Deep-Learning-Based Identification of LPV Models for Nonlinear Systems

Chris Verhoek, Gerben I. Beintema, Sofie Haesaert, Maarten Schoukens, and Roland Tóth

Abstract—The Linear Parameter-Varying (LPV) framework provides a modeling and control design toolchain to address nonlinear (NL) system behavior via linear surrogate models. Despite major research effort on LPV data-driven modeling, a key shortcoming of the current identification theory is that often the scheduling variable is assumed to be a given measured signal in the data set. In case of identifying an LPV model of a NL system, the selection of the scheduling map, which describes the relation to the measurable scheduling signal, is put on the users’ shoulder, with only limited supporting tools available. This choice however greatly affects the usability and complexity of the resulting LPV model. This paper presents a deep-learning-based approach to provide joint estimation of a scheduling map and an LPV state-space model of a NL system from input-output data, and has consistency guarantees under general innovation-type noise conditions. Its efficiency is demonstrated on a realistic identification problem.

Index Terms—System Identification, Deep Learning, Linear Parameter-Varying Systems, Nonlinear Systems.

I. INTRODUCTION

With the continuous push for increasing performance and energy efficiency of systems in engineering, like high-tech mechatronic devices, air and spacecrafts, power systems, etc., nonlinear (NL) and time-varying (TV) effects have become dominant in the dynamic behavior of these systems. To cope with these effects in engineering, an efficient modeling and control design toolchain is required that is capable of building on the already existing, vast experience with Linear Time-Invariant (LTI) design tools. The Linear Parameter-Varying (LPV) framework has been established for this purpose, providing accurate, but low-complexity surrogate representations of NL and TV systems [1]. In LPV systems, the signal relations are considered to be linear, just as in the LTI case, but the parameters defining these relations are assumed to be functions of a measurable, time-varying signal, the so-called scheduling variable \( p : \mathbb{Z} \rightarrow \mathbb{P} \subseteq \mathbb{R}^{n_p} \), which captures the original NL/TV effects in the system [2]. The LPV system class has shown to be capable of representing a wide variety of physical processes, but its major advantage is its well-worked-out and industrially-reputed controller design framework, which allows to guarantee closed-loop stability and performance of the original physical system.

In order to support the use of LPV control design methods, a serious research effort has been spent on deriving a wide range of NL model conversion approaches to transform existing first-principle models of the system to low-complexity LPV models, see e.g. [2]–[5]. To formulate the LPV signal relations, these methods should provide (i) construction of a scheduling map \( \phi : \mathbb{R}^{n_p} \rightarrow \mathbb{P} \) that describes how external or internal signals of the system are mapped to a scheduling variable \( p \), and (ii) derive \( p \)-dependent coefficient functions, like mappings \( A, B, C, D \) in case of an LPV State-Space (SS) model [2]. In general, low complexity of the LPV signal relations is preferred to simplify the follow-up utilization of the resulting model. Low complexity is often interpreted in terms of, e.g., affine scheduling dependence of the coefficient functions and a minimal scheduling dimension \( n_p \). Furthermore, deploying LPV controllers that are designed based on such models require real-time computation of \( p \). Hence, scheduling variables that can be directly computed from measurable signals coming from the system, without the need of NL observers, are highly preferable. In overall, it has been realized that selection of \( \phi \) has paramount importance in the conversion process, as it governs the expected performance and utilization potential of the overall LPV toolchain [3].

However, (LPV) models of the considered engineering systems that have the desired level of accuracy are often not available in practice. Moreover, it is usually difficult to decide which specific aspects of first-principles models are essential for representing the dynamic behavior of the system. Hence, a wide range of data-driven modeling methods have been introduced that aim to estimate LPV models directly from measured input-output (IO) data of the system. Methods ranging from continuous to discrete time, using various model structures, such as, SS, IO, series expansion and linear fractional representations, under a wide range of noise scenarios, and using only local operating-point-based measurements, global measurements or both, have been developed and successfully applied in practice, see the overviews in [2], [6], [7]. Especially prediction error minimization (PEM) methods [8], [9] and sub-space techniques [10], [11] have proven to be quite successful, while sparsity-based model structure selection tools [12] and learning methods, such as, support vector machines [13]–[15], Gaussian processes [16] and Artificial Neural Network (ANN) methods [17]–[20], have also been introduced to provide flexible estimators of the required coefficient functions dependency. However, these methods almost exclusively rely on a predetermined choice of \( p \), assuming that its variation is part of the measured data set. This puts the selection of the scheduling map \( \phi \) on the users’ shoulder, although it is such an important part of the overall modeling process. So far, a data-based choice for the scheduling map is only...
developed for a limited set of LPV-IO PEM methods by using sparse regression-based methods. These, however, still rely on a predefined set of signals and relations on which this choice is based. This difficulty comes from the overall complexity of the joint estimation problem of $\phi$ together with the $p$-dependent coefficient functions, constituting the LPV model. This joint estimation problem is inherently a NL identification problem, requiring flexible function estimators, without restricting a priori choices and constraints.

