A Deep Recurrent Neural Network model for affine quasi-LPV System identification

Alexander Rehmer[1] and Andreas Kroll[2]

***Abstract*— This paper presents a new model structure and structure selection procedure for the identification of control-oriented affine quasi-LPV (qLPV) models in state-space (SS) representation using Deep Recurrent Neural Networks (RNNs). The proposed model structure is intended to be an alternative to the existing black-box approaches [1, 2] for the case where no state measurements are available and the scheduling variables (and hence the dependence of the time-varying parameters on the scheduling variables) are unknown. Existing identification approaches are not able to incorporate deep neural network (DNN) structures but employ (normalized) radial basis functions (RBFs) to model the dependence of the time-varying parameters on the scheduling variables. This may increase the dimension of the time-varying parameter vector unnecessarily, making parameter estimation more difficult and limiting the model's use for LPV controller synthesis. The proposed identification approach aims to reduce the required dimension of the time-varying parameter vector by using a (Deep) Neural Network (NN) to model the parameter variation. In order to curb the complexity of the resulting nonlinear optimization problem and make the developed model approach useful in real-life applications, a structure selection procedure based on an initialization method developed in [3] is proposed. The performance of the presented approach is demonstrated on a nonlinear system identification problem.**

I. INTRODUCTION

The linear parameter varying (LPV) framework has proven to be a suitable tool for modeling and control of highly complex nonlinear systems [4]. Among all synthesis approaches the polytopic approach is the most popular [5]. However, the polytopic control synthesis approach is limited in that it requires an affine/polytopic LPV SS representation of the plant and the implementation complexity of polytopic LPV controllers grows exponentially with the dimension of the time-varying affine parameter vector [5]. How to arrive at an affine qLPV description of a nonlinear plant is in itself non-trivial. In the absence of a physical model, identification from input-output data is the only viable way. If the state-vector cannot be measured and the dependence of the time-varying parameter vector on the scheduling variables is unknown, nonlinear recurrent model structures, which can always be viewed as a RNN, have to be employed. Parameter estimation for nonlinear recurrent structures is a notoriously hard optimization problem. A good initialization point is critical and second-order optimization methods are recommended [6]. To the best of the authors knowledge, two approaches for identifying affine qLPV models from data using recurrent model structures have been proposed [7]: The local linear approach (RBF-RNN), which non linearly interpolates between multiple local linear models, e.g. [1], and a structured RNN (S-RNN) [8], for which a systematic method for deriving an affine qLPV representation was presented in [2]. Both approaches make use of RBFs to model the time-varying parameters' dependence on the scheduling variables. This enables straightforward initialization and the derivation of stability conditions, respectively. However, depending on the system to be identified, the assumption of an RBF-like dependence can unnecessarily increase the dimension of the time-varying parameter vector required to yield a good model fit, as will be shown in the following. To overcome this constricting assumption, a DNN is instead employed to model the parameter variation. To the authors best knowledge deep RNNs have not yet been proposed for direct data-driven identification of affine qLPV models. Deep Learning techniques have been applied in the context of scheduling dimension reduction for affine qLPV models [9], under the assumption that a qLPV model of the plant already exists. In addition to a deep RNN model for affine qLPV identification an efficient structure selection procedure based on an initialization method for nonlinear SS models [3] is proposed. Via structure selection it will be determined where enhancing the linear model with nonlinear terms aids model performance, in order to keep model complexity (in terms of model parameters) as low as possible.

*This work was not supported by any organization

[1]Alexander Rehmer is with the Department of Measurement and Control, University of Kassel, Kassel, Germany alexander.rehmer@mrt.uni-kassel.de

[2]Andreas Kroll is the Head of the Department of Measurement and Control, University of Kassel, Kassel, Germany andreas.kroll@mrt.uni-kassel.de

II. PRELIMINARIES

A. LPV

A linear parameter-varying (LPV) model is a model whose parameters depend on a time-varying parameter vector $\boldsymbol{\rho}_k \in \mathbb{R}^{n_\rho}$. A general LPV plant in discrete-time SS representation can be written as

