}_{\text{lin}} := \left\{ \left(\begin{array}{c} x^{k+1} \\ \hline y^k \\ y_r^k \end{array} \right) = \left(\begin{array}{c|cc} A_{\text{lin}} & B_{\text{lin}} & B_{\text{lin},2} \\ \hline C_{\text{lin}} & D_{\text{lin}} & D_{\text{lin},2} \\ I & 0 & 0 \\ 0 & I & 0 \end{array} \right) \begin{pmatrix} x^k \\ u^k \\ u_r^k \end{pmatrix} \right. \tag{4}$$

with the unknown nonlinear system

$$\begin{pmatrix} x_u^{k+1} \\ \hline u_r^k \end{pmatrix} = \left(\begin{array}{c} g(y_1^k, x_u^k, y_2^k) \\ \hline x_u^k \\ N_f(y_1^k, x_u^k, y_2^k) \\ N_h(y_1^k, x_u^k, y_2^k) \end{array} \right); \tag{5}$$

here, $B_{\text{lin},2} := (\tilde{B}_{\text{lin},2}, \tilde{B}_{\text{lin},3}, 0)$ and $D_{\text{lin},2} = (\tilde{D}_{\text{lin},2}, 0, \tilde{D}_{\text{lin},3})$.

At this point, we follow Forgione and Piga (2020), Wu et al. (2020) and approximate the residual by a neural network, more precisely we use a dynamic RNN of the form

$$\mathcal{R}_\theta := \left\{ \mathcal{R}_{\text{lin}} := \left\{ \begin{pmatrix} x_{\text{rnn}}^{k+1} \\ u_r^k \\ z^k \end{pmatrix} = \begin{pmatrix} A & B & B_2 \\ C & D & D_{12} \\ C_2 & D_{21} & 0 \end{pmatrix} \begin{pmatrix} x_{\text{rnn}}^k \\ y_r^k \\ w^k \end{pmatrix} \right. \right. \\ \left. \left. w^k = \Delta(z^k). \right. \right. \tag{6}$$

Where the unknown state of the residual model x_u is represented by the internal state of the RNN $x_{\text{rnn}} \in \mathbb{R}^{n_x}$.

Remark 3.1: We can recover the structure of model (3) in Forgione and Piga (2020) by choosing $\tilde{B}_{\text{lin},3} = \tilde{D}_{\text{lin},3} = I$ and $\tilde{B}_{\text{lin},2} = \tilde{D}_{\text{lin},2} = \tilde{D}_{\text{lin},2} = 0$. Our hybrid model structure can therefore be seen as a generalisation of the model in Forgione and Piga (2020) that is able to directly integrate the internal state of the residual model (See Section 3.1 for more details on the interconnection between the linear approximation model and the residual model.)

This model (6) constitutes an interconnection of a linear system with a static nonlinearity $\Delta : \mathbb{R}^{n_z} \to \mathbb{R}^{n_w}$ and includes numerous frequently employed model structures. With static, we refer to the class of nonlinear functions that is time-invariant and does not depend on previous inputs. In particular, the RNN in Goodfellow et al. (2016) is recovered by choosing $A = B = C = D = 0$, $B_2 = I$ and $\Delta(z^k) = (\tanh(z_1^k), \ldots, \tanh(z_{n_z}^k))^\top$.

The trainable parameters in (6) are the describing matrices of the RNN's linear part, i.e.

$$\theta = (A, B, B_2, C, C_2, D, D_{12}, D_{21}). \tag{7}$$

We will train these parameters based on a dataset of input–output measurements of the unknown original system (1) with the aim that this system is well-approximated by the interconnection of $\mathcal{H}_{\text{lin}}$ in (4) and the RNN $\mathcal{R}_\theta$ in (6). The latter interconnection is depicted in Figure 1.

It is expected and confirmed by our experiments that making use of the known linear part and merely learning the unknown nonlinear part in (5) using an RNN is to be preferred over directly learning the entire system (1).

We can interpret $\mathcal{R}_\theta$ as a residual model that accounts for the mismatch between the output measurements of (1) and the linear approximation model. Next to small training and test errors in the simulation, we also aim to

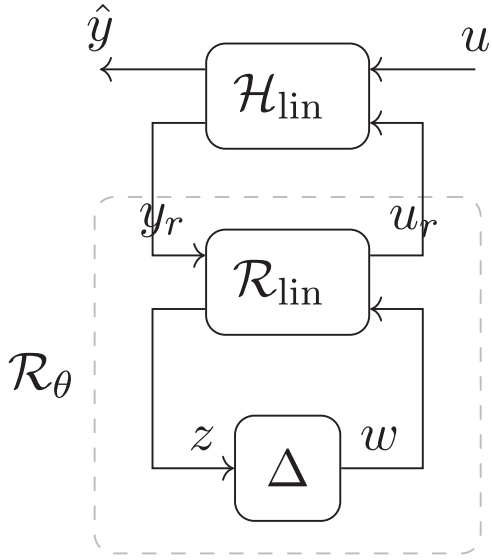

Figure 1. LTI model in feedback interconnection with a nonlinear residual model.

preserve dissipativity of the physical system in our resulting model. Therefore, we make the following assumption on the system that we aim to identify:

Assumption 3.1: *The unknown system (1) is dissipative with respect to a quadratic supply rate that ensures the worst-case amplification is bounded by $\gamma > 0$ and we assume to know the value of γ.*

The parametrised hybrid model (`HybCRnn`) reads as

$$\mathcal{S}_\theta := \left\{ \begin{pmatrix} \xi^{k+1} \\ \hat{y}^k \\ z^k \end{pmatrix} = \begin{pmatrix} \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix} \begin{pmatrix} \xi^k \\ u^k \\ w^k \end{pmatrix}, \; w^k = \Delta(z^k) \right. \tag{8}$$

with internal state $\xi = \begin{pmatrix} x \\ x_{\text{rnn}} \end{pmatrix}$, where $\xi^k \in \mathbb{R}^{2n_x}$, external input u where $u^k \in \mathbb{R}^{n_u}$, predicted output $\hat{y}$, where $\hat{y}^k \in \mathbb{R}^{n_y}$, static nonlinearity $\Delta(z^k) = (\tanh(z_1^k), \ldots, \tanh(z_{n_z}^k))^\top$ and the matrices

$$\left(\begin{array}{c|cc} \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \\ \hline \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{array}\right) = \left(\begin{array}{cc|cc} A_{\text{lin}} + B_{\text{lin},2} D \begin{pmatrix} I \\ 0 \end{pmatrix} & B_{\text{lin},2} C & B_{\text{lin}} + B_{\text{lin},2} D \begin{pmatrix} 0 \\ I \end{pmatrix} & B_{\text{lin},2} D_{12} \\ B \begin{pmatrix} I \\ 0 \end{pmatrix} & A & B \begin{pmatrix} 0 \\ I \end{pmatrix} & B_2 \\ \hline C_{\text{lin}} + D_{\text{lin},2} D \begin{pmatrix} I \\ 0 \end{pmatrix} & D_{\text{lin},2} C & D_{\text{lin}} + D_{\text{lin},2} D \begin{pmatrix} 0 \\ I \end{pmatrix} & D_{\text{lin},2} D_{12} \\ D_{21} \begin{pmatrix} I \\ 0 \end{pmatrix} & C_2 & D_{21} \begin{pmatrix} 0 \\ I \end{pmatrix} & 0 \end{array}\right). \tag{9}$$

We denote the number of input and output channels (the size of vectors u^k and $\hat{y}^k$) by n_u and n_y, respectively. The model $\mathcal{S}_\theta$ maps an input trajectory $u \in \ell_2^{n_u}$ and an initial state ξ^0 to an output trajectory $\hat{y}$. Our goal is to learn a model that makes reliable predictions, thus we aim for $\hat{y} \in \ell_2$.

Remark 3.2: In the work of D'Amico et al. (2023), simple-to-verify LMI conditions are derived to ensure incremental input-to-state stability, called δISS for a class of RNN models. Their model class allows incorporation of prior system information via a linear approximation. Moreover, they show that feedback interconnections of multiple instances of their models remain within the same class. For example, in a feedback configuration where one model represents a dynamical system and the other a controller, they derive conditions on the controller parameters such that the closed loop is δISS.

Their proposed model class differs from ours in (i) the overall objective of the interconnection and (ii) the class of supported controllers for which the derived LMI conditions hold.

The first distinction (i) concerns the intended use: their architecture is designed to provide closed-loop guarantees and to synthesise controller parameters for a given plant model. For instance, given a model of a dynamical system in their framework, the goal is to determine a controller such that the interconnected system is δISS. This contrasts with our iterative training approach, where the interconnected model is optimised to achieve high prediction accuracy on a set of training trajectories.

To illustrate that the model in D'Amico et al. (2023) is more restrictive (ii), we rewrite their proposed architecture (Equation (4) in the paper) in the form of our hybrid model (9), yielding

$$\left(\begin{array}{c} x_{\text{lin}}^{k+1} \\ x_{nl}^{k+1} \\ \hline y^k \\ z^k \end{array}\right) = \left(\begin{array}{cc|ccc} A_{\text{lin}} & 0 & B_{\text{lin}} & 0 & 0 \\ 0 & 0 & 0 & 0 & I \\ \hline C_{\text{lin}} & C_{nl} & D_{\text{lin}} & D_{nl} & 0 \\ 0 & A_{nl} & 0 & B_{nl} & 0 \end{array}\right) \left(\begin{array}{c} x_{\text{lin}}^k \\ x_{nl}^k \\ \hline u_{\text{lin}}^k \\ u_{nl}^k \\ w^k \end{array}\right), \tag{10}$$

$$w^k = \tilde{f}(z^k).$$

Here, $\tilde{f} = [f_1, \ldots, f_{n_w}]^\top$ is a static nonlinearity with each $\tilde{f}_i : \mathbb{R} \to \mathbb{R}$ being Lipschitz continuous with constant L_{pi}. Since feedback interconnections of two such models remain within the same class, we can directly compare (10) with our hybrid architecture (9). Both architectures can incorporate the linear matrices A_{lin}, B_{lin}, C_{lin}, and D_{lin}. However, the internal dynamics of the nonlinear state x_{nl} in (10) are assumed to be zero. As a result, nonlinear memory effects, such as hysteresis, are difficult to capture. In contrast, our hybrid model allows non-trivial internal dynamics x_{rnn} in the nonlinear component.

Furthermore, the controller synthesis results in D'Amico et al. (2023) are restricted to *static state-feedback* controllers. In our framework, interpreting the residual model as a controller yields a *dynamic state-feedback* structure. This broader model class generally leads to synthesis conditions that are non-convex; accordingly, in Section 5.1 we propose a partial convexification that results in LMI conditions on the learnable parameters of our hybrid architecture.

Overall, the model class of D'Amico et al. (2023) can be viewed as a special case of our hybrid architecture, obtained by restricting the nonlinear component to be static and memoryless.

The system identification problem we tackle with our hybrid model reads as follows:

Problem 3.1: Let us be given a set of input and output measurements

$$\mathcal{D} = \left\{ \left((u^k)_m, (y^k)_m \right)_{k=0,\dots,N-1} \right\} \quad \text{for } m = 1, \dots, M, \tag{11}$$

which is taken from an unknown nonlinear system (1), where $M \in \mathbb{N}_+$ is the number of trajectories in the dataset and $N \in \mathbb{N}_+$ the length of the trajectories. Given a linear approximation model $\mathcal{H}_{\text{lin}}$ (S1) and an upper bound on the ℓ_2-gain (S2), find the parameters θ of the nonlinear residual model $\mathcal{R}_\theta$ such that

(O1) The quadratic prediction error

$$\text{MSE}(\mathcal{D}) := \frac{1}{MNn_y} \sum_{m=1}^{M} \sum_{k=0}^{N-1} \|(y^k)_m - (\hat{y}^k)_m\|^2$$
$$\text{with } \hat{y} = \mathcal{S}_\theta(u, \xi^0) \text{ and } u, y \in \mathcal{D} \tag{12}$$

is small and

(O2) The model $\mathcal{S}_\theta$ is dissipative with respect to a quadratic supply rate and has a guaranteed upper bound $\gamma > 0$ on the worst-case amplification.

Besides the matrices that describe the linear dynamics, the model (4) also contains matrices $B_{\text{lin},2}$ and $D_{\text{lin},2}$ that describe the interconnection between the residual model. These interconnection matrices allow us to incorporate structural prior knowledge; more details will be given in the next subsection.

3.1 Structural prior knowledge

The linear approximation model is expected to have high prediction accuracy around a specific operating point. For instance, in (2), we assume zero as the linearisation point. The hybrid structure (9) allows for the incorporation of structural information by selecting appropriate matrices $\tilde{B}_{\text{lin},2}$, $\tilde{B}_{\text{lin},3}$, $\tilde{D}_{\text{lin},2}$, and $\tilde{D}_{\text{lin},3}$. The matrices $\tilde{B}_{\text{lin},2}$ and $\tilde{D}_{\text{lin},2}$ describe how the dynamics of the residual RNN influence the states and outputs of the linear approximation model. For instance, if the nonlinear part of an unknown system is known to have no memory, these matrices can be set to zero. Additionally, $\tilde{B}_{\text{lin},3}$ and $\tilde{D}_{\text{lin},3}$ can be used to specify which states should be corrected by the residual model. In a second-order system, for example, since position depends directly on velocity, correcting only the velocities is sufficient. In Section 7.3, we will discuss specific choices for the system to be identified.

Remark 3.3: To recover a fully learnable setup the interconnection matrices $\tilde{B}_{\text{lin},3}$ and $\tilde{D}_{\text{lin},3}$ can be set to the identity and $\tilde{B}_{\text{lin},2}$ and $\tilde{D}_{\text{lin},2}$ can be set to zero. In this case, the learnable parameter of the output channel of the RNN learn the interconnection between the residual model and the linear approximation model.

Before diving into the details of our approach, we will recall some important stability definitions and give an introduction to the concept of dissipativity.

4. Background

In the introduction, we outlined various stability properties for nonlinear systems. In the first part of this Section, we will recall important stability definitions and relate them to each other. In the second part, we give an overview of the concept of dissipativity and quadratic constraints that will be used in the sequel. For this paper, and the upcoming definitions, we will consider a general feedback interconnection of an LTI system

$$\mathcal{G} := \left\{ \begin{pmatrix} \xi^{k+1} \\ y^k \\ z^k \end{pmatrix} = \begin{pmatrix} \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix} \begin{pmatrix} \xi^k \\ u^k \\ w^k \end{pmatrix} \right. \tag{13a}$$

with a static and memoryless nonlinearity $\Delta : \mathbb{R}^{n_z} \to \mathbb{R}^{n_w}$ via

$$w^k = \Delta(z^k). \tag{13b}$$

This structure matches our hybrid model (8) with matrices (9).

4.1 Different notions of stability

While stability for linear systems is well understood, the notion of stability for nonlinear systems is ambiguous and has many definitions. Each stability characterisation has dedicated literature that we refer the interested reader

to when defining the stability notions. There certainly exist other definitions of stability than those considered in this work. However, deriving a comprehensive list including links between different notions of stability is left as future work.

Definition 4.1 (((Exponential) stability)): The system (13a) is said to be (globally) exponentially stable (for $u \equiv 0$) if there exist constants $M > 0$, $\rho \in (0, 1)$ such that

$$\|\xi^k\|^2 \leq M\rho^k \|\xi^0\|^2 \tag{14}$$

holds for all $k > 0$ and all initial conditions $\xi^0 \in \mathbb{R}^{n_\xi}$.

An incremental version of exponential stability follows as

Definition 4.2 ((Incremental stability)): The system (13a) is said to be exponentially, incrementally stable (for $u \equiv 0$) if there exist constants $M_{\text{inc}} > 0$, $\rho_{\text{inc}} \in (0, 1)$ such that

$$\|(\xi^k)_a - (\xi^k)_b\|^2 \leq M_{\text{inc}} \rho_{\text{inc}}^k \|(\xi^0)_a - (\xi^0)_b\|^2 \tag{15}$$

holds for all $k > 0$ and $a \neq b$.

Remark 4.1: For an incremental stable system, the distance between two arbitrary trajectories $(\xi^k)_a$ and $(\xi^k)_b$ converges to zero, independent of the initial conditions. This is, for example, considered in the work by Revay et al. (2021). A system is contracting if condition (15) holds pointwise, e.g. $\|(\xi^{k+1})_a - (\xi^{k+1})_b\| \leq M_{\text{inc}} \|(\xi^k)_a - (\xi^k)_b\| \; \forall \, k > 0$. Contracting models (considered by Revay et al., 2021) are also incrementally stable (considered by Miller & Hardt, 2018), but the converse is not true in general.

So far, the stability properties considered autonomous systems with constant external input $u \equiv c$, e.g. $c = 0$. In system identification, we are interested in learning the input–output behaviour of an unknown system. Therefore, we will introduce stability properties that consider non-zero inputs.

Definition 4.3 ((Finite gain ℓ_2-stability, cf. Sastry, 2013, Def. 4.14)**):** The system (13a) is finite gain ℓ_2-stable if there exists $\gamma > 0$, $\gamma_0 \geq 0$ such that for a given $u \in \ell_2^{n_u}$ the output $y = \mathcal{G}(u, \xi^0)$ satisfies

$$\sum_{k=0}^{N-1} \|\hat{y}^k\|^2 \leq \gamma^2 \sum_{k=0}^{N-1} \|u^k\|^2 + \gamma_0 \quad \text{for all } N > 0. \tag{16}$$

Remark 4.2: ℓ_2-stability (16) is a global property and holds for all $u \in \ell_2^{n_u}$. The conditions on the nonlinear activation function we employ also hold globally. Thus, `HybCRnn` is globally ℓ_2-stable, local stability properties are, for example, derived in Yin et al. (2021).