To overcome this challenging problem, we present a PEM-based deep-learning approach to provide joint estimation of a low-complexity scheduling map and an affinely dependent LPV-SS model, to which we refer to as the joint LPV identification problem. Our approach allows to accurately identify a NL system directly from IO data, without any structural knowledge or a priori choices required from the user. The presented method builds on recent results in NL system identification in terms of the Sub-Space Encoder Network (SUBNET) method [21]–[23]. SUBNET uses a novel state-encoder and a batch-wise formulation of the prediction cost, which results in an efficient NL identification approach. Our contribution in this paper is the modification of SUBNET to the joint LPV identification problem, under innovation-type noise structures, resulting in an efficient estimator for NL systems. The provided method is able to estimate LPV models for which the internal state of the model itself is also used to determine the scheduling, i.e., the model is capable of self-scheduling. Furthermore, we introduce an alternative formulation to estimate LPV models for which the scheduling is directly computed from past IO data. This alternative formulation yields a practically implementable scheduling map for controllers that are designed for the resulting model. The approach has consistency guarantees under general innovation-type noise conditions and its efficiency is demonstrated on the identification problem of a control moment gyroscope. Note that, compared to other neural network based LPV identification methods (e.g., [17]–[20]), this is by our knowledge the first approach where (i) $p$ is not part of the data set, (ii) the corresponding scheduling map $\phi$ is directly learned from past IO data, and (iii) a noise model is estimated together with the process model.

Section II introduces the joint LPV identification problem for NL systems, while the proposed LPV sub-space encoder approach is discussed in Section III. We demonstrate the strength of the approach on a simulation study in Section IV and the conclusions are given in Section V.

II. PROBLEM STATEMENT

Consider the discrete-time (DT) data-generating system given by the NL SS representation

$$
\Sigma : \begin{cases}
  x_{k+1} = f(x_k, u_k, w_k), \\
  y_k = h(x_k, u_k) + w_k,
\end{cases}
$$

where $x_k \in \mathbb{X} \subseteq \mathbb{R}^{n_x}$ is the state, $u_k \in \mathbb{U} \subseteq \mathbb{R}^{n_u}$ is the input and $y_k \in \mathbb{Y} \subseteq \mathbb{R}^{n_y}$ is the observed output at time moment $k \in \mathbb{Z}$, while $w_k \in \mathbb{W} \subseteq \mathbb{R}^{n_w}$ is the realization of an i.i.d. white noise process with finite variance $\Gamma_w \in \mathbb{R}^{n_w \times n_w}$. Moreover, $\mathbb{X}, \mathbb{U}, \mathbb{Y}$ and $\mathbb{W}$ are considered to be open sets containing the origin, and $f : \mathbb{X} \times \mathbb{U} \times \mathbb{W} \rightarrow \mathbb{X}$ and $h : \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}^{n_y}$ are bounded functions, with $h(\mathbb{X}, \mathbb{U}) \oplus \mathbb{W} \subseteq \mathbb{Y}$, where $\oplus$ is the Minkowski addition on sets. The noise structure in (1) corresponds to an innovation-type noise process [10], [11], [24], [25], which has proven to be rather general for various system classes and has been a cornerstone of the sub-space identification machinery widely used in practice [24]. Finally, we assume that the one-step-ahead predictor of (1) is stable.