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{y}_k \end{bmatrix} = \begin{bmatrix} \mathcal{A}(\boldsymbol{\rho}_k) & \mathcal{B}(\boldsymbol{\rho}_k) \\ \mathcal{C}(\boldsymbol{\rho}_k) & \mathcal{D}(\boldsymbol{\rho}_k) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{u}_k \end{bmatrix} \tag{1}$$

with the state $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$, input $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$, output $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ and matrix-valued functions $\mathcal{A}(\boldsymbol{\rho}_k) : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_x \times n_x}$, $\mathcal{B}(\boldsymbol{\rho}_k) : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_x \times n_u}$, $\mathcal{C}(\boldsymbol{\rho}_k) : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_y \times n_x}$, $\mathcal{D}(\boldsymbol{\rho}_k) : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_y \times n_u}$, which encode the dependence of the system matrices on the time-varying parameters $[\boldsymbol{\rho}_k]_i$ which are arranged in the time-varying parameter vector $\boldsymbol{\rho}_k$. $\boldsymbol{\rho}_k$ is assumed to be an unknown function of $[\boldsymbol{u}_k, \boldsymbol{x}_k]$ and therefore encodes the nonlinear behavior of the plant. In the general LPV representation (1) the system matrices can be arbitrary

nonlinear functions of $\boldsymbol{\rho}_k$, which makes system identification and controller synthesis difficult. Hence, usually models with *affine* parameter dependence are considered as a special case of (1):

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{y}_k \end{bmatrix} = \left(\begin{bmatrix} \boldsymbol{A}_0 & \boldsymbol{B}_0 \\ \boldsymbol{C}_0 & \boldsymbol{D}_0 \end{bmatrix} + \sum_{i=1}^{n_\theta} [\boldsymbol{\theta}_k]_i \begin{bmatrix} \boldsymbol{A}_i & \boldsymbol{B}_i \\ \boldsymbol{C}_i & \boldsymbol{D}_i \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{u}_k \end{bmatrix} \quad (2)$$

with $\boldsymbol{A}_0, \boldsymbol{A}_i \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{B}_0, \boldsymbol{B}_i \in \mathbb{R}^{n_x \times n_u}$, $\boldsymbol{C}_0, \boldsymbol{C}_i \in \mathbb{R}^{n_y \times n_x}$ and $\boldsymbol{D}_0, \boldsymbol{D}_i \in \mathbb{R}^{n_y \times n_u}$. $\boldsymbol{\theta}_k := \boldsymbol{\theta}_k(\boldsymbol{\phi}_k)$ is the *affine* time-varying parameter vector, which depends on the scheduling variables $\boldsymbol{\phi}_k$, which in the qLPV case are a function of the state and possibly the input, i.e. $\boldsymbol{\phi}_k := \boldsymbol{\phi}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)$. This paper provides an identification approach for affine qLPV models, which can then be utilized for polytopic controller synthesis.

B. Identification of Recurrent Model Structures

Recurrent as opposed to feed-forward model structures possess internal states $\hat{\boldsymbol{x}}_k$ that do not have any physical interpretation. Due to their internal feedback they can only be trained in parallel configuration. This corresponds to a nonlinear output error (NOE) model approach and poses a nonlinear optimization problem (3), even if a linear model is to be estimated [10].

$$\begin{aligned} \boldsymbol{\Phi}_{\mathrm{NOE}} = &\arg\min_{\boldsymbol{\Phi}} \mathcal{L}_{\mathrm{NOE}}\left(\mathcal{D}, \mathcal{M}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k; \boldsymbol{\Phi})\right) \\ &\arg\min_{\boldsymbol{\Phi}} \sum_{k=0}^{N} |\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k(\hat{\boldsymbol{x}}_k; \boldsymbol{\Phi})|^2 \end{aligned} \quad (3)$$

Here $\boldsymbol{\Phi}$ are the parameters of the model $\mathcal{M}$ that are subject to optimization given the data $\mathcal{D}$ and the loss function $\mathcal{L}$ which in this paper is always assumed to be the squared error. It has been recognized, that the training of recurrent structures is especially difficult [10]. Although still a subject of research, it seems that bifurcation boundaries in the parameter space and the vanishing/exploding gradient phenomenon cause problematic variations in scale and curvature of the loss function. These problems can be tackled by second-order optimization methods [6], which are much more computationally demanding than first-order optimization methods: In the case study presented in Sec. V a single optimization run took up to 15 hours of computation time on an Intel Core i9. In addition, finding a good initialization $\boldsymbol{\Phi}_{\mathrm{init}}$ or at least one that yields a stable model is a challenge in itself.