Definition 4.4 ((Incremental ℓ_2-stability)): The system (13a) is said to be incrementally ℓ_2-stable if there exists $\gamma_{\text{inc}} > 0$ such that for any two inputs $u_a, u_b \in \ell_2^{n_u}$ the corresponding outputs $\hat{y}_a = \mathcal{G}(u_a, \xi_a^0)$ and $\hat{y}_b = \mathcal{G}(u_b, \xi_b^0)$ satisfy

$$\sum_{k=0}^{N-1} \|((\hat{y}^k)_a - (\hat{y}^k)_b)\|^2 \leq \gamma_{\text{inc}}^2 \sum_{k=0}^{N-1} \|(u^k)_a - (u^k)_b\|^2 \quad \text{for all } N > 0. \tag{17}$$

Remark 4.3: If condition (17) holds pointwise, then γ_{inc}^2 refers to the Lipschitz constant of the nonlinear system. This stability property is, for example, considered in Pauli, Koch, et al. (2021) and Fazlyab et al. (2019).

The properties of ℓ_2-stability and incremental ℓ_2-stability do not take into account the system's internal state ξ but rather look at the input–output behaviour. In ISS, the effect of the external input on the internal state is considered.

Definition 4.5 ((Input-to-state stability, Bonassi et al., 2022)**):** The system (13a) is said to be ISS if there exist functions $\beta(\|\xi\|_2, k) \in \mathcal{KL}$ and $\gamma_u(\|u\|_{2,\infty}) \in \mathcal{K}_\infty$ such that for any $k \geq 0$ any initial condition ξ^0, and any input sequences u the following holds

$$\|\xi^k\|_2 \leq \beta(\|\xi^0\|_2, k) + \gamma_u(\|u^k\|_{2,\infty}) \quad \text{for all } k \geq 0. \tag{18}$$

We repeat the definition of class $\mathcal{KL}$ and $\mathcal{K}_\infty$ functions from Bonassi et al. (2022) in the Appendix A.1.

Definition 4.6 ((Incremental input-to-state stability (δISS), Bonassi et al., 2022)**):** The system (13a) is said to be incrementally ISS (δISS) if there exist functions $\beta_{\text{inc}}(\|\xi_a - \xi_b\|_2, k) \in \mathcal{KL}$ and $\gamma_{\text{inc},u}(\|u_a - u_b\|_{2,\infty}) \in \mathcal{K}_\infty$ such that for any $k \geq 0$, any initial conditions $(\xi^0)_a, (\xi^0)_b$, and any input sequences $u_a \neq u_b$ the following holds

$$\|(\xi^k)_a - (\xi^k)_b\|_2 \leq \beta_{\text{inc}}(\|(\xi^0)_a - (\xi^0)_b\|_2, k) + \gamma_{\text{inc},u}(\|(u^k)_a - (u^k)_b\|_{2,\infty}) \quad \text{for all } k \geq 0. \tag{19}$$

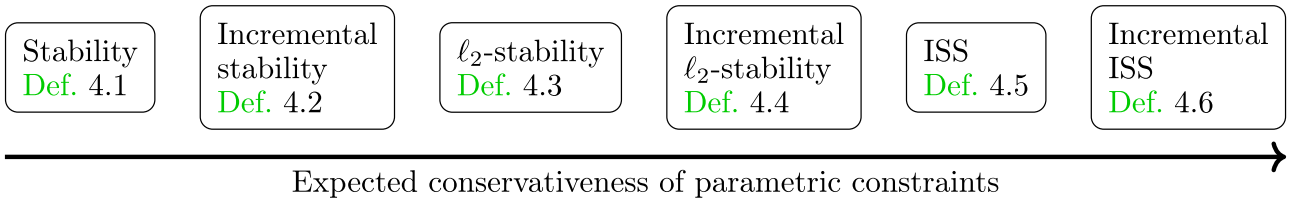

Figure 2. Different notions of stability, the arrow indicates our expectation regarding how much the learnable parameters are constrained.

Remark 4.4: If a model is ISS and has a linear output layer it is also ℓ_2-stable. This follows from a stable internal state, an external input that is bounded and a nonlinearity that is slope restricted. The converse is not true in general, making ISS a stronger notion of stability.

In Figure 2, we classify the different notions of stability by how much we expect the parameters to be constrained when enforcing the respective stability criterion in a neural network architecture.¡fig2/¿

4.2 Dissipativity and quadratic constraints

Jan Willems introduced dissipativity in Willems (1972) with the explicit goal of arriving at a fundamental understanding of the stability properties of feedback interconnections in terms of the input–output behaviour of the individual subsystems (Scherer, 2022). It can be interpreted as a generalisation of energy conservation in physical systems. Since then, and due to the ever-increasing complexity of interconnected systems to be analysed or controlled, it has become a central notion in systems theory (Arcak et al., 2016).

We will use the following definition of dissipativity for the feedback interconnection (13a)

Definition 4.7 ((Dissipativity, Byrnes & Lin, 1994)): The system (13a) is strictly dissipative with respect to the supply rate $s : \mathbb{R}^{n_d} \times \mathbb{R}^{n_e} \to \mathbb{R}$, if there exists a storage function $V : \mathbb{R}^{n_\xi} \to \mathbb{R}$ and some scalar $\epsilon > 0$ such that the inequality

$$V(\xi^{k_2}) \leq V(\xi^{k_1}) + \sum_{k=k_1}^{k_2-1} s(u^k, y^k) - \epsilon \sum_{k=k_1}^{k_2-1} \|u^k\|^2 \quad (20)$$

holds for all admissible trajectories of the system (13a) and all time instants $k_1, k_2 \in \mathbb{N}$ with $k_1 \leq k_2$. The system is said to be dissipative with respect to s if this holds for $\epsilon = 0$.

In the sequel, we will consider quadratic supply rates

$$s_{P_p} : \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}, \ (u, y) \mapsto \begin{pmatrix} u \\ y \end{pmatrix}^\top P_p \begin{pmatrix} u \\ y \end{pmatrix} \quad (21)$$

defined by some symmetric matrix P_p. Such supply rates can readily be used to investigate important system properties such as the energy gain (worst-case amplification) or passivity (energy dissipation) by making a particular choice for P_p. Following Assumption 3.1, we are particularly interested in the worst-case amplification, but want to stress that the concept of dissipativity applies to other input–output properties as well. Based on the definition of the nonlinear system (13a) we can employ tools from the robust control literature, such as integral quadratic constraints (Megretski & Rantzer, 1997), to derive suitable sufficient conditions. Instead of dealing with the interconnection (13a) directly, the general idea is to find numerically suitable quadratic constraints that the nonlinear operator Δ enforces on the interconnection signals z and w based on all the available information on Δ. Afterward, one analyses the behaviour of the interconnection's linear part (13a) subject to the identified quadratic constraints.

In our setting, the nonlinearity Δ consists of the nonlinear activation functions of the RNN. As such, it is usually static, memoryless, and repeated, i.e.

$$\Delta : z \mapsto \begin{pmatrix} \varphi(z_1) \\ \vdots \\ \varphi(z_{n_z}) \end{pmatrix} \quad \text{with a single function } \varphi : \mathbb{R} \to \mathbb{R}. \quad (22)$$

In our analysis, we will exploit sector-boundedness as well as slope-restrictedness of the activation function φ.

Definition 4.8: A function $\varphi : \mathbb{R} \to \mathbb{R}$ is said to be *sector bounded* with bounds $\alpha \leq \beta$ if

$$\varphi(0) = 0 \quad \text{and} \quad \alpha \leq \frac{\varphi(x)}{x} \leq \beta \quad \text{hold for all } x \in \mathbb{R}. \quad (23)$$

It is said to be *slope-restricted* with parameters $\mu \leq \nu$ if

$$\varphi(0) = 0 \quad \text{and} \quad \mu \leq \frac{\varphi(x) - \varphi(y)}{x - y} \leq \nu \quad \text{holds for all } x, y \in \mathbb{R} \text{ with } x \neq y. \quad (24)$$

The standard activation functions $\varphi(x) = \tanh(x)$ and $\varphi(x) = \mathrm{ReLU}(x) = \max(0, x)$ can even be shown to be sector bounded and slope restricted with bounds $\alpha = \mu = 0$ and $\beta = \nu = 1$.

Let us suppose that Δ is defined as in (22) with φ being sector bounded with bounds $\alpha \leq \beta$ and slope restricted with $\mu \leq \nu$. Then, it is a matter of direct computations to observe that the nonlinearity Δ satisfies the quadratic inequality

$$\begin{pmatrix} \Delta(z) \\ z \end{pmatrix}^\top \begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix}^\top \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix} \begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix} \begin{pmatrix} \Delta(z) \\ z \end{pmatrix} \geq 0 \quad \text{for all} \quad z \in \mathbb{R}^{n_z}. \tag{25}$$

Actually, it is not difficult to see that Δ even satisfies an infinite set of quadratic inequalities, including

$$\begin{pmatrix} \Delta(z) \\ z \end{pmatrix}^\top \underbrace{\begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix}^\top \begin{pmatrix} 0 & \Lambda \\ \Lambda^\top & 0 \end{pmatrix} \begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix}}_{P} \begin{pmatrix} \Delta(z) \\ z \end{pmatrix} \geq 0 \quad \text{for all } z \in \mathbb{R}^{n_z}, \tag{26}$$

where Λ is an arbitrary diagonal matrix with positive diagonal entries. The product of the three inner matrices is typically referred to as a multiplier. Moreover, the larger the class of multipliers compatible with Δ, the less conservative the corresponding sufficient conditions for dissipativity will be. We will use the following result from Scherer and Weiland (2000) that applies to any static memoryless nonlinearity Δ.

Theorem 4.9 ((Robust) Strict Dissipativity, Scherer & Weiland, 2000): *Let* $\mathbf{P}$ *be a set of symmetric matrices with the property that*

$$\begin{pmatrix} \Delta(z) \\ z \end{pmatrix}^\top P \begin{pmatrix} \Delta(z) \\ z \end{pmatrix} \geq 0 \quad \text{for all } z \in \mathbb{R}^{n_z} \text{ and all } P \in \mathbf{P}. \tag{27}$$

Then, the system (13) *is strictly dissipative with respect to the supply rate* s_{P_p} *if there exist a symmetric matrix* $\mathcal{X}$ *and a multiplier* $P \in \mathbf{P}$ *such that*

$$\begin{aligned} &\begin{pmatrix} I & 0 & 0 \\ \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \end{pmatrix}^\top \begin{pmatrix} -\mathcal{X} & 0 \\ 0 & \mathcal{X} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \end{pmatrix} \\ &- \begin{pmatrix} 0 & I & 0 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \end{pmatrix}^\top P_p \begin{pmatrix} 0 & I & 0 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \end{pmatrix} \\ &+ \begin{pmatrix} 0 & 0 & I \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix}^\top P \begin{pmatrix} 0 & 0 & I \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix} \prec 0. \end{aligned} \tag{28}$$

If, in addition, $R_p = \left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right)^\top P_p \left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right) \succeq 0$ *and* $\mathcal{X} \succ 0$ *hold, then the system* (13) *is also exponentially stable according to Definition* 4.1.

In continuous time, the proof is presented by Scherer (2000) and in discrete time by De Oliveira et al. (2002).

To apply Theorem 4.9, we must make a concrete and suitable choice for the multiplier set $\mathcal{P}$. Next, we recall some possibilities that could be extracted from Veenman et al. (2016). We start with the simplest one based on the quadratic inequality (26). It has been used for example in Fazlyab et al. (2019), Revay et al. (2020), and Pauli et al. (2022).

Lemma 4.10 (DG-Multipliers, Scherer & Weiland, 2000): *Let* Δ *be static, memoryless, and repeated with* φ *being sector-bounded with bounds* $\alpha \leq \beta$. *Then*

$$\mathcal{P}_{DG} := \left\{ \begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix}^\top \begin{pmatrix} 0 & \Lambda \\ \Lambda & 0 \end{pmatrix} \begin{pmatrix} -I & \beta I \\ I & -\alpha I \end{pmatrix} \middle| \Lambda \in \mathbf{\Lambda}_{DG} \right\} \tag{29}$$

is a set satisfying (27) *with* $\mathbf{\Lambda}_{DG} \in \mathbb{D}_+^{n_z}$.

Another possibility, on which more details can be found, in Willems and Brockett (1968), Fetzer and Scherer (2017) and which requires slope-restrictedness, is given as follows.

Lemma 4.11 ((Static) Zames–Falb–Multipliers, Fetzer & Scherer, 2017): *Let* Δ *be static, memoryless, and repeated with* φ *being slope-restricted with bounds* $\mu \leq \nu$. *Then*

$$\mathcal{P}_{ZF} := \left\{ \begin{pmatrix} -I & \nu I \\ I & -\mu I \end{pmatrix}^\top \begin{pmatrix} 0 & \Lambda \\ \Lambda^\top & 0 \end{pmatrix} \begin{pmatrix} -I & \nu I \\ I & -\mu I \end{pmatrix} \middle| \Lambda \in \mathbf{\Lambda}_{ZF} \right\} \tag{30}$$

is a set satisfying (27) *with*

$$\mathbf{\Lambda}_{ZF} = \Big\{ \Lambda = (\Lambda_{ij}) \in \mathbb{R}^{n_z \times n_z} | \Lambda_{i,j} \leq 0 \text{ for all } i \neq j, \\ \Lambda e \geq 0 \text{ and } e^\top \Lambda \geq 0 \Big\}. \tag{31}$$

Here, e denotes the all-one vector, and the last two inequalities are meant element-wise.

The DG-Multipliers are a subset of the static Zames–Falb–Multiplier, i.e. $\mathcal{P}_{DG} \subset \mathcal{P}_{ZF}$. In this work, we will consider slope-restricted and sector-bounded nonlinearities with $\mu = \alpha$ and $\nu = \beta$ and, therefore, will continue with the parameters μ and ν only.

5. Ensuring ℓ_2-stability for the hybrid model

After fixing the hybrid architecture in Section 3, briefly reviewing the required dissipativity results, and introducing the multipliers in Section 4.2, we will pick a supply rate and formulate constraints on the learnable parameters in this section.

In addition to the linear approximation, we will use the prior system knowledge from Assumption 3.1 and embed it into `HybCRnn`.

The value γ describes the energy gain of the system and can be interpreted as an amplification factor of the input signal. We further assume to have some knowledge about the upper bound of γ. In this work, we assume that γ is either provided by an expert or extracted from the frequency behaviour of the known linear system $\mathcal{H}_{\text{lin}}$ (4). There certainly exist other approaches to obtain γ, e.g. in Koch et al. (2021) an upper bound on the worst-case amplification is captured based on input–output measurements; however, these methods are not considered here.

First, let us introduce the supply rate that captures the bound on the worst-case amplification or equivalently the ℓ_2-gain

$$s_{P_p}(d,e) = \begin{pmatrix} d \\ e \end{pmatrix}^\top \underbrace{\begin{pmatrix} \gamma^2 I & 0 \\ 0 & -I \end{pmatrix}}_{=P_p} \begin{pmatrix} d \\ e \end{pmatrix}. \tag{32}$$

According to the conditions in Theorem 4.9, we can formulate constraints on the parameters θ involving the system matrices of `HybCRnn` (8). In addition to the parameters θ, we must find the symmetric matrix $\mathcal{X} \in \mathbb{S}_+$ and the scaling Λ from the multipliers $\mathcal{P}$. The following corollary can be viewed as a test for ℓ_2-stability of (8) when a parameter set θ is given.

Corollary 5.1: *The system* (8) *has an ℓ_2-gain upper bounded by γ for a given parameter set θ, slope restricted nonlinear activation function with $\mu \leq \nu$ and supply rate* (32) *if there exists $\mathcal{X}$ satisfying $\mathcal{X} \succ 0$ and*

$$\begin{aligned} &(\star)^\top \begin{pmatrix} -\mathcal{X} & 0 \\ 0 & \mathcal{X} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \end{pmatrix} - (\star)^\top \begin{pmatrix} \gamma^2 I & 0 \\ 0 & -I \end{pmatrix} \\ &\begin{pmatrix} 0 & I & 0 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \end{pmatrix} + (\star)^\top \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & \nu\mathbf{\Lambda}^\top + \mu\mathbf{\Lambda} \\ \nu\mathbf{\Lambda} + \mu\mathbf{\Lambda}^\top & -\nu\mu(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) \end{pmatrix} \begin{pmatrix} 0 & 0 & I \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix} \prec 0. \end{aligned} \tag{33}$$

For multipliers $\mathbf{\Lambda} = \Lambda_{DG}$ or $\mathbf{\Lambda} = \Lambda_{ZF}$.

Proof: We left-multiply the inequality (33) with $(u^k)^\top$, $(w^k)^\top$) and right-multiply with $(w^k)^\top)^\top$. This implies

$$\begin{aligned} &-(\xi^k)^\top \mathcal{X} \xi^k + (\xi^{k+1})^\top \mathcal{X} \xi^{k+1} + \|y^k\|^2 - \gamma^2 \|u^k\|^2 \\ &+ \begin{pmatrix} w^k \\ z^k \end{pmatrix}^\top \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & (\nu+\mu)\mathbf{\Lambda} \\ (\nu+\mu)\mathbf{\Lambda}^\top & -\nu\mu(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) \end{pmatrix} \begin{pmatrix} w^k \\ z^k \end{pmatrix} \\ &< 0 \end{aligned} \tag{34}$$

along trajectories of (8). We now exploit the slope-restrictedness of the nonlinearity which ensures $\begin{pmatrix} \Delta(z^k) \\ z^k \end{pmatrix} \geq 0$ for the multipliers

$$P = \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & (\nu+\mu)\mathbf{\Lambda} \\ (\nu+\mu)\mathbf{\Lambda}^\top & -\nu\mu(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) \end{pmatrix} \tag{35}$$

and allows us to cancel the last term in (34). The inequality stays valid if we add a small $\epsilon > 0$, which implies

$$\begin{aligned} &-(\xi^k)^\top \mathcal{X} \xi^k + (\xi^{k+1})^\top \mathcal{X} \xi^{k+1} + \|y^k\|^2 - \gamma^2 \|u^k\|^2 \\ &+ \epsilon \|u^k\|^2 \leq 0. \end{aligned} \tag{36}$$

We now sum from 0 to $N-1$ to get

$$\begin{aligned} V(\xi^N) \leq V(\xi^0) &- \sum_{k=0}^{N-1} \|y^k\|^2 + \gamma^2 \sum_{k=0}^{N-1} \|u^k\|^2 \\ &- \epsilon \sum_{k=0}^{N-1} \|u^k\|^2 \end{aligned} \tag{37}$$

with the storage function $V(\xi) = \xi^\top \mathcal{X} \xi$. Since $\epsilon > 0$, $\|u^k\| \geq 0$ and $\mathcal{X} \succ 0$ the following holds

$$\sum_{k=0}^{N-1} \|y^k\|^2 \leq \gamma^2 \sum_{k=0}^{N-1} \|u^k\|^2 + V(\xi^0). \tag{38}$$

Since $V(\xi^0)$ does not depend on N ℓ_2-stability as defined in (16) follows for $\gamma_0 = V(\xi^0) \geq 0$.