Based on (1), we have $w_k = y_k - h(x_k, u_k)$, which allows to write the state-equation in the filter form, i.e.,

$$
x_{k+1} = f(x_k, u_k, y_k - h(x_k, u_k)),
$$

expressing the state evolution without $w_k$. Based on an observed length-$N$ data sequence $D_N = \{(y_k, u_k)\}_{k=1}^N$ generated by (1), our objective is to identify an LPV model of (1) in the form of

$$
\Sigma_{LPV} : \begin{cases}
  x_{k+1} = A(p_k)x_k + B(p_k)u_k + K(p_k)w_k, \\
  y_k = C(p_k)x_k + D(p_k)u_k + w_k,
\end{cases} \tag{3}
$$

where $p_k \in \mathbb{P} \subseteq \mathbb{R}^{n_p}$ is the scheduling variable, which varies in the scheduling space $\mathbb{P}$, while $A, B, C, D, K$ are bounded matrix functions of $p_k$ with appropriate dimensions.

The main advantage of modeling (1) in the form of (3) is that it expresses the underlying NL system behavior in a linear form. This allows the use of powerful convex control and observer synthesis methods with stability and performance guarantees [2], [26]. This representation capability is achieved via the $p_k$ induced variation of $A, \ldots, K$. For this purpose, the scheduling variable $p_k$ is considered to be a measurable signal, which is assumed to be a free, independent variable during analysis or controller synthesis for (3). However, in terms of an LPV embedding of the NL system $\Sigma$, $p_k$ has an underlying connection to the signals in the system with a so-called scheduling map $\phi : \mathbb{X} \times \mathbb{U} \times \mathbb{Y} \rightarrow \mathbb{P}$ in terms of (2), such that the behaviors, i.e., solution sets, of $\Sigma$ and $\Sigma_{LPV}$ are equivalent.

In such an embedding process, a tradeoff in dividing the complexity of $f$ and $h$ in (1) over the dependency structure of $A, \ldots, K$ on $p_k$, and the scheduling map $\phi$ is required. To ensure applicability of a wide range of LPV control synthesis approaches on (3), it is often desired to restrict the matrix functions $A, \ldots, K$ to have affine dependency on $p_k = [p_{k,1} \ldots p_{k,n_p}]^\top$, which corresponds to

$$
A(p_k) = A_0 + \sum_{i=1}^{n_p} A_i p_{k,i}. \tag{4}
$$

Hence, it is a highly important problem, even in case when (1) is fully known, to determine a low dimensional scheduling map $\phi$, which allows to represent (1) in the form of (3) with affine dependency of $A, \ldots, K$. As explained in Section I, in case of an unknown system (1) with no structural information on $f$ and $h$, there is currently no identification method that can simultaneously determine $A, \ldots, K$ and $\phi$, by reaching an optimal complexity-tradeoff, while also dealing with the general innovation-type noise structure.

To formulate the exact modeling problem of our joint LPV identification problem, consider the model structure, i.e., the
parametrized predictor form of (3), see [10]:
\[ \hat{x}_{k+1} = A_{\theta}(\hat{p}_k)\hat{x}_k + B_{\theta}(\hat{p}_k)u_k + K_{\theta}(\hat{p}_k)\hat{e}_k, \]
\[ \hat{y}_k = C_{\theta}(\hat{p}_k)\hat{x}_k + \ldots \] initial state \( \hat{x}_t|t \) of each subsection is unknown. Instead of introducing a free parametrization of \( \hat{x}_t|t \), which would minimize the prediction error (6) in terms of (5).

A. Batch prediction loss
Consider the scheduling mapping \( \phi_\eta \) as a multi-layer ANN, parametrized in \( \eta \in \Omega \), where each hidden layer is composed from \( n_\sigma \) activation functions \( \sigma : \mathbb{R} \rightarrow \mathbb{R} \) (e.g., sigmoid, ReLU, etc., see [28]) in the form of \( z_{i,j} = \sigma(\sum_{q=1}^{q} \eta_{w,i,j}z_{i-1,j} + \eta_{h,i,j}) \), where \( z_i = \text{col}(z_1, \ldots, z_{n_r}) \) is the latent variable representing the output of layer \( 1 \leq i \leq q \). For \( \phi_\eta \) with \( q \) hidden-layers and linear input and output layers, this means \( \phi_\eta(\hat{x}_t, \hat{u}_k, \hat{y}_k) = \theta_{w,q+1}z_q(k) + \theta_{b,q+1} \) and \( z_q(k) = \text{col}(\hat{x}_k, \hat{u}_k, \hat{y}_k) \). With such a form of \( \phi_\eta \) and linear parametrization of (3) in \( \theta \) it is not very difficult to formulate the minimization problem of (6) in popular software packages, such as PyTorch, and provide deep-learning-based joint estimates.