III. Black-box approaches for identification of affine qLPV systems

In this section, the two approaches for identifying affine qLPV models with unknown dependence of the time-varying parameter $\boldsymbol{\theta}_k$ on the scheduling variables $\boldsymbol{\phi}_k$ and unknown state-vector $\boldsymbol{x}_k$ are shortly revisited and their major advantages and shortcomings are pointed out. Based on this analysis a model, which allows for the incorporation of DNNs in affine qLPV system identification is proposed.

A. Radial Basis Function Recurrent Neural Network (RBF-RNN)

The approach of approximating nonlinear systems via a weighted combination of local linear models has been successfully applied in countless engineering applications, e.g. [1] and the references therein. A local linear SS model can be written as

$$\begin{bmatrix} \hat{\boldsymbol{x}}_{k+1} \\ \hat{\boldsymbol{y}}_k \end{bmatrix} = \sum_{i=1}^{n_{\boldsymbol{\theta}}} [\boldsymbol{\theta}_k]_i(\boldsymbol{\phi}_k) \begin{bmatrix} \boldsymbol{A}_i & \boldsymbol{B}_i \\ \boldsymbol{C}_i & \boldsymbol{D}_i \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k \\ \boldsymbol{u}_k \end{bmatrix} \quad (4)$$

$\boldsymbol{A}_i$, $\boldsymbol{B}_i$, $\boldsymbol{C}_i$ and $\boldsymbol{D}_i$ can be interpreted as local linearizations of the nonlinear plant and $[\boldsymbol{\theta}_k]_i : \mathbb{R}^{n_\phi} \to \mathbb{R}$ are weighting functions of the local models. In the qLPV setting, scheduling variables are the state and possibly the input $\boldsymbol{\phi}^T = \left[\boldsymbol{u}_k^T, \boldsymbol{x}_k^T\right]$. (4) is equivalent to the widely used Takagi-Sugeno (TS) fuzzy model, where the weighting functions are interpreted as fuzzy basis functions. In the LPV framework the weighting functions correspond to the time-varying parameters. They are defined as normalized RBFs with centers $\boldsymbol{c_i}$ and widths $\boldsymbol{w}_i$:

$$[\boldsymbol{\theta}]_i = \frac{e^{-(\boldsymbol{\phi}_k - \boldsymbol{c}_i)^T \mathrm{diag}(\boldsymbol{w}_i)(\boldsymbol{\phi}_k - \boldsymbol{c}_i)}}{\sum_{i=1}^{n_{\boldsymbol{\theta}}} e^{-(\boldsymbol{\phi}_k - \boldsymbol{c}_i)^T \mathrm{diag}(\boldsymbol{w}_i)(\boldsymbol{\phi}_k - \boldsymbol{c}_i)}}. \quad (5)$$

Main advantages of this model structure are that is has been shown to be a universal approximator and it can be sensibly initialized by setting all local models equal to an identified linear model, which can be easily obtained via subspace identification [1]. Major drawbacks are, that at least two time-varying parameters, i.e. local models, are required to model any nonlinear behavior. Additionally, many local models might be necessary, if the nonlinearity cannot be easily described via superposition of linear models. Increasing the number of local models and hence the number of model parameters in turn makes parameter estimation more difficult, as will be illustrated in the case study in sec. V.

B. Structured RNN (S-RNN) for affine qLPV system identification

The model equations of the S-RNN for affine qLPV identification developed in [2] are given by

$$\begin{aligned} \hat{\boldsymbol{x}}_{k+1} &= \boldsymbol{A}\hat{\boldsymbol{x}}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{A}_1 \boldsymbol{f}_1\left(\boldsymbol{E}_1 \hat{\boldsymbol{x}}_k\right) + \boldsymbol{B}_1 \boldsymbol{f}_2\left(\boldsymbol{E}_2 \boldsymbol{u}_k\right) \\ \boldsymbol{y}_k &= \boldsymbol{C}\hat{\boldsymbol{x}}_k + \boldsymbol{C}_1 \boldsymbol{f}_3\left(\boldsymbol{E}_3 \hat{\boldsymbol{x}}_k\right) \end{aligned} \quad (6)$$