■

Corollary 5.1 is a direct consequence of Theorem 4.9 and allows us to verify if `HybCRnn` (8) described by the parameters θ is ℓ_2-stable. Let us formulate Problem 3.1 as constrained optimisation problem

$$\begin{aligned} \min_{\theta,\mathcal{X},\Lambda} \quad & \text{MSE}(\mathcal{D}) \\ \text{such that} \quad & \mathcal{X} \succ 0, \quad \Lambda \in \mathbf{\Lambda} \text{ and (33) holds,} \end{aligned} \tag{39}$$

with the supply rate (32). We use either $\mathbf{\Lambda} = \mathbf{\Lambda}_{DG}$ or $\mathbf{\Lambda} = \mathbf{\Lambda}_{ZF}$ to show that the analysis (and synthesis) works for diagonal multipliers as well as for static Zames–Falb multipliers from Lemmas 4.10 and 4.11.

Next, we will discuss finding solutions to the optimisation problem (39). For learning θ, we follow an iterative training procedure based on the gradient of the predicted error (12). The constraints in problem (39) are semi-definite programming constraints, and (34) depends on the decision variables in a non-affine and non-convex fashion.

Our model (`HybCRnn`) is an interconnection of a linear model with a nonlinear dynamic RNN. In the RNN, we can separate the static nonlinearity from the linear transformations, leading to an interconnected system that consists of a linear part and the nonlinear activation function. In the control literature, finding the linear system's parameters $\mathcal{R}_{\text{lin}}$ is said to be a controller synthesis problem. In the next section, we will use tools known from robust control that allow us to transform (39) into an equivalent problem with constraints depending on a new set of decision variables in an affine fashion.

5.1 (Convex) controller synthesis

In the transformation, we will introduce a new set of parameters and formulate a new optimisation problem with convex constraints with respect to the new parameters. Then, we will show that the two problem formulations are equivalent. Finding a close-to-optimal solution of the resulting problem remains numerically challenging, but we observed higher prediction accuracies if compared to the original formulation in (39) with the non-convex constraint. Details about this observation are given in Section 8. The objective function will remain non-convex with respect to the new parameter set, which is common when optimising the parameters of a neural network.

To transform condition (33), let us first introduce a new set of decision variables

$$\tilde{\omega} = (K, L, L_2, M, \tilde{M}_2, N, N_{12}, \tilde{N}_{21}) \tag{40}$$

which are defined as

$$\begin{aligned} &\begin{pmatrix} K & L & L_2 \\ M & N & N_{12} \\ \tilde{M}_2 & \tilde{N}_{21} & 0 \end{pmatrix} \\ &:= \begin{pmatrix} U & XB_{\text{lin},2} & 0 \\ 0 & I & 0 \\ 0 & 0 & \mathbf{\Lambda} \end{pmatrix} \left(\begin{pmatrix} A & B & B_2 \\ C & D & D_{12} \\ C_2 & D_{21} & 0 \end{pmatrix} \right. \\ &\left. + \begin{pmatrix} XA_{\text{lin}}Y & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \right) \begin{pmatrix} V^\top & 0 & 0 \\ \left(\begin{smallmatrix} Y \\ 0 \end{smallmatrix}\right) & I_{n_y} & 0_{n_y \times n_w} \\ 0 & 0 & I \end{pmatrix} \end{aligned} \tag{41}$$

with $\tilde{M}_2 := \mathbf{\Lambda} M_2$, $\tilde{N}_{21} := \mathbf{\Lambda} N_{21}$. We will refer to X, Y, U, V as coupling matrices necessary to show that the resulting optimisation problem is equivalent to (39). With the new set of learnable parameters, we can state the main technical result of the paper.

Theorem 5.2: *Let $\mathbf{P} = \mathcal{P}_{DG}$ or $\mathbf{P} = \mathcal{P}_{ZF}$ be the set of multipliers satisfying* (27)*, characterised by $\mathbf{\Lambda} = \mathbf{\Lambda}_{DG}$ or $\mathbf{\Lambda} = \mathbf{\Lambda}_{ZF}$, respectively. Consider the quadratic supply rate* (32) *and nonlinear activation function that is slope restricted with $\mu = 0$ and $\nu = 1$. Then the following statements are equivalent*

(1) *There exists θ, $\mathcal{X}$ and $\mathbf{\Lambda}$ that satisfy* (33) *and $\mathcal{X} \succ 0$*
(2) *There exists $\tilde{\omega}$, X, Y and $\mathbf{\Lambda}$ that satisfy*

$$\left(\begin{array}{ccc|cc} -\mathbf{X} & 0 & (\star)^\top & & \\ 0 & -\gamma^2 I & (\star)^\top & (\star)^\top & \\ \tilde{\mathbf{C}}_2 & \tilde{\mathbf{D}}_{21} & -(\mathbf{\Lambda}^\top + \mathbf{\Lambda}) & & \\ \hline \mathbf{A} & \mathbf{B} & \mathbf{B}_2 & -\mathbf{X} & 0 \\ \mathbf{C} & \mathbf{D} & \mathbf{D}_{12} & 0 & -I \end{array}\right) \prec 0, \tag{42}$$

for the matrices

$$\left(\begin{array}{c|cc} \mathbf{A} & \mathbf{B} & \mathbf{B}_2 \\ \hline \mathbf{C} & \mathbf{D} & \mathbf{D}_{12} \\ \tilde{\mathbf{C}}_2 & \tilde{\mathbf{D}}_{21} & 0 \end{array}\right) = \left(\begin{array}{cc|cc} A_{\text{lin}}Y + B_{\text{lin},2}N\left(\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right) & A_{\text{lin}} + B_{\text{lin},2}M & B_{\text{lin}} + B_{\text{lin},2}N\left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right) & B_{\text{lin},2}N_{12} \\ L\left(\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right) & XA_{\text{lin}} + K & XB_{\text{lin}} + L\left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right) & L_2 \\ \hline C_{\text{lin}}Y + D_{\text{lin},2}N\left(\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right) & C_{\text{lin}} + D_{\text{lin},2}M & D_{\text{lin}} + D_{\text{lin},2}N\left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right) & D_{\text{lin},2}N_{12} \\ \tilde{N}_{21}\left(\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right) & \tilde{M}_2 & \tilde{N}_{21}\left(\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right) & 0 \end{array}\right) \tag{43a}$$

and

$$\mathbf{X} = \begin{pmatrix} Y & I \\ I & X \end{pmatrix} \tag{43b}$$

with $\tilde{\mathbf{C}}_2 := \mathbf{\Lambda}\mathbf{C}_2$, $\tilde{\mathbf{D}}_{21} := \mathbf{\Lambda}\mathbf{D}_{21}$ *and* $\mathbf{\Lambda} \in \mathbf{\Lambda}$.

Proof: To prove the theorem, we will show that we can recover the inequality (33) from (42). First we apply the Schur-complement to (42) to get

$$\begin{aligned} &(\star)^\top \begin{pmatrix} \mathbf{X}^{-1} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{B}_2 \\ \mathbf{C} & \mathbf{D} & \mathbf{D}_{12} \end{pmatrix} \\ &\quad + \begin{pmatrix} -\mathbf{X} & 0 & (\star)^\top \\ 0 & -\gamma^2 I & (\star)^\top \\ \tilde{\mathbf{C}}_2 & \tilde{\mathbf{D}}_{21} & -(\mathbf{\Lambda}^\top + \mathbf{\Lambda}) \end{pmatrix} \prec 0. \end{aligned} \tag{44}$$

The inequality can be rewritten such that it matches the structure of (33) and follows as

$$\begin{aligned} &(\star)^\top \begin{pmatrix} -\mathbf{X} & 0 \\ 0 & \mathbf{X}^{-1} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ \mathbf{A} & \mathbf{B} & \mathbf{B}_2 \end{pmatrix} \\ &\quad - (\star)^\top P_p \begin{pmatrix} 0 & I & 0 \\ \mathbf{C} & \mathbf{D} & \mathbf{D}_{12} \end{pmatrix} \\ &\quad + (\star)^\top \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & I \\ I & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & I \\ \mathbf{\Lambda}\mathbf{C}_2 & \mathbf{\Lambda}\mathbf{D}_{21} & 0 \end{pmatrix} \\ &\quad \prec 0. \end{aligned} \tag{45}$$

Now, let us introduce transformation matrices that contain the coupling parameters and are based on the structure of $\mathcal{A}$. We define

$$\mathcal{Y} := \begin{pmatrix} Y & I \\ V^\top & 0 \end{pmatrix}, \quad \mathcal{X} := \mathcal{Y}^{-\top}\mathbf{X}\mathcal{Y}^{-1} \quad \text{and}$$
$$\mathcal{Z} := \mathbf{X}\mathcal{Y}^{-1} = \begin{pmatrix} I & 0 \\ X & U \end{pmatrix}. \tag{46}$$

Given X and Y, we can construct square coupling matrices U and V such that the factorisation $XY + UV^\top = I$ holds, which is required for the identity $\mathbf{X} = \mathcal{Y}^\top\mathcal{X}\mathcal{Y}$ to be valid. Condition (42) implies that $\mathbf{X} \succ 0$ and due to symmetry of X and Y we infer $\mathbf{X} \in \mathbb{S}_+$. In addition, it follows that $X, Y, \mathcal{X} \in \mathbb{S}_+$. The transformation matrices (46), which includes the coupling matrices, allow us to write the matrices (43a) as

$$\begin{pmatrix} \mathcal{Z}\mathcal{A}\mathcal{Y} & \mathcal{Z}\mathcal{B} & \mathcal{Z}\mathcal{B}_2 \\ \mathcal{C}\mathcal{Y} & \mathcal{D} & \mathcal{D}_{12} \\ \mathbf{\Lambda}\mathcal{C}_2\mathcal{Y} & \mathbf{\Lambda}\mathcal{D}_{21} & 0 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{B}_2 \\ \mathbf{C} & \mathbf{D} & \mathbf{D}_{12} \\ \tilde{\mathbf{C}}_2 & \tilde{\mathbf{D}}_{21} & 0 \end{pmatrix}. \tag{47}$$

We provide a more detailed calculation of (47) in the Appendix 1. By inserting transformed matrices into the inequality (45), we get

$$\begin{aligned} &(\star)^\top \begin{pmatrix} -\mathcal{Y}^\top\mathcal{X}\mathcal{Y} & 0 \\ 0 & (\mathcal{Y}^\top\mathcal{X}\mathcal{Y})^{-1} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ \mathcal{Z}\mathcal{A}\mathcal{Y} & \mathcal{Z}\mathcal{B} & \mathcal{Z}\mathcal{B}_2 \end{pmatrix} - (\star)^\top P_p \begin{pmatrix} 0 & I & 0 \\ \mathcal{C}\mathcal{Y} & \mathcal{D} & \mathcal{D}_{12} \end{pmatrix} \\ &\quad + (\star)^\top \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & \mathbf{\Lambda} \\ \mathbf{\Lambda} & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & I \\ \mathcal{C}_2\mathcal{Y} & \mathcal{D}_{21} & 0 \end{pmatrix} \prec 0. \end{aligned} \tag{48}$$

With the identity $\mathcal{X}^\top\mathcal{Y}(\mathcal{Y}^\top\mathcal{X}\mathcal{Y})^{-1}\mathcal{Y}^\top\mathcal{X} = \mathcal{X}^\top$ and due to symmetry of $\mathcal{X}$ we can write the inequality as

$$\begin{aligned} &(\star) \begin{pmatrix} -\mathcal{X} & 0 \\ 0 & \mathcal{X} \end{pmatrix} \begin{pmatrix} \mathcal{Y} & 0 & 0 \\ \mathcal{A}\mathcal{Y} & \mathcal{B} & \mathcal{B}_2 \end{pmatrix} \\ &\quad - (\star)^\top P_p \begin{pmatrix} 0 & I & 0 \\ \mathcal{C}\mathcal{Y} & \mathcal{D} & \mathcal{D}_{12} \end{pmatrix} \\ &\quad + (\star)^\top \begin{pmatrix} -(\mathbf{\Lambda} + \mathbf{\Lambda}^\top) & \mathbf{\Lambda} \\ \mathbf{\Lambda} & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & I \\ \mathcal{C}_2\mathcal{Y} & \mathcal{D}_{21} & 0 \end{pmatrix} \prec 0. \end{aligned}$$

To recover the structure of the multipliers (26), we moved the scaling $\mathbf{\Lambda}$ to the middle part of the last term. Next we will apply a congruence transformation with $\mathrm{diag}(\mathcal{Y}^{-1}, I, I)$ to arrive at (33). We then can apply Corollary 5.1 to show that the original model is ℓ_2-stable, which finishes the proof.

For completeness, let us also add the transformation from the new parameters $\tilde{\omega}$ to the original simulation parameter θ, which will be necessary for simulating the hybrid model. First we revert the substitution $M_2 = \Lambda^{-1}\tilde{M}_2$, $N_{21} = \Lambda^{-1}\tilde{N}_{21}$ and then get the original parameters by

$$\begin{aligned} &\begin{pmatrix} A & B & B_2 \\ C & D & D_{12} \\ C_2 & D_{21} & 0 \end{pmatrix} \\ &\quad = \begin{pmatrix} U & XB_{\mathrm{lin},2} & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix}^{-1} \left(\begin{pmatrix} K & L & L_2 \\ M & N & N_{12} \\ M_2 & N_{21} & 0 \end{pmatrix} \right. \end{aligned}$$

$$- \begin{pmatrix} XA_{\text{lin}}Y & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \Bigg) \begin{pmatrix} V^\top & 0 & 0 \\ \binom{Y}{0} & I_{n_y} & 0_{n_y \times n_w} \\ 0 & 0 & I \end{pmatrix}^{-1} . \tag{49}$$

■

We have now shown that if (42) holds, the hybrid model (8) is dissipative with respect to the quadratic supply rate (32) and according to Corollary 5.1 is ℓ_2-stable. With the new set of learnable matrices $\tilde{\omega}$, we can write the system identification problem as a constrained optimisation problem with convex constraints

$$\begin{aligned} &\min_{\tilde{\omega}, X, Y, \mathbf{\Lambda}} && \text{MSE}(\mathcal{D}) \\ &\text{such that} && \mathbf{X} \succ 0, \quad \Lambda \in \mathbf{\Lambda} \text{ and (42) holds.} \end{aligned} \tag{50}$$

In order to get $\hat{y} = \mathcal{S}_\theta(u, \xi^0)$ for calculating the squared error, we require the original simulation parameter set θ, which now depends on the new set of parameters $\tilde{\omega}, X, Y, \mathbf{\Lambda}$. To recover simulation parameters θ, the following steps known from robust control are done.

Remark 5.1: In controller synthesis problems, the procedure to obtain the original parameters θ usually consists of the following steps (Scherer & Weiland, 2000):

- Solve the semidefinite program (SDP) to find transformed system parameters $\tilde{\omega}, X, Y, \mathbf{\Lambda}$.
- Determine non-singular U, V that satisfy $XY + UV^\top = I$, in our transformation we use $U = Y^{-1} - X$, $V = Y$.
- Recover original parameters θ with (49) and the quadratic storage function defined by $\mathcal{X}$ with $\mathcal{X} = (\mathcal{Y}^\top)^{-1}\mathcal{Z}$.

In contrast to our problem (50), the objective in robust controller synthesis problems is usually convex, and thus, the first step can be done with off-the-shelf solvers. Due to the non-convex objective function, we will use an iterative algorithm to learn the simulation parameters θ ($\tilde{\omega}$) and explain the details in the next section.

6. Training of RNN parameter

After fixing the model structure and introducing new parameters with convex constraints, we can now use the input–output dataset to learn the parameters, e.g. solve the optimisation problem (50).

6.1 Data preprocessing

The dataset $\mathcal{D}$ consists of M input–output trajectories of length N. We first split it into three separate datasets: one for training, one for testing, and one for validation and hyperparameter optimisation. We denote the subset by a subscript. The training dataset $\mathcal{D}_{\text{train}}$ is subdivided into sub-trajectories of length $w + h + 1$, where w indicates a window of past inputs used for initialisation and h is the prediction horizon. Note that usually $h \ll N$, which reduces the risk of exploding gradients and stabilises the training process (Ribeiro et al., 2020). The subtrajectories are shuffled into random batches and then processed by the model, leading to a stochastic gradient descent scheme.

6.2 Loss function

To find the parameters of the RNN that satisfy the constraints, we use logarithmic barrier functions known from interior-point methods in convex optimisation (Boyd et al., 2004) and define them as :

$$\phi(\mathbf{C}) := \begin{cases} -\log\det(-\mathbf{C}) & \text{if } \mathbf{C} \prec 0 \\ \infty & \text{if } \mathbf{C} \not\prec 0. \end{cases} \tag{51}$$

The intuition behind the barrier function is to push the parameters away from the constraint boundaries, by adding them as a penalty in our loss function. Regularisation is a soft constraint that is not guaranteed to be satisfied after an update step. Therefore, it is necessary to check whether conditions from (50) are satisfied for the updated parameters.