However, the first obstacle to successfully solve the identification problem in this way is that the computational cost of (6) scales at least linearly with \( N \). Furthermore, despite the advanced automatic gradient calculation tools and stochastic gradient descent (SGD) methods [28], the forward iterated prediction cost function (6) is sensitive to local minima, and gradient-based optimization methods commonly display unstable behaviors [29]. To avoid this problem, an important observation is that (6) can be well-represented as a sum of prediction subsections of truncation length \( T \):

\[ V^\text{pred}_{D_N}(\theta, \eta) = \frac{1}{N-T+1} \sum_{k=0}^{N-T} \left( \sum_{i=0}^{T-1} \sum_{j=i}^{T-1} \| \hat{y}_{t+k|i} - y_{t+k} \|^2 \right), \]
\[ \hat{x}_{t+k+1|i} = A_{\theta}(\hat{p}_{t+k|i})\hat{x}_{t+k|i} + B_{\theta}(\hat{p}_{t+k|i})u_{t+k} + K_{\theta}(\hat{p}_{t+k|i})\hat{e}_{t+k|i}, \]
\[ \hat{y}_{t+k|i} = C_{\theta}(\hat{p}_{t+k|i})\hat{x}_{t+k|i} + D_{\theta}(\hat{p}_{t+k|i})u_{t+k}, \]
\[ \hat{e}_{t+k|i} = y_{t+k} - \hat{y}_{t+k|i}, \]
\[ \hat{p}_{t+k|i} = \phi_\eta(\hat{x}_{t+k|i}, u_{t+k}, y_{t+k}), \]

where the notation \( | \) is introduced to distinguish between subsections as current index [start index]. If the truncation length is set to \( T = N \), then (6) is recovered. A significant advantage of (7) is that it increases the loss function smoothness [29], making gradient-based optimization methods more stable and consistent. Moreover, the computational cost of (7) can be reduced from \( O(N) \) to \( O(T) \) by using modern hardware (e.g., GPU architectures), as the sum over \( t \) in (7a) can be computed in parallel.

The computational cost of (7) can be further decreased by not summing over all the subsections of the complete data set \( D_N \) for each optimization step, but only over a subset of subsections. This results in a batch-ℓ2-loss formulation:

\[ V^\text{batch}_{D_N}(\theta, \eta) = \frac{1}{T \cdot N_{\text{batch}}} \sum_{t \in I_1} \sum_{k=0}^{T-1} \| \hat{y}_{t+k|t} - y_{t+k} \|^2 \]

where \( I_1 \subseteq I_1^{N-T+1} = \{1, 2, \ldots, N - T + 1\} \) contains a selection of possible batches in (7) and \( N_{\text{batch}} = |I_1| \). (8) allows the use of powerful batch SGD algorithms such as the Adam optimizer [30]. Despite the computational advantage, an important shortcoming of minimization of (8) is that the initial state \( \hat{x}_{t|t} \) of each subsection is unknown. Instead of introducing a free parametrization of \( \hat{x}_{t|t} \), which would...
quickly result in an explosion of the computational costs and the model variance, we now propose an encoder-based state estimator that uses the concept of state reconstructability, which is an LPV variant of the core idea of SUBNET.

B. LPV sub-space encoder

To derive a state-estimator for the batches in (8), note that the future evolution of (3) is

\[
y_k = C(p_k)x_k + D(p_k)u_k + w_k,
\]

\[
y_{k+n} = C(p_{k+n}) \prod_{t=0}^{n-1} \hat{A}(p_{k+t})x_k + D(p_{k+n})u_{k+n} + w_{k+n}
\]

\[
+ C(p_{k+n}) \prod_{t=0}^{n-1} \hat{A}(p_{k+t+\tau})\hat{B}(p_{k+\tau})u_{k+\tau}
\]

\[
+ C(p_{k+n}) \prod_{t=0}^{n-1} \hat{A}(p_{k+t+\tau})K(p_{k+\tau})y_{k+\tau},
\]

where \( \hat{A} = A - KC \) and \( \hat{B} = B - KD \). By introducing \( y_{k+n}^k = [u_{k}^T \ldots u_{k+n}^T] \) and \( p_{k+n}^k \) similarly defined, (9) can be written more compactly as

\[
(y_k \circ p)_k y_{k+n}^k = (\Omega_n \circ p)_k x_k + (\mathcal{L}_n \circ p)_k u_{k+n} + w_{k+n}
\]