with $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{B} \in \mathbb{R}^{n_x \times n_u}$, $\boldsymbol{A}_1 \in \mathbb{R}^{n_x \times n_\mathrm{A}}$, $\boldsymbol{B}_1 \in \mathbb{R}^{n_x \times n_\mathrm{B}}$, $\boldsymbol{E}_1 \in \mathbb{R}^{n_\mathrm{A} \times n_x}$, $\boldsymbol{E}_2 \in \mathbb{R}^{n_\mathrm{B} \times n_u}$, $\boldsymbol{C} \in \mathbb{R}^{n_y \times n_x}$, $\boldsymbol{C}_1 \in \mathbb{R}^{n_x \times n_\mathrm{C}}$, $\boldsymbol{E}_3 \in \mathbb{R}^{n_\mathrm{C} \times n_x}$, $\boldsymbol{f}_1 : \mathbb{R}^{n_\mathrm{A}} \to \mathbb{R}^{n_\mathrm{A}}$, $\boldsymbol{f}_2 : \mathbb{R}^{n_\mathrm{B}} \to \mathbb{R}^{n_\mathrm{B}}$ and $\boldsymbol{f}_3 : \mathbb{R}^{n_\mathrm{C}} \to \mathbb{R}^{n_\mathrm{C}}$. $\boldsymbol{f}_1$, $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$ are nonlinear activation functions, which cannot be chosen arbitrarily, as will be shown in the following. In order to rewrite the model (6) in affine qLPV representation, the time-

varying parameters $\boldsymbol{\theta}_k$ have to be defined as

$$\begin{aligned}
[\boldsymbol{\theta}_1]_i &= \frac{f_1\left(\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}}\right)}{\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}}}, && i = 1, \ldots, n_\mathrm{A} \\
[\boldsymbol{\theta}_2]_j &= \frac{f_2\left(\boldsymbol{e}_{2,j}\boldsymbol{u}\right)}{\boldsymbol{e}_{2,j}\boldsymbol{u}}, && j = 1, \ldots, n_\mathrm{B} \\
[\boldsymbol{\theta}_3]_l &= \frac{f_3\left(\boldsymbol{e}_{3,l}\hat{\boldsymbol{x}}\right)}{\boldsymbol{e}_{3,l}\hat{\boldsymbol{x}}}, && l = 1, \ldots, n_\mathrm{C}
\end{aligned} \tag{7}$$

$\boldsymbol{e}_{1,i}$ denotes the i-th row of the matrix $\boldsymbol{E}_1$ and so on. With these definitions, (6) can be rewritten in affine qLPV form:

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1} &= \left(\boldsymbol{A} + \sum_{i=1}^{n_\mathrm{A}} \boldsymbol{A}_1 \boldsymbol{e}_{1,i} \left[\boldsymbol{\theta}_{1,k}\right]_i\right) \hat{\boldsymbol{x}}_k + \\
&\qquad \left(\boldsymbol{B} + \sum_{j=1}^{n_\mathrm{B}} \boldsymbol{B}_1 \boldsymbol{e}_{2,j} \left[\boldsymbol{\theta}_{2,k}\right]_j\right) \boldsymbol{u}_k \\
\hat{\boldsymbol{y}}_k &= \left(\boldsymbol{C} + \sum_{l=1}^{n_\mathrm{C}} \boldsymbol{C}_1 \boldsymbol{e}_{3,l} \left[\boldsymbol{\theta}_{3,k}\right]_l\right) \hat{\boldsymbol{x}}_k
\end{aligned} \tag{8}$$

Major advantages of (6) compared to (4) are that it distinguishes between three sets of time-varying parameters $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ and $\boldsymbol{\theta}_3$ which encode nonlinearities in the system, input and output matrix, respectively. This enables the user to choose where to allow for nolinear behavior in the model based on previous knowledge, which can simplify the optimization problem considerably. Also, increasing the dimension of the time-varying parameter vector does not correspond to adding a fully parameterized SS model to the optimization problem. Stability conditions can be derived, which is not possible for the RBF-RNN. However, the definitions of the time-varying parameters (7) exhibit discontinuities at