If we denote by $\mathbf{F}$ the matrix on the left-hand side of (42), our loss function reads as

$$\begin{aligned} &\mathcal{L}_\theta(\hat{y}, y) \\ &\quad = \frac{1}{h}\sum_{k=0}^{h-1} \|(y^k) + (\hat{y}^k)\|^2 \\ &\qquad + \kappa \left(\phi(\mathbf{F}) + \sum_{j=1}^{C} \phi(\mathbf{C}_j) \right) \text{ with } \hat{y} = \mathcal{S}_\theta(u, \xi^0). \end{aligned} \tag{52}$$

The first part is the quadratic simulation error that measures the accuracy of the model's predictions (O1), and the second part is the regularisation that ensures the parametric constraints (O2). The barrier parameter κ regulates the penalisation of the constraints. If κ is large, parameters close to the constraint boundary result in a high loss. In order to put more emphasis on the original loss function, the barrier parameter κ is gradually reduced during training.

The regularisation consists of the LMI constraint (42) and the condition of the chosen multipliers ($\mathbf{C}_i$). For the multipliers, we distinguish the following two cases:

- $P \in \mathbf{P}_{DG}, \Lambda \in \mathbf{\Lambda}_{DG}$: $\Lambda \succ 0$ and since $\mathbf{\Lambda}_{DG}$ is diagonal we can write it as scalar constraints in order to use $\sum_{j=1}^{n_z} \phi(-\Lambda_j)$ (see Lemma 4.10).
- $P \in \mathbf{P}_{ZF}, \Lambda \in \mathbf{\Lambda}_{ZF}$: $\Lambda)_i) + \sum_{j,i=1,j\neq i}^{n_z} \phi(\Lambda_{i,j})$ (see Lemma 4.11).

Note that the logarithmic barrier functions (51) are only defined when (42) holds ($\mathbf{F} \prec 0$) and $\mathbf{C}_i \prec 0$ in (52) is satisfied. Therefore, starting from parameters that fulfill the convex conditions is necessary.

6.3 Initial parameters

The regularisation, which are the logarithmic barrier functions (51), requires the initial parameter set $\tilde{\omega}^0$ to be feasible.

Remark 6.1: We indicate the initial parameters with a superscript 0. In contrast to the discrete trajectories the index refers to the zero-th training step and not the time step.

The linear approximation model allows us to set the original initial parameters θ^0 to zero, resulting in a residual model without error correction. To initialise the transformed parameters $\tilde{\omega}^0$, we also need initial coupling parameters X^0, Y^0, and Λ^0. Therefore, the initialisation can be written as the following projection problem

$$\begin{aligned} \min_{\tilde{\omega},X,Y,\Lambda} \quad & \|\tilde{\omega}\| \\ \text{such that} \quad & \text{(42) is satisfied for some } \Lambda \in \mathbf{\Lambda}_{DG} \text{ or} \\ & \Lambda \in \mathbf{\Lambda}_{ZF} \text{ and } \mathbf{X} \succ 0 \end{aligned} \tag{53}$$

As we have seen in the previous section, the constraints (42) are convex with respect to the optimisation parameters $\tilde{\omega}$. In addition, the objective function, which aims to minimise the size of $\tilde{\omega}$, is convex. Therefore, we can efficiently solve problem (53) with standard algorithms.

Remark 6.2: If we find a solution to (53) we find a feasible parameter set $\tilde{\omega}^0, X^0, Y^0, \Lambda^0$, where the size of $\tilde{\omega}$ is minimal. This objective is not identical to the intuition of zero error correction due to the relation between $\tilde{\omega}^0$ and θ^0 in (41). We leave it as a topic of future research to investigate how to find initial parameters $\tilde{\omega}^0$ close to zero when transformed to the original parameters θ.

6.4 Initial hidden state

The initial hidden state is the internal state of the hybrid model at time step $k = 0$ and represents physical properties such as position or velocity. Hence, a good approximation of the initial state is crucial for accurate predictions (M. Schoukens, 2021). However, the physical state is generally not part of the input–output dataset. For `HybCRnn`, we use *washout* initialisation introduced in Mohajerin and Waslander (2019), while for the non-constrained baselines we use *NN-based* initialisation.

6.5 Training process

Let us now summarise the training process shown in Algorithm 1.

Before training, we split the training dataset into sub-trajectories and shuffle them into mini-batches. We update the parameters by taking the mean gradient over all sub-trajectories in one batch. This training procedure leads to stochastic gradient descent. For updating the parameters, we use the Adam optimiser (Kingma & Ba, 2015) (lines 7–11 in Algorithm 1).

After a parameter update, we check if the new parameters satisfy the conditions required for ℓ_2-stability (we check whether inequality (42) holds and whether the conditions for the multipliers are satisfied). If the conditions are not satisfied, we step back towards the old parameter set until the conditions are satisfied. When no feasible parameter set is found after 100 backtracking steps, we assume that we converged to a local optimum and stop the training process. This training procedure is also used in Revay et al. (2020) and Pauli et al. (2022) (lines 12–14 in Algorithm 1).

In addition to the training dataset, we also use the validation dataset to schedule the learning rate and the barrier parameter. To replicate the behaviour of interior point methods from convex optimisation, we evaluate the parameters at the end of one epoch on the validation set. We adjust the barrier parameter if the validation loss does not decrease for a fixed number of steps. With the same scheduling rule, we also reduce the learning rate (lines 15–18 in Algorithm 1). This allows parameters close to the constraint boundaries and avoids overshooting a local optimum towards the end of the training process.

The parameter transformation that we introduced in Section 5.1 is used in line 5 of Algorithm 1 to recover simulation parameters of the hybrid model.

7. Evaluation

This section will provide numerical examples to validate our proposed hybrid architecture.

Algorithm 1 Training transformed parameters $\tilde{\omega}$ for hybrid architecture (8)

Require: $\mathcal{D}_{\text{train}}$ (training dataset), $\mathcal{D}_{\text{val}}$ (validation dataset)

$A_{\text{lin}}, B_{\text{lin}}, B_{\text{lin},2}, C_{\text{lin}}, D_{\text{lin}}, D_{\text{lin},2}, \gamma$ ▷ side information (S1), (S2)

τ (learning rate) κ (barrier parameter)

$\tilde{\omega}, X, Y, \Lambda \leftarrow \arg\min_{\tilde{\omega},X,Y,\Lambda} \|\tilde{\omega}\|$ s.t. $\mathbf{F} \prec 0$ ▷ Solve Problem (53)

while not converged **do**

  $\theta, \mathcal{X}, \Lambda \leftarrow$ BijectiveTransformation$(\tilde{\omega}, X, Y, \Lambda)$

  for $(u^{0:w})_b$, $(u^{w+1:w+1+h})_b$, $(y^{w+1:w+1+h})_b$ from $\mathcal{D}_{\text{batch}}$ **do**

    $(\xi^{w+1})_b \leftarrow$ Simulate$(\mathcal{S}_\theta((u^{0:w})_b, 0))$ ▷ Intialisation

    $(\hat{y}^{w+1:w+1+h})_b \leftarrow$ Simulate$(\mathcal{S}_\theta((u^{w+1:w+1+h})_b, \xi^{w+1}))$ ▷ Simulation

  end for

  $G_{\tilde{\omega},X,Y,\Lambda} \leftarrow \nabla_{\tilde{\omega},X,Y,\Lambda} \frac{1}{B} \sum_{b=1}^{B} \mathcal{L}_\theta((\hat{y})_b, (y)_b)$ ▷ Mean gradient of mini-batch

  $\tilde{\omega}, X, Y, \Lambda \leftarrow$ OptimisationStep$(G_{\tilde{\omega},X,Y,\Lambda}, \tau, \eta)$ ▷ `Adam` optimiser

  if not Feasible$(\tilde{\omega}, X, Y, \Lambda)$ **then**

    $\tilde{\omega}, X, Y, \Lambda \leftarrow$ Backtracking$(\tilde{\omega}, X, Y, \Lambda)$

  end if

  $l_{\text{val}} \leftarrow$ Evaluate$(S_\theta, \mathcal{D}_{\text{val}})$

  if l_{val} did not decay for K steps **then**

    $\kappa \leftarrow \frac{\kappa}{10}, \tau \leftarrow \frac{\tau}{4}$ ▷ Scheduling

  end if

end while

All the models are implemented in *Python* using *PyTorch*. The code can be found on *Github* in the package `deepsysid`[1]. This package allows hyperparameter optimisation on a finite grid.

7.1 Baselines

We will compare `HybCRnn`(ours) (8) against the following baseline:

- `LSTM`: State-of-the-art LSTM in system identification as presented in Mohajerin and Waslander (2019). The LSTM does not provide input–output guarantees and is trained without parametric constraints.
- `cLSTM`: A constrained LSTM with regularisation that punishes parameter sets that are not ISS as described in Bonassi et al. (2020, 2022).
- `ReLiNet`: A switched linear system where the parameters are predicted by an LSTM as described in Baier et al. (2023). ReLiNet has two variants, namely an unconstrained model, denoted as `ReLiNet`, which does not provide stability guarantees, and a variant that ensures exponential stability by constraining the linear system matrices to be Hurwitz and pairwise commutative. We refer to the stable variant as `StableReLiNet`.
- `CRnn`: A constrained RNN without side information of the linearised system but guaranteed ℓ_2-stability as presented in Revay et al. (2020).

The `HybCRnn`(ours) and `CRnn` were trained on a CPU cluster, while we trained the `LSTM`, cLSTM, `ReLiNet`, and `StableReLiNet` model on *A100* GPUs. In our experiments, we encountered faster computation on CPUs for the constrained models than training on GPUs. While *PyTorch* is highly parallelised for established model structures, we could not yet exploit the GPU potential for the constrained networks. For the constrained LSTM (`cLSTM`), we added a regularisation term to the loss function as suggested in Bonassi et al. (2022). After training, the conditions (A.3 from Bonassi et al., 2022) can be analysed. In our experiments the conditions where not satisfied for `LSTM` and `cLSTM` model.

7.2 Metrics

We split our evaluation into accuracy (O1) (small quadratic prediction error) and stability (O2) (ℓ_2-stability), which matches the objectives in Problem 3.1.

7.2.1 Accuracy

To test the prediction accuracy of a model, we evaluate the distance of the model's output and the measurements from the test dataset (Based on the datasplit introduced in Section 6.1). As a metric, we use the normalised, root mean squared error (MSE), which is defined as

$$\overline{\text{NRMSE}}(y, u) = \frac{1}{n_y} \sum_{m=1}^{n_y} \text{NRMSE}_m(y, u) \tag{54a}$$

with

$$\begin{aligned}&\mathrm{NRMSE}_m(y,u):\\&=\frac{1}{\sigma_m}\sqrt{\frac{1}{M_{\mathrm{test}}h_{\mathrm{test}}}\sum_{i=1}^{M_{\mathrm{test}}}\sum_{k=0}^{h-1}((y_m^k)_i-(\hat{y}_m^k)_i)^2}\\&\quad\text{for } m=1,\dots,n_y,\ (y,u)\in\mathcal{D}_{\mathrm{test}},\end{aligned}\tag{54b}$$

where M_{test} is the number of sequences from the data set $\mathcal{D}_{\mathrm{test}}$, h_{test} the prediction horizon of the test trajectories and $\sigma_m=\sqrt{\sum_{i=1}^{M}\sum_{k=0}^{N-1}(((y_m^k)_i-\bar{y}_m))}$ refers to the standard deviation of the output channel y_m with the mean $\bar{y}_m=\frac{1}{MN}\sum_{i=1}^{M}\sum_{k=0}^{N-1}(y_m^k)_i$. The statistics are derived from the training data. The NRMSE_m describes the accuracy of the output y_m of the dynamical system, which is useful to identify outputs that the model does not precisely recover. This empirical property is based on the samples in the datasets. Normalisation is required to make each output equally important. Quadratic error measures are commonly used in system identification to verify if the model under test can accurately predict the unknown system. More details can be found in the book by Pillonetto et al. (2022).

7.2.1.1 *In-distribution (ID) evaluation.*

ID data describes the test dataset $\mathcal{D}_{\mathrm{test}}^{\mathrm{ID}}$ that stems from the same distribution as the training and validation dataset and is a subset of $\mathcal{D}$.

7.2.1.2 *Out-of-distribution (OOD) evaluation.*

To test the generalisation capabilities of the identification method, we additionally evaluate OOD datasets, denoted by $\mathcal{D}_{\mathrm{test}}^{\mathrm{OOD}}$. For our experiments in Section 7, the OOD dataset is either provided or synthetically generated. Details on how these OOD datasets are generated can be found in the corresponding description of the numerical experiment. The intuition behind the OOD datasets is that the input trajectories have different characteristics than the input trajectories in the training dataset. This can, for example, stem from higher frequencies or larger amplitudes. The OOD evaluation indicates if the identification method recovered the dynamics of the true underlying system (1). We indicate the OOD dataset by a superscript. If multiple OOD datasets are available, we report the average prediction error across all datasets in the table and provide a plot with the input trajectory variation on the x-axis and the prediction error on the y-axis. If only one OOD dataset is available, the table includes the corresponding error value, but no plot is provided.

7.2.2 Input–output stability

One of the main advantages of the hybrid model architecture is guaranteed ℓ_2-stability, which results in reliable simulation outputs. This guarantee is independent over prediction horizon and the chosen input data. To showcase this, we optimise input sequences u to achieve a large amplification, e.g. a large ℓ_2-gain with the following optimisation problem:

$$\gamma_*^2:=\sup_u\frac{\|\mathcal{S}_\theta(u,\xi^0)\|^2}{\|u\|^2},\quad u\in\ell_2,\ \xi^0=0.\tag{55}$$

We run gradient ascent optimisation for 1000 steps and report the highest value as γ_*. This value is an empirical upper bound on the model's ℓ_2-gain. This metric was also considered in the work by Revay et al. (2020).

7.2.2.1 *Different ℓ_2-gains.*

So far, we have introduced two different values for the ℓ_2-gain of the identified model. We refer to γ as the upper bound on the ℓ_2-gain which is fixed before training and assumed to be known (Assumption 3.1). After training, we try to find large model amplifications (empirical ℓ_2-gains) and indicate them by γ_*. For our hybrid model, we know that $\gamma_*\leq\gamma$ due to the constraints that we imposed during training.

We will introduce two more ℓ_2-gains that indicate how tight our preset upper bound γ is for the trained parameters. We do this analysis for the `HybCRnn`(ours) and `CRnn` models by solving the constrained optimisation problem

$$\begin{aligned}\gamma_{\mathrm{opt}}:=\min_{\theta,\mathcal{X},\Lambda}\quad&\gamma\\\text{such that}\quad&\text{(28) is satisfied for some }\Lambda\in\mathbf{\Lambda}_{DG}\text{ or}\\&\Lambda\in\mathbf{\Lambda}_{ZF}\text{ and }\mathcal{X}\succ 0\end{aligned}\tag{56}$$

for the two different multiplier sets Λ_{DG} and Λ_{ZF} which results in $\gamma_{\mathrm{opt},DG}$ and $\gamma_{\mathrm{opt},ZF}$. As mentioned in Section 4, since $\Lambda_{DG}\subset\Lambda_{ZF}$ we expect $\gamma_{\mathrm{opt},ZF}\leq\gamma_{\mathrm{opt},DG}$ to hold when solving (56). Note that (56) is convex and can hence be solved efficiently with off-the-shelf solvers.

For the unconstrained models `LSTM` and `ReLiNet` and the exponential stable model `StableReLiNet`, we are not aware of any method in order to calculate a tight upper bound on the ℓ_2-gain and, therefore, only provide the empirical ℓ_2-gain γ_*.

7.3 Experiments

The datasets used in the experiments are publicly available and can be found on DaRUS (damped pendulum Frank, 2025b, coupled mass-spring-damper (MSD) system Frank, 2025c and nonlinear ship motion in open water Baier & Staab, 2022). In addition, the code for producing the damped pendulum and the coupled MSD system can be found at Frank (2025a).

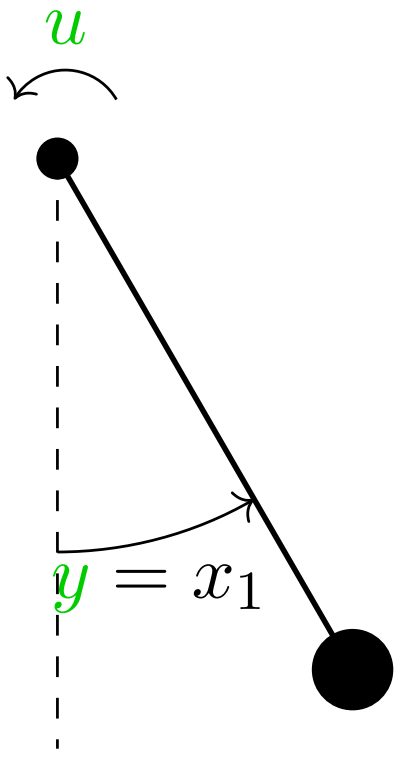

Figure 3. Pendulum system that is actuated by force u at the mounting point, the system's output y is the angle with respect to the negative y-axis.

7.3.1 Damped pendulum

We evaluate our hybrid architecture on a damped pendulum driven by an external force. The true continuous system equations are given by

$$\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ e(t) \end{pmatrix} = \begin{pmatrix} x_2(t) \\ -\eta \sin(x_1(t)) - \varsigma x_2(t) + d(t) \\ x_1(t) \end{pmatrix}.$$

where $x_1(t)$ refers to the pendulums angle and $x_2(t)$ to the angular velocity. We set the damping parameter to $\varsigma = \frac{1}{100}$ and $\eta = 1$, which captures the mass, pole length, and gravitational constant. An illustration is shown in Figure 3.