(10)

where \( \Omega_n \) is the n-step observability matrix of (3), \( \mathcal{L}_n \) is the input transition matrix with lower triangular Toeplitz structure, \( \mathcal{E}_n \) is the output filtering matrix (can be seen as part of an extended observability matrix) corresponding to the innovation process and \( \circ \) expresses the dynamic polynomial dependence of these matrix functions on \( p_{k+n}^k \). Under a given scheduling sequence \( p_{k+n}^k \) (i.e., known scheduling map \( \phi \)) and structural observability of (3) (i.e., invertibility of \( \Omega_n \)),

\[
x_k = (\Omega_n \circ p)_k (\mathcal{E}_n \circ p)_k y_{k+n}^k - (\Omega_n \circ p)_k u_{k+n}^k + w_k - w_{k+n}
\]

(11)

has been exploited in LPV sub-space identification to provide estimates of \( A, \ldots, K \) [10], [11]. However, in case of the considered joint estimation problem, \( \phi \) is unknown and it is potentially dependent on the state \( x_k \), requiring to consider (10) as an NL observability problem in \( x_k \) [31]. Based on the back-substitution of \( \phi \), such a problem can be solved, giving the n-step observability map \( x_k = \Lambda_n(u_{k+n}^k, y_{k+n}^k, w_k^k) \) under \( n \geq n_x \). Similarly, an n-step reconstructability map \( x_k = \Psi_n(u_{k-n}^k, y_{k-n}^k, w_{k-n}^k) \) can be established, by considering the reconstruction of \( x_k \) from past samples of \( u_k, y_k, w_k \). For the exact construction and the required conditions, see [23]. Note that in practice, the noise sequence \( w_{k-n}^k \) is not directly available to compute this recovery based on \( \Psi_n \). We can however exploit the i.i.d. white noise property of \( w_k \) to arrive at an efficient estimator \( \hat{x}_k \) of \( x_k \) given by

\[
\hat{x}_k = E_w[x_k \mid u_{k-n}^k, y_{k-n}^k] = \Psi_n(u_{k-n}^k, y_{k-n}^k).
\]

(12)

Considering (5), we can similarly use \( \Psi_n \) to provide an estimate of \( \hat{x}_{k+1} \) for initialization of each batch computation in (8) from past IO data. However, the exact calculation of this estimator for a given ANN parametrization of the scheduling map \( \phi_{\eta} \) and \( A_0, \ldots, K_0 \) is practically infeasible. This is due to the required analytic inversion for \( \Psi_n \), and the computation of the conditional expectation of \( \Psi_n \) under an unknown \( \Gamma_{\phi} \). Hence, similar to [23], we aim to approximate \( \Psi_n \) by introducing a NL sub-space encoder function:

\[
\hat{x}_{k+1} \triangleq \psi(x_{k-n}, y_{k-n}^t).
\]

(13)

Here, \( n \) corresponds to the number of past inputs and outputs, i.e., lag window, considered to estimate the initial state \( \hat{x}_{k+1} \), while \( \xi \in \Xi \subseteq \mathbb{R}^{n_x} \) is the collection of the parameters associated with \( \psi \) in terms of a corresponding ANN with multiple hidden layers. In order to provide an estimator for the initial state of the considered model structure (5), the encoder function \( \psi \) is co-estimated with the scheduling map \( \phi_{\eta} \) and the matrices in \( A_0, \ldots, K_0 \) by including the parameters \( \xi \) in the loss function (8).

C. Network structure of the estimator

Fig. 1 summarizes the ANN structure of our proposed method, which allows to minimize the batch loss (8) by exploiting the derived sub-space encoder (13) and the forward propagation model structure in (7b)-(7c). This structure, named as LPV-SUBNET, cleverly embeds the structural LPV information in SUBNET. The LPV-SUBNET consists of a (deep) forward ANN-based encoder \( \psi \) that takes as input the last \( n \) measurements of \( u \) and \( y \) from the data-generating system. From this data, the encoder estimates the model state at time-step \( t \), which in turn is forward propagated by the \( \theta \)-parametrized LPV model equations in terms of (7b)-(7c) and the innovation (7d). The required scheduling sequence is determined by a (deep) forward ANN scheduling map \( \phi_{\eta} \), called p-net, that uses the previously calculated \( \hat{x}_{t+1|t} \) to compute the scheduling for the next state update. This enforces that the LPV-SS model state can be used to determine the scheduling, which is often called self-scheduling in the LPV literature. Note that the structure in Fig. 1 is formulated under the general innovation-type noise model. In case the expected noise structure in the system is OE, i.e., \( K(p_k) \equiv 0 \) in the LPV embedding of the original system, the network structure can be simplified by dropping the parts highlighted in blue. Furthermore, we only depicted the first two steps in time in Fig. 1 for the batch calculation, while in fact \( T \) steps are considered in the algorithm.