$$\lim_{\boldsymbol{x}_k \to 0} \left[\boldsymbol{\theta}_1\right]_i, \quad \lim_{\boldsymbol{u}_k \to 0} \left[\boldsymbol{\theta}_2\right]_j, \quad \lim_{\boldsymbol{x}_k \to 0} \left[\boldsymbol{\theta}_3\right]_l \quad \forall\, i, j, l$$

which are only removable if $f_{1,2,3}$ themselves tend to zero, leaving $\tanh()$ as the only choice for $f_{1,2,3}$. This can be verified via L'Hospital's Rule. Choosing $\tanh()$ as activation function however renders all time-varying parameters to be RBFs centered at the origin of $\boldsymbol{x}_k$ or $\boldsymbol{u}_k$ respectively. This can be shown exemplary for $[\boldsymbol{\theta}_1]_i$ by employing the point symmetrie of $\tanh()$

$$\frac{\tanh(\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}})}{\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}}} = \frac{\tanh(\|\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}}\|)}{\|\boldsymbol{e}_{1,i}\hat{\boldsymbol{x}}\|}$$

i.e. $[\boldsymbol{\theta}_1]_i\left(\hat{\boldsymbol{x}}\right) = [\boldsymbol{\theta}_1]_i\left(\|\hat{\boldsymbol{x}}\|\right)$, which is the definition of a RBF centered at the origin. This rather restrictive assumption will limit this models capability to approximate nonlinearities that do not fit this assumption, as will be illustrated in the case study.

IV. Proposed approach

A. Model structure

The proposed model is based on the idea to lift the restrictive assumptions regarding the functional dependency of the time-varying parameters $\boldsymbol{\theta}_k$ on the scheduling variables $\boldsymbol{\phi}_k$. Rather than assuming $\boldsymbol{\theta}_k(\boldsymbol{\phi}_k)$ take the form of (normalized) RBFs, a possibly DNN is employed to estimate $\boldsymbol{\theta}_k(\boldsymbol{\phi}_k)$. The equations of the proposed model structure depicted in Fig. 1 and henceforth called deep LPV-RNN (dLPV-RNN) are given by

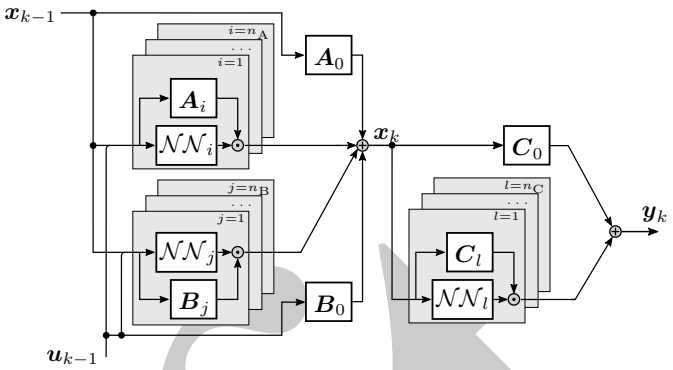

Fig. 1: Proposed model approach (dLPV-RNN) for blackbox qLPV system identification.

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1} &= \left(\boldsymbol{A}_0 + \sum_{i=1}^{n_\mathrm{A}} \boldsymbol{A}_i\, \mathcal{NN}_i\right) \hat{\boldsymbol{x}}_k + \left(\boldsymbol{B}_0 + \sum_{j=1}^{n_\mathrm{B}} \boldsymbol{B}_j\, \mathcal{NN}_j\right) \boldsymbol{u}_k \\
\hat{\boldsymbol{y}}_k &= \left(\boldsymbol{C}_0 + \sum_{l=1}^{n_\mathrm{C}} \boldsymbol{C}_l\, \mathcal{NN}_l\right) \hat{\boldsymbol{x}}_k
\end{aligned} \tag{9}$$

with

$$\begin{aligned}
[\boldsymbol{\theta}_1]_i &= \mathcal{NN}_i\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k\right), && i = 1, \ldots, n_\mathrm{A} \\
[\boldsymbol{\theta}_2]_j &= \mathcal{NN}_j\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k\right), && i = 1, \ldots, n_\mathrm{B} \\
[\boldsymbol{\theta}_3]_l &= \mathcal{NN}_l\left(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k\right), && i = 1, \ldots, n_\mathrm{C}
\end{aligned} \tag{10}$$