To generate input trajectories u we follow the work of Lu et al. (2021), Hu et al. (2024) and sample from a multivariate-normal distribution

$$u \sim \mathcal{N}(0, \mathcal{K}) \quad \text{with } \mathcal{K} \succ 0. \tag{57}$$

The idea is to get smooth input trajectories that follow from a high correlation between consecutive time steps. Therefore, the covariance kernel is chosen to be the radial-basis function (RBF), which is defined as $k_l(z_1, z_2) = \exp(-\frac{(z_1-z_2)^2}{2l^2})$ and characterised by the length scale l. The numerator of the RBF leads to a decaying correlation if the distance between data points $z_1 - z_2$ increases. We choose the datapoints to be the discrete time steps $z = \{(z^k)\}_{k=0}^{N-1}$ with $z^k = k\delta$ for $k = 0, \ldots, N-1$, resulting in the covariance matrix $\mathcal{K} = k_l(z, z)$. In this experiment, we choose the step size $\delta = 0.01$ and a trajectory length $N = 150$. For sampling the input trajectory u, we use a random normal trajectory v where $v^k \sim \mathcal{N}(0, I)$ and multiply it with the Cholesky factorisation of the kernel matrix $\mathcal{K}$, e.g. $u = Lv$ where $\mathcal{K} = LL^\top$. This process turns the uncorrelated trajectory v into a correlated and smooth input trajectory u where $\mathcal{K}$ specifies the correlation. Note that in this example, we only have one input channel $n_u = 1$ for systems with multiple inputs each channel has to be sampled separately.

Remark 7.1: Sampling from a multivariate-normal distribution with an RBF kernel is closely related to low-pass filtering white-noise, which is frequently used in classical system identification literature, e.g. Ljung (1999) and Pintelon and Schoukens (2012).

The output is obtained by integrating the true nonlinear dynamics with an iterative Runge-Kutta method. The numerical solver is evaluated at the discrete time steps that are given by the input trajectory. This leads to output trajectories y with the same discretisation as the input u. We add white gaussian noise to the output with mean zero and standard deviation $\sigma^2 = 0.03$, this refers to a signal-to-noise ratio of approximately 30 dB.

We create a dataset of $M = 20{,}000$ input–output trajectories and split it into 50% training, 10% validation, and 40% test data. The initial state is set to $x^0 = \binom{x_1(0)}{x_2(0)} = \binom{0}{0}$. We train the models on a horizon $h_{\text{train}} = 100$, evaluate on $h_{\text{test}} = 400$ and set the initialiser window to $w = 40$ (lines 7 and 8 in Algorithm 1). The initial learning rate is set to $\tau = 10^{-2}$ and the barrier parameter to $\kappa = 10^{-3}$.

The discretised system matrices in the form of (2) are obtained by the Euler method with the same discretisation step size δ as the input trajectory, which results in:

$$\begin{aligned} \begin{pmatrix} x_1^{k+1} \\ x_2^{k+1} \end{pmatrix} &= \begin{pmatrix} 1 & \delta \\ -k\delta & 1-\varsigma \end{pmatrix} \begin{pmatrix} x_1^{k+1} \\ x_2^{k+1} \end{pmatrix} + \begin{pmatrix} 0 \\ \delta \end{pmatrix} u^k \\ &\quad + \begin{pmatrix} 0 \\ 1 \end{pmatrix} N_f(x^k) \\ y^k &= \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1^k \\ x_2^k \end{pmatrix} + w_n^k \\ N_f(x^k) &= k\delta x_1^k + \sin(x_1^k). \end{aligned} \tag{58}$$

Table 1. Accuracy evaluation of the damped pendulum.

Model	$\mathcal{D}_{\text{test}}^{\text{ID}}$	$\mathcal{D}_{\text{test}}^{\text{OOD}}$
LSTM	0.0209	0.175
cLSTM	0.0214	0.126
ReLiNet	0.0237	0.189
StableReLiNet	**0.0134**	***0.124***
CRnn	0.599	0.697
HybCRnn(ours)	**0.0187**	**0.0399**

Note: We indicate the best values in **bold** and the second best in **bold+italic**.

We extract $A_{\text{lin}} = \begin{pmatrix} 1 & \delta \\ -k\delta & 1-\varsigma \end{pmatrix}$, $B_{\text{lin}} = \begin{pmatrix} 0 \\ \delta \end{pmatrix}$, $C_{\text{lin}} = (1 \;\; 0)$ and $D_{\text{lin}} = 0$ from (58) for our hybrid architecture. Additionally, we will make use of the following structural knowledge: (i) The nonlinear part does not have memory (e.g. $g = 0$ in (2)) resulting in $\tilde{B}_{\text{lin},2} = 0$. (ii) The nonlinearity $N_f(x^k)$ is only acting on x_2 which yields $\tilde{B}_{\text{lin},3} = (0 \;\; 1)^\top$. (iii) The output does not directly depend on the nonlinearity. Therefore, we set $\tilde{D}_{\text{lin},2} = \tilde{D}_{\text{lin},3} = 0$.

For OOD evaluation we scale the input trajectories $u^{\text{OOD}} = su$, where u is drawn from a multivariate-normal distribution as in (57) and keep the length scale of the RBF fixed. This leads to higher amplitudes in the input data. We pick scaling factors s from $\{0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$ and generate $M^{\text{OOD}} = 200$ sequences with $N^{\text{OOD}} = 500$ steps. The evaluation of the models on ID and OOD data is shown in Table 1. We observe that the hybrid architecture outperforms all baseline models when evaluated on OOD datasets and that it achieves comparable prediction accuracy on ID data.

In Figure 7 we show the absolute error between the model's output and the true output from the OOD dataset with scaling factor $s = 3.5$ for one sample trajectory. The corresponding input trajectory u is shown in Figure 6. The error in Figure 7 shows that the models without parametric constraints make accurate predictions for the horizon of the training dataset. Only the hybrid architecture stays close to the true system output for longer horizons h. This observation supports our hypothesis that prior-system knowledge leads to better generalisation capabilities of the model.

For the CRnn model, we did not find hyperparameters such that the training converged, which can be seen by the comparable low prediction accuracy for the training dataset and almost equal accuracy values for the different OOD datasets. This is visualised in the OOD evaluation in Figure 4. Additionally, we see that our hybrid model generalises better to inputs that are drawn from different distributions than the training dataset.

Next, we analyse the input–output stability of the models. Before training, we fixed the upper bound on the ℓ_2-gain to $\gamma = 10$ for the HybCRnn(ours) and $\gamma = 100$ for the CRnn model. The first observation is that the LSTM model has the largest empirical ℓ_2-gain that is two

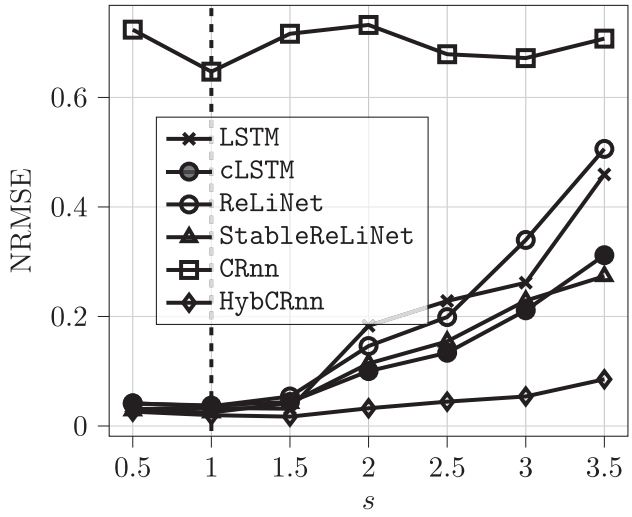

Figure 4. OOD evaluation for different scalings of the input trajectory. The vertical dashed line indicates the scaling value s that was used to generate the ID test set.

orders of magnitude higher than the upper bound of the HybCRnn(ours) and one order of magnitude higher than the upper bound of the CRnn. Even though the ReLiNet models do not provide an upper bound on the ℓ_2-gain, the empirical evaluation shows that they are bounded in the same order of magnitude as the upper bound of the HybCRnn(ours) model. The second observation is that for the HybCRnn(ours), we get tighter bounds between the empirical upper bound and γ compared to the CRnn. This effect is visualised in Figure 5.

7.3.2 Coupled MSD system

As a second example, we identify the dynamics of a coupled MSD system with four masses. It is a SISO system, where the input u is a force applied to the first mass. At the output y, the displacement of the fourth mass is measured. The system is described by a nonlinear differential equation with 8 physical states that represent the position and velocity of each mass. An illustration of the system is shown in Figure 8. The nonlinearity stems from a piecewise linear force profile of each spring. We excite the system with a piecewise-constant input trajectory, where each input value is held constant for a duration randomly drawn from a uniform distribution over the interval $[80, 120]$. The input values themselves are sampled independently from a uniform distribution within the range $[-\sigma_u, \sigma_u]$. The trajectory length is set to $N = 7500$ and we use a discretisation step size of $\delta = 0.2$.

The output trajectories are obtained by integrating the true nonlinear dynamics as above and evaluate the models at each discretisation step to get an output trajectory with the same size as the input trajectory. At the output we add white Gaussian nosie with a standard deviation

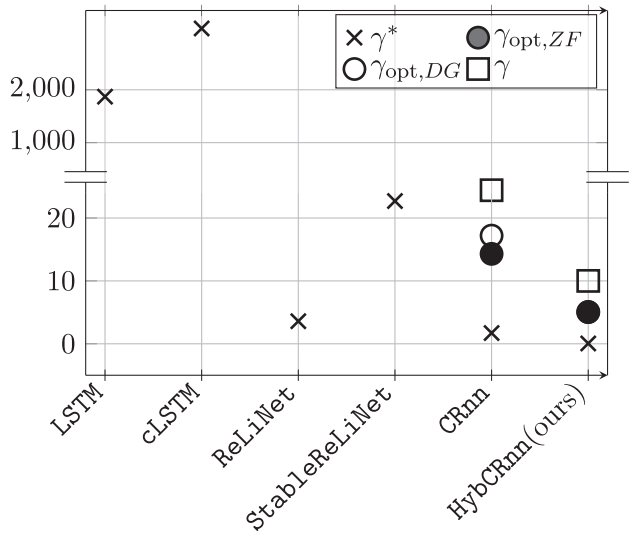

Figure 5. Stability evaluation of the different models for the damped pendulum example, a high value on the *y*-axis refers to a large amplification of the input trajectory. The empirical ℓ_2-gain (blue cross) for the unconstrained models do not have an upper bound (red cross). The static Zames–Falb multiplier do not lead to a smaller ℓ_2-gain bound when compared to the diagonal multiplier (yellow and green cross overlap). In addition, we see that our hybrid model has a smaller gap between the empirical ℓ_2-gain and the smallest upper bound (according to (56)) compared to the constrained RNN.

of $\sigma^2 = 0.03$. The dataset is then subdivided into 60%-training, 10%-validation and 30% testing. During training, the simulation horizon is $h_{\text{train}} = 1000$ steps (200 s) and we test the model for a horizon of $h_{\text{test}} = 5000$ steps (1000 s). The initialisation window is set to $w = 50$ for training and testing. We set the initial learning rate to $\tau = 10^{-2}$ and the barrier parameter to $\kappa = 10^{-3}$.

The linearised system matrices A_{lin}, B_{lin}, C_{lin}, and D_{lin} and the system's parameters can be found in the Appendix A.3. We infer from the true nonlinear dynamics that the residual model has no memory. Therefore, we set $\tilde{B}_{\text{lin},2} = \tilde{D}_{\text{lin},2} = 0$. We also know that the position of each mass directly depends on the velocity and, therefore, assume that it is sufficient to correct the velocity values leading to $\tilde{B}_{\text{lin},3} = \text{blkdiag}(\binom{0}{1}, \binom{0}{1}, \binom{0}{1}, \binom{0}{1})$. Since we have the position of the last mass in our state vector of the linearised system we do not have a dependency between $\hat{y}$ and u_r and, therefore, can set $\tilde{D}_{\text{lin},3} = 0$.

For ID data we set $\sigma_e = 1.5$, for OOD data we pick σ_e from $\{0.5, 1, 2, 2.5, 3.5, 4, 4.5, 5\}$ and set $N_{\text{OOD}} = 1200$ and $M_{\text{OOD}} = 50$. This system was also used in the work by Revay et al. (2020) where they evaluate the incremental ℓ_2-gain.

In Table 2, we show the prediction accuracy on ID test data and the mean over all OOD datasets. The results

Table 2. Accuracy evaluation of the coupled MSD system, we show the NRMSE values of the corresponding datasets.

Model	$\mathcal{D}_{\text{test}}^{\text{ID}}$	$\mathcal{D}_{\text{test}}^{\text{OOD}}$
`LSTM`	**0.0207**	0.491
`cLSTM`	0.021706	0.236
`ReLiNet`	0.0546	0.228
`StableReLiNet`	0.0842	0.232
`CRnn`	0.0969	0.222
`HybCRnn`(ours)	**0.0573**	**0.155**

Note: For OOD evaluation we used the mean over all OOD datasets. We indicate the best values in **bold** and the second best in **bold+italic**.

show that the hybrid model architecture achieves comparable results on ID data and outperforms all baseline models on OOD data. This is also visualised in Figure 9 where on the *x*-axis, we show the values of σ_e, which is used to generate the OOD datasets.

The different γ values are shown in Figure 10. Compared to the `CRnn` model, our hybrid architecture achieves ℓ_2-values that are closer to the preset upper bound. It is worth mentioning that for the unstable models `LSTM`, `ReLiNet`, we can find an input sequence that leads to a high amplification indicated by the empirical γ^*. However, the `StableReLiNet`, which does not provide input–output guarantees but ensures exponential stability, reaches a comparable empirical ℓ_2-gain.

7.3.3 Nonlinear ship motion in open water

A more realistic evaluation is presented on the four degrees-of-freedom (DOF) ship dataset (Baier & Staab, 2022). The dataset is generated from a simulation model that consists of a nonlinear motion model of the ship (Fossen, 2011), a wind model (Isherwood, 1972), a model for wind-induced waves (Hasselmann et al., 1973) and actuator models.

The input u consists of environmental measurements from a wind sensor and includes the wind force V_w and direction α_x, α_y. Additionally, we have access to the input signals sent to the actuators, the rotation rate n, and the rudder angle of the left and right rudder δ_l and δ_r. Note that the ship has two rudder propellors with the same rotation rate. Therefore, we will only use one rotation rate in the input. We use one angular value for the rudder angle because they do not exceed 180 degrees. Thus, the input consists of six dimensions $u^k = [n^k, \delta_l^k, \delta_r^k, V_w^k, \alpha_x^k, \alpha_y^k]^\top$. The trajectories are sampled at one second, and $M = 96$ measurements of one hour are taken, leading to trajectories of length $N = 3600$. The dataset $\mathcal{D}$ is divided into three subsets: 60% for training, 10% for validation, and 30% for testing. The window is set to $w = 50$, and the training horizon to $h_{\text{train}} = 50$. We evaluate the model on horizon $h_{\text{test}} = 900$, referring to a 15-minute simulation of ship movement. The initial

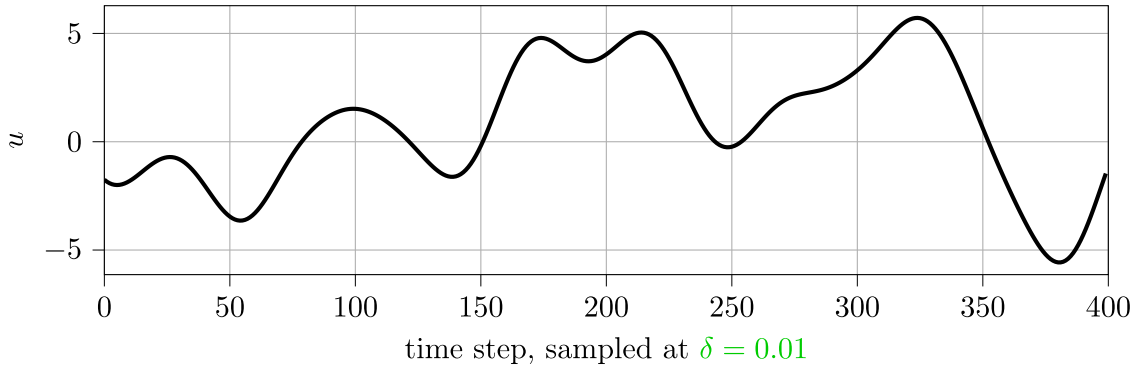

Figure 6. Input trajectory of damped pendulum for OOD dataset scaled with $s = 3.5$.