Alternatively, the encoder \( \psi \) can be used to estimate the possibly required \( \hat{x}_k \) in each time-step, separating the scheduling map calculation from the forward propagation of the model. This formulation considers the scheduling as an external signal determined by a filter operation directly from the data-generating system, which is in line with the
intended use of the model for analysis and control purposes. The corresponding LPV-SUBNET structure with external scheduling estimation is depicted in Fig. 2 for the first two calculation steps in a batch.

**D. Parameter estimation and properties**

The LPV-SUBNET approach can be easily implemented in open-source software packages that support the training of neural networks, e.g., PyTorch. In particular, the Adam [30] optimizer with default parameters provides an efficient training method for the considered cost function and network structure. The original SUBNET method corresponds to a gradient-based NL SS model estimation method under a PEM-like criterion. It has been shown to provide consistency guarantees with the assumption of persistently exciting data and convergence of the optimization to the global minimum of (8), see [23], [32] for details. Moreover, as LPV-SUBNET is based on a rather specialized structure of SUBNET, the consistency properties of the SUBNET method are inherited. Note that while the approach provides a rather automated estimation of LPV models, certain hyper-parameters of the model structure, such as the state-order $n_x$, the lag-window $n$ and the scheduling dimension $n_p$ are still needed to be either selected by the user or optimized by popular hyper-parameter optimization methods (e.g., Bayesian optimization based approaches). The dependency structure of $A, \ldots, K$ (e.g., affine, polynomial, etc.) is mainly a user-based design choice that depends on the further utilization of the model, e.g., analysis, optimal LPV controller synthesis, etc.

**IV. SIMULATION RESULTS**

In this section, we study the estimation performance of our LPV-SUBNET method on the identification problem of a simulated control moment gyroscope system and compare it with the current state-of-the-art LPV identification methods that use an a priori given scheduling map. The data sets and code is available at https://tinyurl.com/lpvsunet.

**A. System description**

We consider a high-fidelity NL simulator of the 3 degrees-of-freedom Control Moment Gyroscope (CMG) setup, depicted in Fig. 3, for LPV identification. This system consists of a flywheel (with angular position $q_3$), mounted inside an inner blue gimbal (with angular position $q_2$), which in turn is mounted inside an outer red gimbal (with angular position $q_3$). The entire structure is supported by a rectangular silver frame (with angular position $q_4$) that can rotate around its vertical axis of symmetry. The gimbals can rotate freely. The setup is equipped with four DC motors and encoders, actuating and measuring the position of the flywheel and the gimbals. In this study, we consider the high-fidelity simulation model of the CMG as the data-generating system, which is a highly-complex and NL system due to the involved rotational dynamics. We consider the following scenario:

- $q_3 = 0$ is locked, $q_2$ is actuated with motor current $i_2$ as an input, while we measure $\dot{q}_4$ as an output.
- $\dot{q}_4$ is controlled independently with a random multi-level-reference signal with amplitude in $[30, 50]$ rad/s and a dwell-time between 4 and 8 seconds.

Simulation of the CMG is done in continuous time (CT) with fixed-step RK4, sub-sampled to 100 Hz to obtain our data sets. By analyzing the first-principles based dynamics (see, e.g., [33]), it can be noted that, due to the independent control of the flywheel and the locked status of $q_3$, $\dot{q}_3$ can be regarded as an external input to a model describing the motion of $q_2$ and $q_4$. Finally, we want to emphasize here that the LPV identification methods with which we compare our learning approach, require a scheduling map definition. Analyzing a possible global LPV embedding of the DT CMG dynamics yields a scheduling map that is defined as

$$ p := \phi(x, u) = \left[ q_1 \sin(q_2) \cos(q_2) \right]^\top. \tag{14} $$

We will now compare state-of-the-art LPV identification methods under the choice of (14) on the CMG with our proposed LPV-SUBNET method, which is capable of the joint estimation of $\phi$ and an affine LPV-SS model.