$\boldsymbol{A}_0$, $\boldsymbol{B}_0$ and $\boldsymbol{C}_0$ represent the linear, i.e. non-parameter-varying part of the model and can be initialized with an identified linear model as usual. In order to curb the model complexity introduced by modeling each component of the time-varying parameter vector with a dedicated NN, it is reasonably assumed, that many nonlinear systems can be described with a set of mainly time-invariant parameters with only few time-varying parameters. Therefore $\boldsymbol{A}_i$, $\boldsymbol{B}_j$ and $\boldsymbol{C}_l$ do not need to be fully populated, but only need to have non-zero entries, where the linear, i.e. non-parameter varying, parts of the model cannot explain the true systems behavior sufficiently. I.e. $\boldsymbol{A}_i$, $\boldsymbol{B}_j$ and $\boldsymbol{C}_l$ are presumed to be sparse. Hence, instead of broadly viewing all parameters as time-varying, as (4) and (6), the linear model is only selectively enhanced. Finding the appropriate model structure, i.e. non-zero entries in $\boldsymbol{A}_i$, $\boldsymbol{B}_j$ and $\boldsymbol{C}_l$, will be achieved via the structure selection procedure described in the next subsection.

The NNs should be designed as simple as possible, starting with a shallow NN with a few neurons in the hidden layer. If that does not yield the desired performance and a DNN is to be employed, the bottleneck structure depicted in Fig. 2 is suggested in order to aid the automatic feature extraction performed by DNNs. The idea is to let the layers below the bottleneck extract useful low-dimensional scheduling variables $\boldsymbol{\phi}_k$ from $[\boldsymbol{x}_k, \boldsymbol{u}_k]$, if possible. The layers above the bottleneck are then supposed to map the scheduling variables to the time-varying parameter. It should be noted at this point,

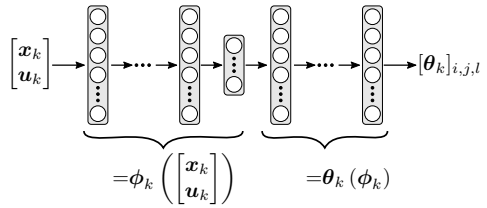

Fig. 2: DNN with bottleneck for extraction of low dimensional scheduling variables in proposed dLPV-RNN.

that although the structure of the network in Fig. 2 resembles that of an autoencoder, in this application it is neither useful to pre-train the proposed network as an autoencoder [11] nor to employ an actual autoencoder instead of the proposed network. An autoencoder strives to learn nonlinear *static* relations in the provided input. Since $\boldsymbol{u}_k$ is an exogenous signal and the states $\boldsymbol{x}_k$ are only dynamically related, no static relations should be present in $[\boldsymbol{x}_k, \boldsymbol{u}_k]$.

B. Initialization and structure selection

In order for the dLPV-RNN to be practically useful in system identification, efficient parameter initialization and structure selection are required. The first will be achieved via the method for parameter initialization for nonlinear SS models presented in [3], the second via a wrapper approach. The initialization procedure presented in [3] is based on the idea to break the recurrence in the unknown states by estimating a state sequence $\boldsymbol{x}_{1:N}^{\mathrm{LS}}$ solving the nonlinear optimization problem in (11) using an identified linear SS model $\mathcal{M}_{\mathrm{lin}} = [\boldsymbol{A}_0, \boldsymbol{B}_0, \boldsymbol{C}_0]$.

$$\begin{aligned} \boldsymbol{x}_{1:N}^{\mathrm{LS}} &= \underset{\boldsymbol{x}_{1:N}}{\arg\min}\, \mathcal{L}_{\mathrm{LS}}\left(\mathcal{D}, \mathcal{M}_{\mathrm{lin}}\right) \\ &= \underset{\boldsymbol{x}_{1:N}}{\arg\min} \sum_{k=0}^{N-1} |\boldsymbol{x}_{k+1} - \boldsymbol{A}_0\boldsymbol{x}_k - \boldsymbol{B}_0\boldsymbol{u}_k|^2 \\ &\quad + \lambda \sum_{k=0}^{N} |\boldsymbol{y}_k - \boldsymbol{C}_0\boldsymbol{x}_k|^2 \end{aligned} \tag{11}$$