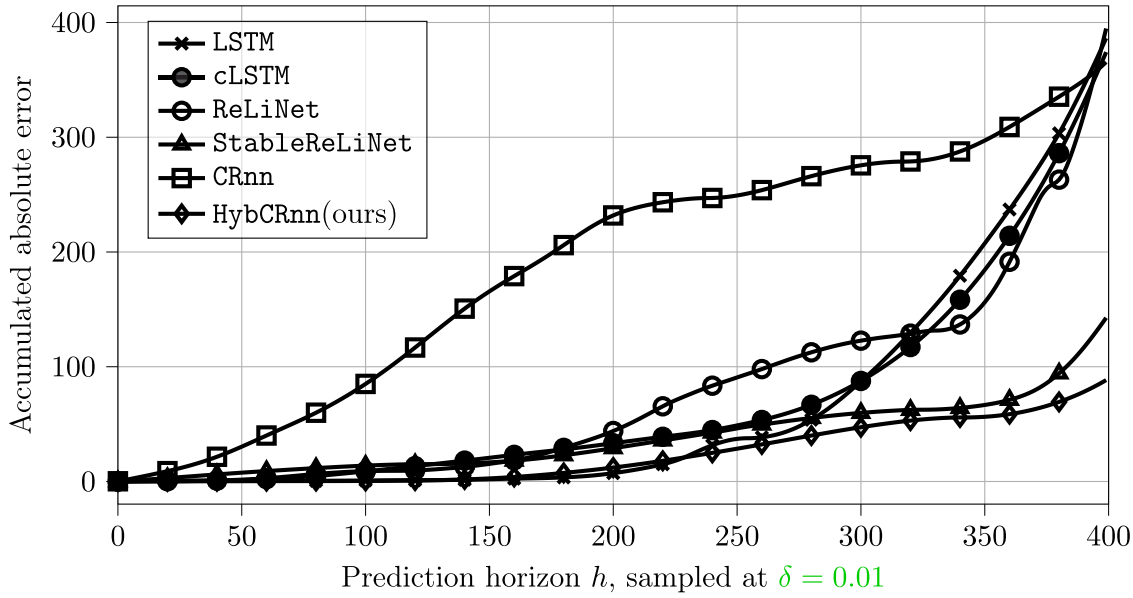

Figure 7. Accumulated absolute error with respect to true output y for OOD datasets scaled with $s = 3.5$. The models are excited with input trajectory from Figure 6.

learning rate is $\tau = 0.0025$, and the initial decay rate is $\kappa = 10^{-3}$. The accuracy evaluation is shown in Table 3.

Remark 7.2: The dataset is synthetically generated but matches a real-world scenario, where only inputs that are reasonable to measure on a real ship during regular operation are present. The dataset is generated in an open-loop behaviour where random maneuvers are taken. Gaussian sensor noise is added to the output measurements. More details on the dataset can be found in Baier and Staab (2022).

The output consists of the forward and sideward velocities s, v, and the rotation rate of the roll and yaw angle p, r

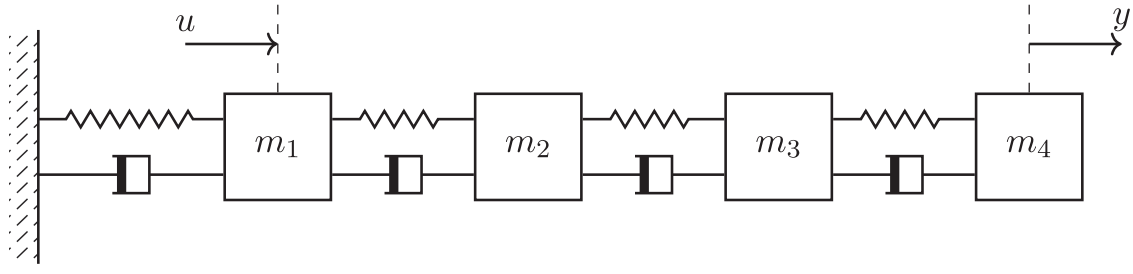

Figure 8. Coupled mass-spring-damper system with four masses. The input u is a force applied to the first mass, and the output y is the displacement of the fourth mass.

Table 3. Accuracy evaluation of the different models on the ship dataset (Baier & Staab, 2022), we show each output of the ship individually and calculate a normalised mean in the column $\mathcal{D}_{\text{test}}^{\text{ID}}$.

Model	s	v	p	r	$\mathcal{D}_{\text{test}}^{\text{ID}}$	$\mathcal{D}_{\text{test}}^{\text{OOD}}$
LSTM	0.124	0.228	1.272	0.222	0.462	0.562
cLSTM	0.439	0.713	1.000	0.656	0.702	1.005
ReLiNet	***0.117***	***0.213***	**0.982**	***0.199***	**0.378**	**0.422**
StableReLiNet	0.119	**0.176**	1.106	**0.183**	***0.396***	***0.486***
CRnn	0.173	0.424	0.991	0.427	0.504	0.642
HybCRnn(ours)	**0.050**	0.243	***1.026***	0.390	0.427	0.534

Note: We indicate the best values in **bold** and the second best in **bold+italic**.

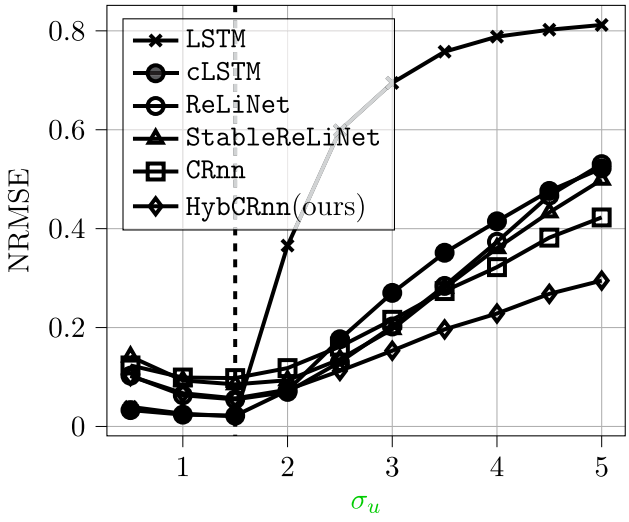

Figure 9. Out-of-distribution evaluation for coupled mass-spring-damper system. The x-axis shows the value of σ_e used to generate the OOD datasets and the vertical dashed line indicates the σ_e value that was used for the ID dataset.

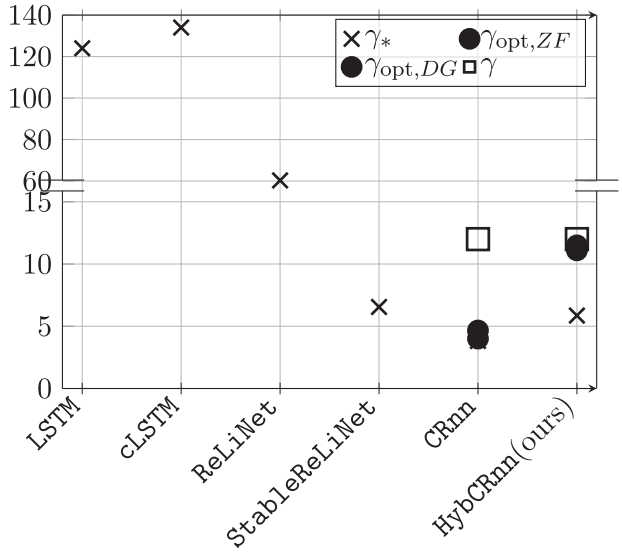

Figure 10. Stability evaluation of the different models for the coupled mass-spring-damper example, a high value on the y-axis refers to a large amplification of the input trajectory. The unconstrained models do not provide an upper bound on the ℓ_2-gain (red crosses) and the empirical ℓ_2-gain exceeds the values of the constrained models (blue crosses). We observe that the empirical ℓ_2-gain and the analytical upper bounds (green and yellow crosses) are closer to the predefined upper bound (red crosses) compared to the CRnn model.

leading the four output dimensions $y^k = [s^k, v^k, p^k, r^k]^\top$ with the same length as the input sequence. An Illustration of the ship's outputs is shown in Figure 11.

We also have $M_{\text{OOD}} = 29$ hours of OOD measurements, where the maneuvers are executed more frequently, and the input range of the actuators is extended.

We assume to have access to a linear motion model of the four DOF ship that approximates the nonlinear ship model (S1). Such models can, for example, be found in Fossen (2011). The linear model maps forces into ship velocities. However, the dataset does not provide forces but rather environmental inputs that are translated to forces that act on the ship. Therefore, the residual model is required to learn not only the error of the linear model but also an input model that maps the input measurements to forces. More details on the linear ship model can be found in Appendix A.4, including an illustration of the environmental model that visualises the above mentioned interconnection.

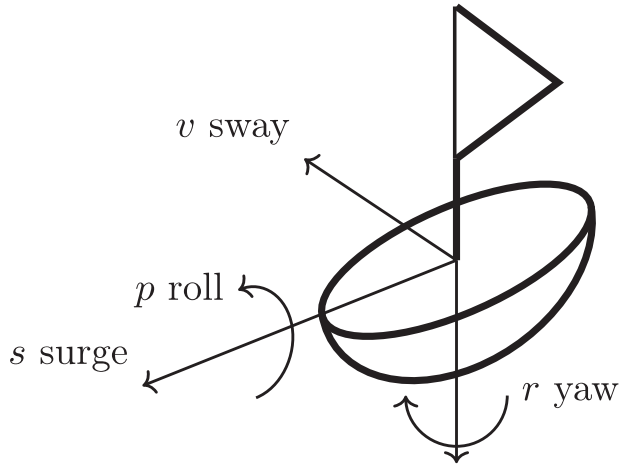

Figure 11. Four degrees-of-freedom motion model of a ship.

Our hybrid architecture allows for this type of structural information by choosing.

$$B_{\text{lin},2} = \left(\; I \;\middle|\; B_{d,\text{ship}} \;\middle|\; 0 \; \right), \quad D_{\text{lin},2} = \left(\; 0 \;\middle|\; 0 \;\middle|\; I \; \right) \tag{59}$$

and $B_{\text{lin}} = 0$. This choice reflects that the linear dynamics are not directly dependent on the environmental input measurements ($B_{\text{lin}} = 0$). Since we expect the output of the RNN to contain the forces, we set $\tilde{B}_{\text{lin},3}$ to the known input matrix of the linear model. We also assume that the environmental model has some memory effects and set $\tilde{B}_{\text{lin},2} = I$. For the direct feedthrough term $\tilde{D}_{\text{lin},3}$, we do not have any prior knowledge and also set it to be the identity, allowing the RNN to learn that part.

The accuracy evaluation is shown in Table 3. Our hybrid architecture does not reach the same level of accuracy compared to the linear switching models for ID and OOD evaluation. This behaviour may be attributed to the system having multiple inputs and outputs. In this case, the ℓ_2-gain may not help to improve generalisation. More detailed investigation on individual ℓ_2-gains for specific input–output channels for multiple-input-multiple-output (MIMO) systems will be left for future research. However, compared to the `CRnn` that does not make use of the linearisation, we observe higher accuracies for ID as well as OOD evaluation.

Before training, we fix the upper bound on the energy gain (ℓ_2-gain) to be $\gamma = 10$ (S2). The different ℓ_2-values are illustrated in Figure 12. As before, we find input sequences that lead to a large output amplification, shown by a high empirical γ_* value for the `LSTM` model. Additionally, we observe that the linear switching models `ReLiNet` and `StableReLiNet` have an empirical γ_* value that is below γ , so even though they do not provide a guaranteed upper bound, the models are making reliable predictions. When comparing the `HybCRnn`(ours) and `CRnn` results, we observe that we have slightly tighter bounds for the `HybCRnn`(ours) model.

8. Convex and non-convex constraints

In Section 5.1 we introduced new parameters with convex constraints. These constraints allow us to solve projection problems such as (53) or analysis problems such as (55) with off-the shelf solver but it remains unknown if they help to find parameters that are more accurate when solving the non-convex objective in Problem (39). In this Section, we will address this question by empirically evaluating the prediction accuracy of the resulting models when trained with non-convex and convex constraints. The training method that we introduced in Section 6.5 to find optimal stimulation parameters θ for the hybrid model architecture (8) works similarly for the non-convex constraints (28).

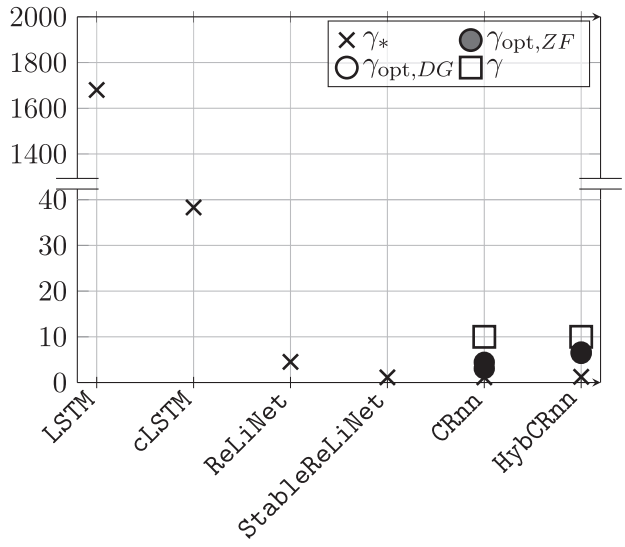

Figure 12. Stability evaluation for ship dataset (Baier & Staab, 2022).

We will use the coupled-MSD experiment for our evaluation.

8.1 Learning θ parameters with non-convex constraints

In the first approach, we use constraint (28) as regularisation in the optimisation problem. We set the initial parameters θ^0 to zero to ensure feasibility. The hybrid model then reduces to the linear model (4), which we assume to be ℓ_2 stable with $\gamma > 0$ (Assumption 3.1).

In the non-convex case we will denote the LMI from (28) as $\tilde{\mathbf{F}}$, the loss function then reads as

$$\begin{aligned} &\mathcal{L}_{\theta,\mathcal{X},\Lambda}(\hat{y}, y) \\ &\quad = \frac{1}{h}\sum_{k=0}^{h-1} \|(y^k) + (\hat{y}^k)\|^2 \\ &\qquad + \kappa \left(-\log\det(\Lambda) - \log\det(\mathcal{X}) - \log\det(-\tilde{\mathbf{F}}) \right) \\ &\qquad \text{with } \hat{y} = \mathcal{S}_\theta(u, \xi^0), \end{aligned} \tag{60}$$

we use the gradient of $\mathcal{L}$ to update the parameters. After an update step, we check the feasibility of the new parameters. If they are not feasible, we use a backtracking line search algorithm to find the closest feasible set. That way, we ensure feasibility during the entire training (see Algorithm 1).

8.2 Learning ω parameters with convex constraints

We will compare the non-convex training against our proposed training method with transformed parameters, see Section 5.1.

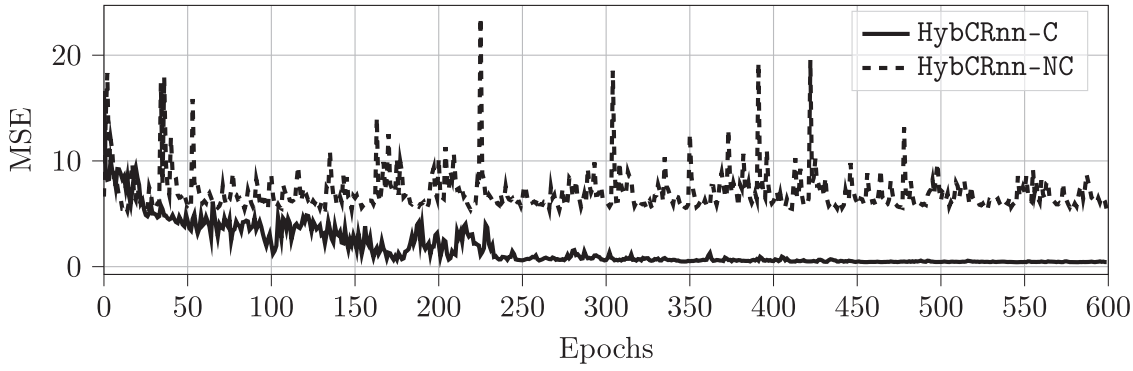

Figure 13. MSE on validation dataset $\mathcal{D}_{\text{val}}$, after each training epoch the quadratic error is measured.

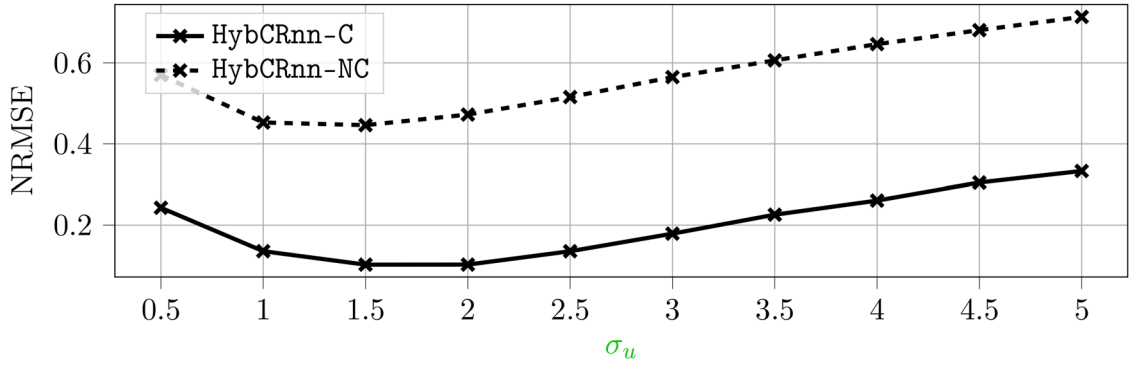

Figure 14. Evaluation of OOD datasets. The *x*-axis shows the absolute input amplitude used to generate random static input sequences.

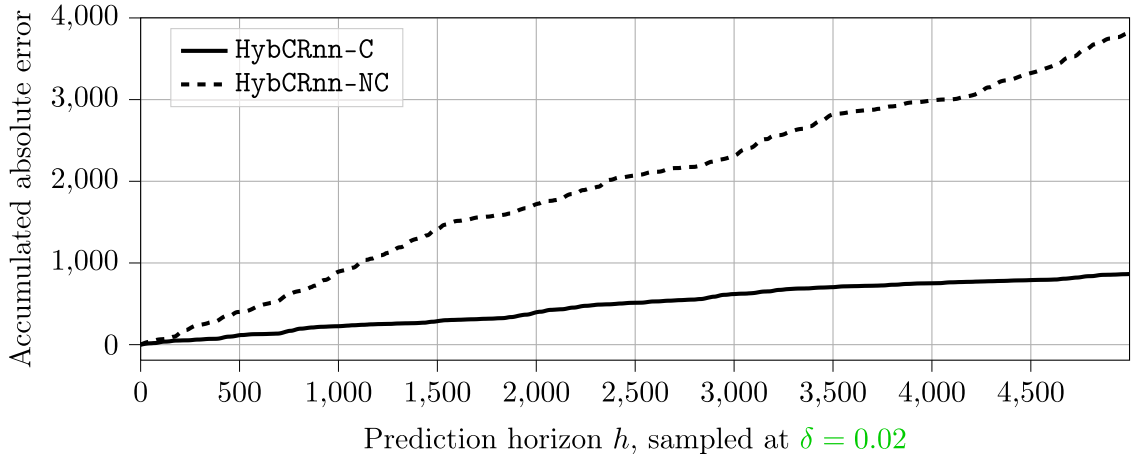

Figure 15. Accumulated output error compared to the true output *y*, when the model is excited with an input trajectory from the test dataset and trained with convex (`-C`) and non-convex (`-NC`) constraints.