**B. Experiment design**

We excite the CMG with the current signal:

$$ i_{2,k} := u_k = \frac{1}{2} \sin(\omega T_k) + v_k $$ \tag{15}

where $v$ is white with $v_k \sim \mathcal{N}(0, \sigma_v^2)$ and $\sigma_v = \frac{1}{3}$ together with $\omega \sim \mathcal{U}(1, 2)$ per data set realization. This gives a sinusoidal carrier signal with a frequency in $[1, 2]$ Hz and a superimposed white noise excitation with a total magnitude of $1.5$ and a confidence level of $99.7\%$. Moreover, we consider an ODE noise structure. The white noise signal $\omega$ is such that $\omega_k \sim \mathcal{N}(0, \sigma_\omega^2)$ with variance $\sigma_\omega^2 = 2.2 \cdot 10^{-5}$, which corresponds to a signal to noise ratio (SNR) of 35 dB. In order to capture the variation along a wide operating range of the CMG, the estimation data set is of size $N_d = 10^4$, while the validation data set is of size $N_s = 3 \cdot 10^4$. The validation set is uncorrelated to the estimation set. The resulting data sets are depicted in Fig. 4. For the LPV-SUBNET method, we consider the measured $i_2$ and $\dot{q}_1$ as...
the inputs and $\dot{q}_4$ as the output in terms of an available data set, while for regular LPV identification methods we use only $i_2$ as input, $\dot{q}_4$ as output and (14) as scheduling signal, providing significant structural information.

C. Estimation of LPV models

We compare our approach to the current state-of-the-art global LPV system identification methods. Particularly, LPV-OE [8], [9] and LPV PEM-SS [10], [34] identification. Only the considered hyper-parameters and initialization methods are discussed, while the details on these approaches are omitted. For the execution of the identification procedures, we use the MATLAB toolbox LPVCore [35].

1) LPV-OE identification: This approach aims to estimate the system in the form of an LPV-IO model with OE noise structure (i.e., $y_k = \hat{y}_k + \epsilon_k$):

$$\hat{y}_k + \sum_{i=1}^{n_a} a_i(p_k-i)\hat{y}_{k-i} = \sum_{j=0}^{n_b} b_j(p_k-j)u_{k-j},$$

(16)

under the cost function (6). In (16), $\{a_i\}_{i=1}^{n_a}$ and $\{b_j\}_{j=0}^{n_b}$ correspond to coefficient functions that have shifted affine dependency on $p_k$ [36]. Note that such a dependency structure ensures a direct minimal SS realization of the estimated models in the form of (3) with affine dependency and $K = 0$, see [36], making the results comparable with LPV-SUBNET. Based on the given data, an estimate of (16) is computed with lpvoe in LPVCore, under $n_n = n_b = 5$.

2) LPV PEM-SS identification: This approach corresponds to a direct gradient based estimation of (5a)-(5b) via the minimization of (6), often initialized via sub-space or OE estimation. A model estimate under $n_n = 5$ has been computed by lpsvset in LPVCore using default options and an LPV-OE initialization for the sake of consistency.

3) LPV-SUBNET identification: In order to have a fair comparison, we tune the hyper-parameters for the LPV-SUBNET method similar to the LPV identification methods. Therefore, the output dimension of $\psi_2$, i.e., the state-dimension is set to 5, as well as the lag window $n$. The output dimension of the p-net $\phi_p$, i.e., the scheduling dimension, is set to 3. The encoder $\psi_2$ has 2 hidden layers, each with 64 neurons, tanh activation functions, and a linear bi-pass to capture linear dependencies. The p-net has the same structure as $\psi_2$. The truncation length $T$ is initially set to 5 and has been increased to 60 in the first few epochs to enhance training stability. The batch size during optimization has been 256. We have used the Adam optimizer [30] with the standard configuration settings as implemented in PyTorch.

D. Comparison of the results

Simulation responses of the estimated models are computed on an uncorrelated test data set, generated according to Section IV-B with $N = 3 \cdot 10^4$, and are depicted in Fig. 5 and Fig. 6, respectively. Note that due to the OE noise structure, simulation and prediction responses of the estimated models are equivalent. We use the Best Fit Rate $^1$(BFR) in terms of the simulation accuracy as our criterion for the correctness of the result. Under the OE noise structure and the SNR of 35 dB, we can at most achieve a BFR of 98.22%. Based on Fig. 5, the resulting simulation accuracy of the LPV models shows around 90% of BFR, which shows that the obtained models using these current state-of-the-art LPV identification methods give a relatively accurate representation of the system behavior with the selected scheduling map (14). Computation of these models lasted for a few minutes.