λ is a trade-off parameter controlling how much emphasis is given to fit to data vs. linear model fit. Once the state sequence is "known", the parameter optimization problem is no longer recurrent in the states, but an ordinary *static* nonlinear optimization problem. Subsequently the linear part of the model (9) is initialized with the identified linear model and the nonlinear optimization problem

$$\begin{aligned} \boldsymbol{\Phi}_{\mathrm{init}} &= \underset{\boldsymbol{\Phi}}{\arg\min}\, \mathcal{L}_{\mathrm{init}}\left(\mathcal{D}, \mathcal{M}(\boldsymbol{x}_k, \boldsymbol{u}_k; \boldsymbol{\Phi})\right) \\ &= \underset{\boldsymbol{\Phi}}{\arg\min} \sum_{k=0}^{N-1} |\boldsymbol{x}_{k+1}^{\mathrm{LS}} - \hat{\boldsymbol{x}}_{k+1}\left(\boldsymbol{x}_k^{\mathrm{LS}}, \boldsymbol{u}_k; \boldsymbol{\Phi}\right)|^2 \\ &\quad + \sum_{k=0}^{N} |\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k\left(\boldsymbol{x}_k^{\mathrm{LS}}; \boldsymbol{\Phi}\right)|^2 \end{aligned} \tag{12}$$

is solved. From experience the authors suggest to freeze the parameters corresponding to the linear part $\boldsymbol{\Phi}_{\mathrm{lin}} = \{\boldsymbol{A}_0, \boldsymbol{B}_0, \boldsymbol{C}_0\}$ and only optimize the parameters corresponding to the nonlinear behavior $\boldsymbol{\Phi}_{\mathrm{nl}}$. The parameters $\boldsymbol{\Phi}_{\mathrm{init}}$ obtained from (12) are then used as initial parameters for the original problem (3). It should be noted at this point, that (12) yields a model optimized for one-step prediction, which may still be unstable when employed in a simulation setting, i.e. in parallel configuration.

Since (12) poses a simpler and better conditioned optimization problem than (3), it can be solved in a matter of seconds to minutes using first-order methods with, which makes it perfectly suitable for model structure selection. By solving (12) for a number M of candidate structures $\mathcal{M}^m$ and evaluating them on a validation dataset (in parallel configuration) the most promising candidates can be selected before solving the very time-consuming original problem (3).

Algorithm 1: Initialization and structure selection procedure

Input: $\mathcal{D}_{\mathrm{train}} = [\boldsymbol{u}_{\mathrm{train}}, \boldsymbol{y}_{\mathrm{train}}]$, $\mathcal{D}_{\mathrm{val}} = [\boldsymbol{u}_{\mathrm{val}}, \boldsymbol{y}_{\mathrm{val}}]$, $\mathcal{M}_{\mathrm{lin}}$, $\mathcal{M}^m, m = 1, \ldots, M$;
Parameters: $\lambda \in [10^{-3}, 10]$;
$\boldsymbol{x}_{1:N}^{\mathrm{LS}} \leftarrow \underset{\boldsymbol{x}_{1:N}}{\arg\min}\, \mathcal{L}_{\mathrm{LS}}\left(\mathcal{D}_{\mathrm{train}}, \mathcal{M}_{\mathrm{lin}}\right)$;
for $m = 1, \ldots, M$ **do**
  $\boldsymbol{\Phi}_{\mathrm{init}}^m \leftarrow \underset{\boldsymbol{\Phi}}{\arg\min}\, \mathcal{L}_{\mathrm{init}}\left(\mathcal{D}_{\mathrm{train}}, \mathcal{M}^m\right)$;
  $\hat{\boldsymbol{y}}_{\mathrm{val}} \leftarrow \mathcal{M}^m\left(\boldsymbol{y}_{\mathrm{val}}^m; \boldsymbol{\Phi}_{\mathrm{init}}^m\right)$;
  $\mathcal{L}_{\mathrm{val}}^m \leftarrow \|\boldsymbol{y}_{\mathrm{val}} - \hat{\boldsymbol{y}}_{\mathrm{val}}\|_2^2$
end for
Select L candidates with lowest validation error $\mathcal{L}_{\mathrm{val}}$;
for $l = 1, \ldots, L$ **do**
  Initialize $\mathcal{M}^l$ with $\boldsymbol{\Phi}_{\mathrm{init}}^l$;
  $\boldsymbol{\Phi}_{\mathrm{NOE}}^m \leftarrow \underset{\boldsymbol{\Phi}}{\arg\min}\, \mathcal{L}_{\mathrm{NOE}}\left(\mathcal{D}_{\mathrm{train}}, \mathcal{M}^m\right)$;
end for