The constraints are LMIs in the decision variables. The loss function then reads as

$$\begin{aligned}&\mathcal{L}_{\omega,\mathbf{X},\Lambda}(\hat{y},y)\\&\quad=\frac{1}{h}\sum_{k=0}^{h-1}\|(y^k)+(\hat{y}^k)\|^2\\&\quad\quad+\kappa\left(-\log\det(\Lambda)-\log\det(\mathbf{X})-\log\det(-\mathbf{F})\right)\\&\quad\text{with }\hat{y}=\mathcal{S}_\theta(u,\xi^0),\end{aligned}\tag{61}$$

We will denote the model with **N**on-**C**onvex constraints by `HybCRnn-NC` and the model with **C**onvex constraints by `HybCRnn-C`.

The plot in Figure 13 shows the validation loss during training.

Table 4. Evaluation of models trained with convex and non-convex constraints.

Model	$\mathcal{D}_{\text{test}}^{\text{ID}}$	$\mathcal{D}_{\text{test}}^{\text{OOD}}$	γ_*	$\gamma_{\text{opt},DG}$	$\gamma_{\text{opt},ZF}$	γ
`HybCRnn-NC`	0.449	0.567	0.98	0.99	0.99	12.0
`HybCRnn-C`	**0.104**	**0.202**	1.61	2.26	2.25	12.0

When evaluating the trained models on OOD datasets, we see that the model that is trained with convex constraints outperforms the model that is trained with non-convex constraints (see Figure 14 and 15 for details). The values are shown in Table 4. We leave it as an interesting open research question to confirm this empirical result theoretically.

9. Conclusion

In this work, we introduced a hybrid model architecture that leverages side information from the true system, combined with a RNN, for nonlinear system identification. The model is trained using input–output measurements, and the parameters are constrained to ensure a predefined energy gain (ℓ_2-gain).

The experiments from Section 7.3 are systems with one equilibrium point where the linear approximation model makes accurate predictions. For systems with multiple equilibria, the hybrid model approach presented in this work is not expected to make more accurate predictions than black-box approaches. Considering multiple linear approximation models as prior system knowledge and deriving stability conditions is left as future work.

Since ISS implies a bounded ℓ_2-gain, we expect the worst case amplification γ^* to drop when the parametric constraints are enforced. In our experiments, when we enforced the constraints by parameter scaling, the performance dropped to a level comparable to predicting zero, therefore, it is not reported.

Our results show that the hybrid architecture outperforms all the baselines on SISO systems in terms of generalisation capabilities. We could not outperform the switching linear models for the more realistic ship-motion dataset. For MIMO systems, more detailed input–output properties are required to transfer the generalisation capabilities of the architecture to a broader range of nonlinear systems.

In addition to prediction accuracy, we also evaluated two classes of multipliers, where we see in the analysis that static Zames–Falb multiplier leads to a smaller upper bound on the energy gain for all constrained models on all datasets. However, incorporating Zames–Falb multipliers during training did not lead to an improvement in prediction accuracy. This unexpected outcome will be addressed in future research.

When we compare our hybrid model to other constrained methods that do not use the linearised system dynamics, we see significant improvement in prediction accuracy on ID and OOD datasets.

Note

1. https://github.com/AlexandraBaier/deepsysid

Acknowledgments

We acknowledge the support of the Stuttgart Center for Simulation Science (SimTech).

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This research is funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016. We acknowledge the support of the Stuttgart Center for Simulation Science (SimTech).

References

Andersson, C., Ribeiro, A. H., Tiels, K., Wahlström, N., & Schön, T. B. (2019). Deep convolutional networks in system identification. In *2019 IEEE 58th Conference on Decision and Control (CDC)* (pp. 3670–3676). IEEE.

Arcak, M., Meissen, C., & Packard, A. (2016). *Networks of dissipative systems: Compositional certification of stability, performance, and safety*. Springer.

Baier, A., Aspandi, D., & Staab, S. (2023). ReLiNet: Stable and explainable multistep prediction with recurrent linear parameter varying networks. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence* (pp. 3461–3469). International Joint Conferences on Artificial Intelligence. https://www.ijcai.org/proceedings/2023/

Baier, A., & Staab, S. (2022). *A simulated 4-DOF ship motion dataset for system identification under environmental disturbances*. DaRUS. https://doi.org/10.18419/darus-2905

Beintema, G. I., Schoukens, M., & Tóth, R. (2023). Deep subspace encoders for nonlinear system identification. *Automatica*, *156*, Article 111210. https://doi.org/10.1016/j.automatica.2023.111210

Bonassi, F., Farina, M., Xie, J., & Scattolini, R. (2022). On recurrent neural networks for learning-based control: Recent results and ideas for future developments. *Journal of Process Control*, *114*, 92–104. https://doi.org/10.1016/j.jprocont.2022.04.011

Bonassi, F., Terzi, E., Farina, M., & Scattolini, R. (2020). LSTM neural networks: Input to state stability and probabilistic safety verification. In *Learning for dynamics and control* (pp. 85–94). Proceedings of Machine Learning Research.

Boyd, S., Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Brunton, S. L., Nathan Kutz, J., Manohar, K., Aravkin, A. Y., Morgansen, K., Klemisch, J., Goebel, N., Buttrick, J., Poskin, J., Blom-Schieber, A. W., Hogan, T., & McDonald, D. (2021).

Data-driven aerospace engineering: Reframing the industry with machine learning. *AIAA Journal*, *59*(8), 2820–2847. https://doi.org/10.2514/1.J060131

Byrnes, C. I., & Lin, W. (1994). Losslessness, feedback equivalence, and the global stabilization of discrete-time nonlinear systems. *IEEE Transactions on Automatic Control*, *39*(1), 83–98. https://doi.org/10.1109/TAC.9

D'Amico, W., La Bella, A., & Farina, M. (2023). An incremental input-to-state stability condition for a class of recurrent neural networks. *IEEE Transactions on Automatic Control*, *69*(4), 2221–2236. https://doi.org/10.1109/TAC.2023.3327937

De Oliveira, M. C., Geromel, J. C., & Bernussou, J. (2002). Extended H_2 and H_∞ norm characterizations and controller parametrizations for discrete-time systems. *International Journal of Control*, *75*(9), 666–679. https://doi.org/10.1080/00207170210140212

Fazlyab, M., Robey, A., Hassani, H., Morari, M., & Pappas, G. (2019). Efficient and accurate estimation of Lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems, 32*. Neural Information Processing Systems Foundation, Inc. (NeurIPS).

Fetzer, M., & Scherer, C. W. (2017). Absolute stability analysis of discrete time feedback interconnections. *IFAC-PapersOnLine*, *50*(1), 8447–8453. https://doi.org/10.1016/j.ifacol.2017.08.757

Forgione, M., Mejari, M., & Piga, D. (2022). Learning neural state-space models: Do we need a state estimator? *Preprint*. arXiv:2206.12928

Forgione, M., & Piga, D. (2020). Model structures and fitting criteria for system identification with neural networks. In *2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)* (pp. 1–6). Institute of Electrical and Electronics Engineers (IEEE).

Fossen, T. I. (2011). *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons.

Frank, D. (2025a). *Code for statesim*. DaRUS. https://doi.org/10.18419/DARUS-4769

Frank, D. (2025b). *Coupled mass-spring-damper system*. DaRUS. https://doi.org/10.18419/DARUS-4768

Frank, D. (2025c). *Damped pendulum*. DaRUS. https://doi.org/10.18419/DARUS-4770

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Gu, F., Yin, H., El Ghaoui, L., Arcak, M., Seiler, P., & Jin, M. (2022). Recurrent neural network controllers synthesis with stability guarantees for partially observed systems. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, pp. 5385–5394). Association for the Advancement of Artificial Intelligence (AAAI).

Hasselmann, K., Barnett, T. P., Bouws, E., Carlson, H., Cartwright, D. E., Enke, K., Ewing, J. A., Gienapp, A., Hasselmann, D. E., Kruseman, P., Meerburg, A., Müller, P., Olbers, D. J., Richter, K., Sell, W., & Walden, H. (1973). *Measurements of wind-wave growth and swell decay during the joint north sea wave project (JONSWAP). Ergaenzungsheft zur Deutschen Hydrographischen Zeitschrift, Reihe A*.

Hu, Z., Daryakenari, N. A., Shen, Q., Kawaguchi, K., & Karniadakis, G. E. (2024). State-space models are accurate and efficient neural operators for dynamical systems. *Preprint*. arXiv:2409.03231

Isherwood, R. (1972). Wind resistance of merchant ships. *The Royal Institution of Naval Architects*, *115*, 327–338.

Junnarkar, N., Arcak, M., & Seiler, P. (2024). Synthesizing neural network controllers with closed-loop dissipativity guarantees. *Preprint*. arXiv:2404.07373

Junnarkar, N., Yin, H., Gu, F., Arcak, M., & Seiler, P. (2022). Synthesis of stabilizing recurrent equilibrium network controllers. In *2022 IEEE 61st Conference on Decision and Control (CDC)* (pp. 7449–7454). Institute of Electrical and Electronics Engineers (IEEE).

Karpatne, A., Kannan, R., & Kumar, V. (2022). *Knowledge guided machine learning: Accelerating discovery using scientific knowledge and data*. CRC Press.

Khosravi, M., & Smith, R. S. (2023). Kernel-based identification with frequency domain side-information. *Automatica*, *150*, Article 110813. https://doi.org/10.1016/j.automatica.2022.110813

Kingma, D. P., & Ba, J. (2015). ADAM: A method for stochastic optimization. In Bengio, Y. & LeCun, Y. (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1412.6980

Koch, A., Berberich, J., & Allgöwer, F. (2021). Provably robust verification of dissipativity properties from data. *IEEE Transactions on Automatic Control*, *67*(8), 4248–4255. https://doi.org/10.1109/TAC.2021.3116179

Ljung, L. (1998). *System identification: Theory for the user*. Pearson Education. https://books.google.de/books?id = fYSrk4wDKPsC

Ljung, L. (1999). *System identification: Theory for the user* (2nd ed.). Prentice Hall.

Ljung, L., Chen, T., & Mu, B. (2020). A shift in paradigm for system identification. *International Journal of Control*, *93*(2), 173–180. https://doi.org/10.1080/00207179.2019.1578407

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, *3*(3), 218–229. https://doi.org/10.1038/s42256-021-00302-5

Lusch, B., Kutz, J. N., & Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, *9*(1), Article 4950. https://doi.org/10.1038/s41467-018-07210-0

Masubuchi, I., Ohara, A., & Suda, N. (1998). LMI-based controller synthesis: A unified formulation and solution. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, *8*(8), 669–686. https://doi.org/10.1002/(ISSN)1099-1239

Megretski, A., & Rantzer, A. (1997). System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, *42*(6), 819–830. https://doi.org/10.1109/9.587335

Miller, J., & Hardt, M. (2018). Stable recurrent models. *Preprint*. arXiv:1805.10369

Mohajerin, N., & Waslander, S. L. (2019). Multistep prediction of dynamic systems with recurrent neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(11), 3370–3383. https://doi.org/10.1109/TNNLS.5962385

Pauli, P., Berberich, J., & Allgöwer, F. (2022). Robustness analysis and training of recurrent neural networks using dissipativity theory. *at-Automatisierungstechnik*, *70*(8), 730–739. https://doi.org/10.1515/auto-2022-0032

Pauli, P., Gramlich, D., Berberich, J., & Allgöwer, F. (2021a). Linear systems with neural network nonlinearities: Improved

stability analysis via acausal Zames–Falb multipliers. In *2021 60th IEEE Conference on Decision and Control (CDC)* (pp. 3611–3618). Institute of Electrical and Electronics Engineers (IEEE).

Pauli, P., Koch, A., Berberich, J., Kohler, P., & Allgöwer, F. (2021b). Training robust neural networks using Lipschitz bounds. *IEEE Control Systems Letters*, *6*, 121–126. https://doi.org/10.1109/LCSYS.2021.3050444

Perez, T., Ross, A., & Fossen, T. (2006). A 4-DOF simulink model of a coastal patrol vessel for manoeuvring in waves. In *Proceedings of the 7th IFAC Conference on Manoeuvring and Control of Marine Craft* (pp. 1–6). IFAC.

Pillonetto, G., Aravkin, A., Gedon, D., Ljung, L., Ribeiro, A. H., & Schön, T. B. (2025). Deep networks for system identification: A survey. *Automatica*, *171*, Article 111907. https://doi.org/10.1016/j.automatica.2024.111907

Pillonetto, G., Chen, T., Chiuso, A., De Nicolao, G., & Ljung, L. (2022). *Regularized system identification: Learning dynamic models from data*. Springer Nature.

Pintelon, R., & Schoukens, J. (2012). *System identification: A frequency domain approach*. Wiley. https://books.google.de/books?id = 3lGJWtjGDzsC

Revay, M., Wang, R., & Manchester, I. R. (2020). A convex parameterization of robust recurrent neural networks. *IEEE Control Systems Letters*, *5*(4), 1363–1368. https://doi.org/10.1109/LCSYS.7782633

Revay, M., Wang, R., & Manchester, I. R. (2021). Recurrent equilibrium networks: Unconstrained learning of stable and robust dynamical models. In *2021 60th IEEE Conference on Decision and Control (CDC)* (pp. 2282–2287). Institute of Electrical and Electronics Engineers (IEEE).

Ribeiro, A. H., Tiels, K., Umenberger, J., Schön, T. B., & Aguirre, L. A. (2020). On the smoothness of nonlinear system identification. *Automatica*, *121*, Article 109158. https://doi.org/10.1016/j.automatica.2020.109158

Sastry, S. (2013). *Nonlinear systems: Analysis, stability, and control* (Vol. 10). Springer Science & Business Media.

Scherer, C. W. (2000). Robust mixed control and linear parameter-varying control with full block scalings. In *Advances in linear matrix inequality methods in control* (pp. 187–207). Society for Industrial and Applied Mathematics (SIAM). https://epubs.siam.org/doi/abs/10.1137/1.9780898719833.ch10

Scherer, C. W. (2022). Dissipativity, convexity and tight O'shea–Zames–Falb multipliers for safety guarantees. *IFAC-PapersOnLine*, *55*(30), 150–155. https://doi.org/10.1016/j.ifacol.2022.11.044

Scherer, C. W., Gahinet, P., & Chilali, M. (1997). Multiobjective output-feedback control via LMI optimization. *IEEE Transactions on Automatic Control*, *42*(7), 896–911. https://doi.org/10.1109/9.599969

Scherer, C. W., & Weiland, S. (2000). Linear matrix inequalities in control. *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 3(2).

Schoukens, J., & Ljung, L. (2019). Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, *39*(6), 28–99. https://doi.org/10.1109/MCS.2019.2938121

Schoukens, M. (2021). Improved initialization of state-space artificial neural networks. In *2021 European Control Conference (ECC)* (pp. 1913–1918). IEEE.

Suykens, J. A., Vandewalle, J. P., & De Moor, B. L. (1995). *Artificial neural networks for modelling and control of non-linear systems*. Springer Science & Business Media.

Veenman, J., Scherer, C. W., & Köroğlu, H. (2016). Robust stability and performance analysis based on integral quadratic constraints. *European Journal of Control*, *31*, 1–32. https://doi.org/10.1016/j.ejcon.2016.04.004

Wang, R., & Manchester, I. (2023). Direct parameterization of Lipschitz-bounded deep networks. In *International Conference on Machine Learning* (pp. 36093–36110). Proceedings of Machine Learning Research (PMLR).

Willems, J. C. (1972). Dissipative dynamical systems part I: General theory. *Archive for Rational Mechanics and Analysis*, *45*(5), 321–351. https://doi.org/10.1007/BF00276493

Willems, J. C., & Brockett, R. (1968). Some new rearrangement inequalities having application in stability analysis. *IEEE Transactions on Automatic Control*, *13*(5), 539–549. https://doi.org/10.1109/TAC.1968.1098999

Wu, Z., Rincon, D., & Christofides, P. D. (2020). Process structure-based recurrent neural network modeling for model predictive control of nonlinear processes. *Journal of Process Control*, *89*, 74–84. https://doi.org/10.1016/j.jprocont.2020.03.013

Yin, H., Seiler, P., & Arcak, M. (2021). Stability analysis using quadratic constraints for systems with neural network controllers. *IEEE Transactions on Automatic Control*, *67*(4), 1980–1987. https://doi.org/10.1109/TAC.2021.3069388

Appendices

Appendix 1. Definitions and transformation

A.1 Definitions

To keep the paper self contained we repeat the definitions of class $\mathcal{K}$, $\mathcal{K}_\infty$ and $\mathcal{KL}$ from Bonassi et al. (2022).

Definition A.1: A continuous function $\gamma : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ is called a class $\mathcal{K}$ function if $\gamma(s) > 0$ for all $s > 0$, it is strictly increasing and $\gamma(0) = 0$. The function γ is a class $\mathcal{K}_\infty$ function if it is class $\mathcal{K}$ function and $\gamma(s) \to \infty$ for $s \to \infty$.