The results for LPV-SUBNET were obtained by training the network on a consumer grade computer for 10 hours with 200,000 batch updates on the estimation data set and early stopping on the validation data set to avoid over-fitting. The resulting simulation accuracy of the trained models shows around 97% of BFR in Fig. 6, which shows that the trained model is an almost perfect approximation of the underlying NL system. Note that this is about 3 times better than the LPV identification results, without requiring a user specified scheduling map. Also note that the same data set has been used for all methods, meaning that the number of data points has been sufficient for both the regular LPV identification methods and also for our LPV-SUBNET method, despite being a deep-learning-based approach. The latter underlines the efficiency of our method. In terms of utilization of the obtained models, the self-scheduled LPV-SUBNET model (Fig. 1) is interesting as it can self-generate its scheduling signal without needing a direct estimation or measurement of internal signals of the represented system, making it useful for simulation purposes. The externally scheduled LPV-SUBNET model (Fig. 2) is of particular

$$^1\text{BFR} = \max\{1 - \frac{1}{N} \sum_{i=1}^{N} \frac{\|y_i - \hat{y}_i\|^2}{\|y_i - \bar{y}\|^2}, 0\} \cdot 100\% \text{ with } \bar{y} \text{ being the sample mean of } y.$$
interest for LPV model-based controller design as it provides a scheduling calculation for controller implementation from direct IO measurements of the NL system.

V. CONCLUSIONS

LPV-SUBNET is the first approach capable of simultaneously estimate an LPV-SS model jointly with scheduling map for a complex NL system, directly from measured input-output data without additional structural information. The resulting scheduling map estimate not only solves the difficult selection procedure for the user, but provides a particularly useful calculation of the scheduling signal, which makes the resulting model directly usable for follow-up analysis and control. LPV-SUBNET has consistency guarantees under a general innovation noise setting and is easily implementable in open-source software packages. Our simulation results on the CMG showcase that the LPV-SUBNET can efficiently achieve highly accurate models.

REFERENCES

[1] J. S. Shamma and M. Athans, “Gain scheduling: potential hazards and possible remedies,” IEEE Control Systems Magazine, vol. 12, no. 3, pp. 101–107, 1992.
[2] R. Tóth, Modeling and Identification of Linear Parameter-Varying Systems, ser. Lecture Notes in Control and Information Sciences, Vol. 403. Heidelberg: Springer, 2010.
[3] C. Hoffmann and H. Werner, “Compact LFT-LPV modeling with automated parameterization for efficient LPV controller synthesis,” in Proc. of the 6th Vienna Conf. on Mathematical Modeling, 2009.
[4] A. Sadeghzadeh, B. Sharif, and R. Tóth, “Affine linear parameter-varying embedding of non-linear models with improved accuracy and minimal overbounding,” IET Control Theory & Applications, vol. 14, pp. 3363–3373, 2020.
[5] F. Casella, F. Donida, and M. Lovera, “Automatic generation of LFTs from object-oriented non-linear models with uncertain parameters,” in Proc. of the 6th Vienna Conf. on Mathematical Modeling, 2009.
[6] P. L. dos Santos, C. Novara, D. Rivera, J. Ramos, and T. Perdicoúlis, editors, Linear Parameter-Varying System Identification: New Developments and Trends. Singapore: World Scientific Publishing, 2011.
[7] A. A. Bachnas, R. Tóth, A. Mesbah, and J. Ludlage, “A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study,” Journal of Process Control, vol. 24, pp. 272–285, 2014.
[8] R. Tóth, P. S. Heuberger, and P. M. van den Hof, “Prediction-error identification of LPV systems: present and beyond,” in Control of linear parameter-varying systems with applications. Springer, 2012, pp. 27–58.
[9] Y. Zhao, B. Huang, H. Su, and J. Chu, “Prediction error method for identification of LPV models,” Journal of Process Control, vol. 22, no. 1, pp. 180–193, 2012.
[10] P. B. Cox and R. Tóth, “Linear parameter-varying subspace identification: A unified framework,” Automatica, vol. 123, p. 109296, 2021.