Definition A.2: A continuous function $\gamma : \mathbb{R}_{>0} \times \mathbb{Z} \to \mathbb{R}_{>0}$ is called a class $\mathcal{KL}$ function if $\beta(s,k)$ is a class $\mathcal{K}$ function with respect to s for all k, it is strictly decreasing in k for all $s > 0$, and $\beta(s,k) \to 0$ as $k \to \infty$ for all $s > 0$.

A.2 Transformation

Let us rewrite the system matrices from (47) to show the interconnection with the new controller matrices

$$\left(\begin{array}{c|cc} \mathcal{Z}\mathcal{A}\mathcal{Y} & \mathcal{Z}\mathcal{B} & \mathcal{Z}\mathcal{B}_2 \\ \hline \mathcal{C}\mathcal{Y} & \mathcal{D} & \mathcal{D}_{12} \\ \mathbf{\Lambda}\mathcal{C}_2\mathcal{Y} & \mathbf{\Lambda}\mathcal{D}_{21} & 0 \end{array}\right) = \begin{pmatrix} \mathcal{Z} & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \mathbf{\Lambda} \end{pmatrix}$$

$$\begin{pmatrix} \mathcal{A} & \mathcal{B} & \mathcal{B}_2 \\ \mathcal{C} & \mathcal{D} & \mathcal{D}_{12} \\ \mathcal{C}_2 & \mathcal{D}_{21} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{Y} & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix}$$

$$= \begin{pmatrix} \mathcal{Z} & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \mathbf{\Lambda} \end{pmatrix} \left[\left(\begin{array}{cc|cc} A_{\text{lin}} & 0 & B_{\text{lin}} & 0 \\ 0 & 0 & 0 & 0 \\ \hline C_{\text{lin}} & 0 & D_{\text{lin}} & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \right.$$

$$
\begin{aligned}
&+\left(\begin{array}{cc|c}0 & B_{\text{lin},2} & 0\\ I & 0 & 0\\ \hline 0 & D_{\text{lin},2} & 0\\ 0 & 0 & I\end{array}\right)\left(\begin{array}{ccc}A & B & B_2\\ C & D & D_{12}\\ C_2 & D_{21} & 0\end{array}\right)\\
&\quad\times\left(\begin{array}{cc|cc}0 & I & 0 & 0\\ \binom{I}{0} & 0_{n_y\times n_x} & \binom{0}{I} & 0_{n_y\times n_w}\\ 0 & 0 & 0 & I\end{array}\right)\Bigg]\left(\begin{array}{ccc}\mathcal{Y} & 0 & 0\\ 0 & I & 0\\ 0 & 0 & I\end{array}\right)\\
&=\left(\begin{array}{cc|cc}A_{\text{lin}}Y & A_{\text{lin}} & B_{\text{lin}} & 0\\ 0 & XA_{\text{lin}} & XB_{\text{lin}} & 0\\ \hline C_{\text{lin}}Y & C_{\text{lin}} & D_{\text{lin}} & 0\\ 0 & 0 & 0 & 0\end{array}\right)+\left(\begin{array}{cc|c}0 & B_{\text{lin},2} & 0\\ I & 0 & 0\\ \hline 0 & D_{\text{lin},2} & 0\\ 0 & 0 & I\end{array}\right)\\
&\left(\begin{array}{ccc}K & L & L_2\\ M & N & N_{12}\\ \boldsymbol{\Lambda}M_2 & \boldsymbol{\Lambda}N_{21} & 0\end{array}\right)\\
&\quad\times\left(\begin{array}{cc|cc}I & 0 & 0 & 0\\ 0_{n_y\times n_x} & \binom{I}{0} & \binom{0}{I} & 0_{n_y\times n_w}\\ 0 & 0 & 0 & I\end{array}\right)\\
&=\left(\begin{array}{cc|c}\mathbf{A} & \mathbf{B} & \mathbf{B}_2\\ \hline \mathbf{C} & \mathbf{D} & \mathbf{D}_{12}\\ \boldsymbol{\Lambda}\mathbf{C}_2 & \boldsymbol{\Lambda}\mathbf{D}_{21} & 0\end{array}\right)
\end{aligned}
$$

The last line follows by plugging in the definition of the new controller matrices (41).

Appendix 2. Experiments

A.3 Coupled mass-spring-damper system

The mass-spring-damper system has four links where the first mass is attached to a wall, the scalar input represents a force at the first mass, and the measured output is the position of the fourth mass; an illustration is shown in Figure 8. The nonlinearity of the system results from the spring constant. This system was used in Revay et al. (2020, 2021).

Each mass has the distance and velocity as internal state $x_i = [d_i \;\; v_i]^\top$ for $i = 1, \ldots, 4$ and we linearise around $\bar{x} = [0\;\;0\;\;0\;\;0\;\;0\;\;0\;\;0\;\;0]^\top$, $\bar{u} = 0$ to get

$$
\begin{aligned}
&A_{d,\text{msd}}\\
&=\begin{pmatrix}
\overset{1}{-\frac{\eta(k_0+k_1)}{4m_0}} & \overset{\eta}{\frac{-\eta(c_0+c_1)+m_0}{m_0}} & \overset{0}{\frac{\eta k_1}{4m_0}} & \overset{0}{\frac{c_1\eta}{m_0}} & \overset{0}{0} & \overset{0}{0} & \overset{0}{0} & \overset{0}{0}\\
\overset{0}{\frac{\eta k_1}{4m_1}} & \overset{0}{\frac{c_1\eta}{m_1}} & \overset{1}{-\frac{\eta(k_1+k_2)}{4m_1}} & \overset{\eta}{\frac{-\eta(c_1+c_2)+m_1}{m_1}} & \overset{0}{\frac{\eta k_2}{4m_1}} & \overset{0}{\frac{c_2\eta}{m_1}} & \overset{0}{0} & \overset{0}{0}\\
\overset{0}{0} & \overset{0}{0} & \overset{0}{\frac{\eta k_2}{4m_2}} & \overset{0}{\frac{c_2\eta}{m_2}} & \overset{1}{-\frac{\eta(k_2+k_3)}{4m_2}} & \overset{\eta}{\frac{-\eta(c_2+c_3)+m_2}{m_2}} & \overset{0}{\frac{\eta k_3}{4m_2}} & \overset{0}{\frac{c_3\eta}{m_2}}\\
\overset{0}{0} & \overset{0}{0} & \overset{0}{0} & \overset{0}{0} & \overset{0}{\frac{\eta k_3}{4m_3}} & \overset{0}{\frac{c_3\eta}{m_3}} & \overset{1}{-\frac{\eta k_3}{4m_3}} & \overset{\eta}{-\frac{c_3\eta}{m_3}+1}
\end{pmatrix}\\
&B_{d,\text{msd}}=\begin{pmatrix}0\\ \frac{\eta}{m_0}\\ 0\\ 0\\ 0\\ 0\\ 0\end{pmatrix}\\
&C_{d,\text{msd}}=\begin{pmatrix}0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\end{pmatrix}\\
&D_{d,\text{msd}}=\begin{pmatrix}0\end{pmatrix}
\end{aligned}
\tag{A1}
$$

with $m = (1/4\;\; 1/3\;\; 5/12\;\; 1/2)^\top$, $c = (1/4\;\; 1/3\;\; 5/12\;\; 1/2)^\top$ and $k = (1\;\; 5/6\;\; 2/3\;\; 1/2)^\top$

A.4 Ship dataset

The nonlinear ship system consists of an environmental model for the wind (Isherwood, 1972), wind-induces waves (Hasselmann et al., 1973) and a nonlinear, four degrees-of-freedom model of the ship motion (Fossen, 2011).

The system identification task is to learn a model that maps environmental measurements and actuator inputs to changes in system states, e.g. the ship's velocities. The environment model is unknown; however, we assume it has an approximation model of the ship dynamics. The book by Fossen (2011) provides nonlinear differential equations that map forces to ship velocities. A generic description of such a nonlinear differential equation is given

$$
\begin{aligned}
\dot{\eta} &= J(\eta)\nu\\
\dot{\boldsymbol{\nu}} &= M_{\text{mass}}^{-1}(\text{actuator} + \text{environment} - D(\nu)\nu\\
&\quad - C_{\text{RB}}(\nu)\nu - g(\eta)).
\end{aligned}
\tag{A2}
$$

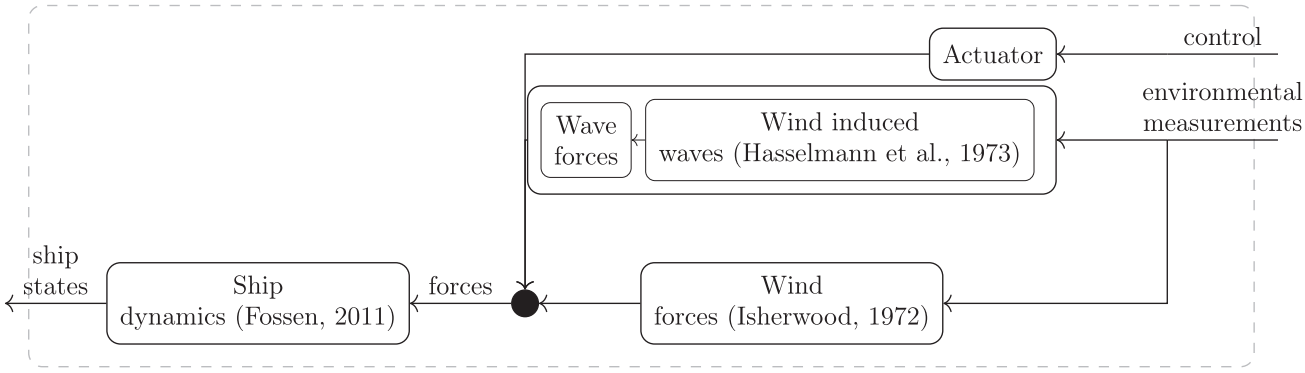

Figure A1. Nonlinear ship motion model, which is an interconnection between an environmental model of the wind, input actuators, and a nonlinear dynamical four degrees-of-freedom ship model.

Where $\eta = [x_{pos}, y_{pos}, \phi, \psi]^{\mathrm{T}}$ is the position and angles and $\boldsymbol{\nu} = [s, v, p, r]^{\mathrm{T}}$ are the velocities (surge, sway) and angular velocities (roll, yaw) of the ship. The inputs to the ship dynamics are the forces from the environment environment and the actuator (see Figure A1), M_{mass} is the mass matrix that contains added masses resulting from the hydrodynamics. The rotation matrix $J(\eta)$ transforms the body coordinates into positions in the two-dimensional plane and the angles. The damping and Coriolis term is denoted by $D(\nu)$ and $C_{RB}(\nu)$, respectively, where the subscript RB is used to indicate the rigid-body forces. The restoring forces of the buoyancy are denoted by $g(\eta)$. The nonlinear differential equation in (A2) depends on the parameters from Table A1.

We linearise (A2) to get a continuous linear model described by

$$\dot{x} = A_{\mathrm{ship}} x + B_{\mathrm{ship}} u \tag{A3}$$

where $x = [u, v, p, r]^{\top}$ are the ship states and $F_r]$ are the external forces. The matrices A_{ship} and B_{ship} describe the linear behaviour of the ship states. We discretise (A3) with a zero-order-hold method to get a linear approximation model $\mathcal{H}_{\mathrm{lin}}$ that we can use in our hybrid model architecture

$$\mathcal{H}_{\mathrm{lin,ship}} := \begin{cases} x^{k+1} = A_{d,\ \mathrm{ship}}\, x^k + B_{d,\ \mathrm{ship}} u^k \\ y^k = C_{d,\mathrm{ship}} x^k + D_{d,\mathrm{ship}} u^k \end{cases} \tag{A4}$$

with the matrices

$$A_{d,\mathrm{ship}} = \begin{pmatrix} 0.9501 & 0 & 0 & 0 \\ 0 & 0.9725 & 0.0391 & -0.2953 \\ 0 & 0.0127 & 0.8678 & -0.4917 \\ 0 & -0.0058 & 0.0007 & 0.7946 \end{pmatrix}$$

$$B_{d,\mathrm{ship}} = 10^{-5} \begin{pmatrix} 0.2544 & 0 & 0 & 0 \\ 0 & 0.0435 & -0.0068 & -0.0003 \\ 0 & -0.0009 & 0.0229 & -0.0003 \\ 0 & 0.0008 & -0.0001 & 0.0011 \end{pmatrix}$$

$$C_{d,\mathrm{ship}} = I$$

$$D_{d,ship} = 0 \tag{A5}$$

The ship is linearised around a constant forward movement $\bar{x} = [5, 0, 0, 0]^{\top}$ that is achieved by the constant input $\bar{u} = [25X_{u|u|}, 0, 0, 0]^{\top}$

Appendix 3. Architecture

Remark A.1: From the RNN, we get the size of the uncertainty channel $n_w = n_z$ and the size of the internal state $n_{x_{\mathrm{rnn}}}$ as hyperparameters. We require the state of the linear system to be equal to the state of the RNN. From classical controller synthesis problems, we know that increasing the size of the controller does not improve the controller's performance (Scherer & Weiland, 2000). This is not necessarily true for our residual model since our objective is high prediction accuracy and dissipativity. To increase the size of the RNN model, we can extend the state of the linear model $\mathcal{H}_{\mathrm{lin}}$ by adjoining zeros to the linear matrices A_{lin}, B_{lin}, C_{lin}, and D_{lin}. We then get

$$\tilde{\mathcal{H}}_{\mathrm{lin}} = \left\{ \begin{pmatrix} x^{k+1} \\ \hat{y}^k \\ \binom{y_1}{y_2}^k \end{pmatrix} = \left(\begin{array}{cc|c} \begin{pmatrix} A_{\mathrm{lin}} & 0_{n_x \times n_k} \\ 0_{n_x \times n_k} & 0_{n_k \times n_k} \end{pmatrix} & \begin{pmatrix} B_{\mathrm{lin}} \\ 0_{n_k \times n_d} \end{pmatrix} & \left(\begin{pmatrix} \tilde{B}_{\mathrm{lin},2} & 0_{n_x \times n_k} \\ 0_{n_x \times n_k} & 0_{n_k \times n_k} \end{pmatrix} \begin{pmatrix} \tilde{B}_{\mathrm{lin},3} \\ 0_{n_k \times n_{N_f}} \end{pmatrix} 0 \right) \\ \begin{pmatrix} C_{\mathrm{lin}} & 0_{n_e \times n_k} \end{pmatrix} & D_{\mathrm{lin}} & \begin{pmatrix} \tilde{D}_{\mathrm{lin},2} & 0_{n_e \times n_k} \end{pmatrix} 0 \ \tilde{D}_{\mathrm{lin},3} \\ \hline I & 0 & 0 \\ 0 & I & 0 \end{array} \right) \begin{pmatrix} x^k \\ u^k \\ u^k \end{pmatrix} \right. \tag{A6}$$

Where n_k is the number of additional states, this extension does not change the behaviour of the linear system but leads to larger parameter matrices. We will compare different sizes of the RNN state and uncertainty channel in Section 7.

Table A1. Ship parameters from Perez et al. (2006).

Parameter	Value	Description
ρ_{water}	1025.0	seawater density
g	9.81	gravity constant
d	355.88	displacement
m	$365.79 \cdot 10^3$	Mass
I_{zz}	$3.3818 \cdot 10^7$	Yaw Inertia
I_{xx}	$3.4263 \cdot 10^6$	Roll Inertia
g_m	1.0	Transverse Metacenter Height
L_{CG}	20.41	Longitudinal CG
V_{CG}	3.36	Vertical CG above baseline
x_G	−3.38	coordinate of CG from the body fixed frame adopted for the PMM test
z_G	−1.06	coordinate of CG from the body fixed frame adopted for the PMM test
$X_{\dot{u}}$	−17,400.0	Data for surge equation
$X_{u\|u\|}$	−1960.0	
X_{vr}	$0.33 * \text{m}$	
$Y_{\dot{v}}$	$-1.9022 \cdot 10^6$	Hdrodynamic coefficients in sway equation
$Y_{\dot{p}}$	$-0.296 \cdot 10^6$	
$Y_{\dot{r}}$	$-1.4 \cdot 10^6$	
$Y_{\|u\|v}$	−11,800	
Y_{ur}	131,000	
$Y_{v\|v\|}$	−3700	
$Y_{r\|r\|}$	0	
$Y_{v\|r\|}$	−794,000	
$Y_{r\|v\|}$	−182,000	
$Y_{\phi\|uv\|}$	10,800	
$Y_{\phi\|ur\|}$	251,000	
$Y_{\phi\|uu\|}$	−74	
$K_{\dot{v}}$	296,000	Hdrodynamic coefficients in roll equation
$K_{\dot{p}}$	−674,000	
$K_{\dot{r}}$	0	
$K_{\|u\|v}$	9260	
K_{ur}	−102,000	
$K_{v\|v\|}$	29,300	
$K_{r\|r\|}$	0	
$K_{v\|r\|}$	621,000	
$K_{r\|v\|}$	142,000	
$K_{\phi\|uv\|}$	−8400	
$K_{\phi\|ur\|}$	−196,000	
$K_{\phi uu}$	−1180	
$K_{\|u\|p}$	−15,500	
$K_{p\|p\|}$	−416,000	
K_p	−500,000	
K_ϕ	0	
$K_{\phi\phi\phi}$	$-0.325\rho_{\text{water}}gd$	
$N_{\dot{v}}$	538,000	Hdrodynamic coefficients in yaw equation
$N_{\dot{p}}$	0	
$N_{\dot{r}}$	$-4.3928 \cdot 10^7$	
$N_{\|u\|v}$	−92,000	
$N_{\|u\|r}$	−4,710,000	
$N_{v\|v\|}$	0	
$N_{r\|r\|}$	−202,000,000	
$N_{v\|r\|}$	0	
$N_{r\|v\|}$	−15,600,000	
$N_{\phi\|uv\|}$	−214,000	
$N_{\phi\|ar\|}$	−4,980,000	
$N_{\phi\|au\|}$	−8000	