V. Case Study

A. Test system: Mass-spring-damper system with nonlinear friction

A mass-spring-damper system with a nonlinear LuGre friction model is chosen as nonlinear identification task:

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\frac{c}{m} & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{1}{m} \end{bmatrix} F_{\mathrm{nl}}(x_2) + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \\ y &= x_1 \end{aligned} \tag{13}$$

with

$$\begin{aligned} F_{\mathrm{nl}} &= \sigma_0 z + \sigma_1 \dot{z} \\ \dot{z} &= x_2 - \frac{|x_2|}{g(x_2)} z \\ g(x_2) &= \frac{1}{\sigma_0}\left(F_{\mathrm{c}} + (F_{\mathrm{s}} - F_{\mathrm{c}}) \exp(-(\frac{x_2}{v_{\mathrm{s}}})^2)\right). \end{aligned} \tag{14}$$

Parameter values are listed in TABLE I. As test signal a multisine with randomly distributed odd harmonics, maximal frequency $f_{\max} = 20$ Hz and phases distributed uniformly at random in $[0, 2\pi)$ was chosen. Three different realizations of this multisine signal were generated as input signal for

TABLE I: Parameters of mass-spring-damper system.

m	0.1 kg	σ_1	$2\sqrt{\sigma_0 m}$
c	20 N/m	F_c	0.2 N
d	4 Ns/m	F_s	0.25 N
σ_0	1000	v_s	0.002 m/s

TABLE II: Parameters of candidate structures for the dLPV-RNN.

$\boldsymbol{A}_1$	$\mathcal{NN}_i$ width	depth	f_act	$\dim(\boldsymbol{\phi}_k)$
zero matrix with only one non-zero entry a_{ij}	5	3	{ReLu, tanh, logistic}	$\{1, 2, 5\}$

the system (13), which was simulated with an ode45 solver. As initial state the equilibrium at the origin was chosen. No measurement noise was added to the output. The obtained datasets were used as training, validation and test data, respectively.

B. Identification Procedure

A linear SS model was identified using the Matlab® System Identification Toolbox™ implementation of the N4SID-Algorithm. It's performance on the test data is 71.46 %. The linear model was used to estimate the state sequence and to initialize the appropriate parts of the nonlinear models. All three model approaches where restricted to three internal states $\hat{\boldsymbol{x}}_k$. As is customary the RBF-RNN was initialized by setting all local model parameters equal to the globally identified subspace model. Centers $\boldsymbol{c}_i$ of the RBFs are distributed uniformly at random over the range the scheduling variables $\boldsymbol{\phi}_k = [\boldsymbol{x}_k, \boldsymbol{u}_k]$ took on during simulation of the linear model. Widths $\boldsymbol{w}_i$ were drawn from a normal distribution with zero mean and unit variance. Subsequently, parameters of the RBF-RNN were estimated by solving (3). Results are shown in Fig. 5 for up to 4 local models, i.e. $\dim(n_{\boldsymbol{\theta}}) = 1, \ldots, 4$. Both the S-RNN and the dLPV-RNN were trained according to the procedure laid out in Alg. 1. Considered candidate structures for the dLPV-RNN are listed in TABLE II. The structure of RBF-RNN and S-RNN is entirely determined by the dimension of the time-varying parameters, such that only $n_{\boldsymbol{\theta}}$ and n_A were varied, respectively. $n_{\boldsymbol{\theta}_2}$ and $n_{\boldsymbol{\theta}_3}$ were set to zero by incorporating previous knowledge about the system (13). Implementation and optimization of all models was performed using Casadi [12] and Ipopt [13]. 10 multi-starts were performed for each candidate structure. The performance of each model configuration is measured in terms of the Best Fit Rate (BFR), which corresponds to the coefficient of determination restricted to positive values::

$$\mathrm{BFR} = 100\% \cdot \max\left(1 - \frac{\|\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k\|_2}{\|\boldsymbol{y}_k - \bar{\boldsymbol{y}}\|_2}, 0\right)$$

C. Results

The results of the structure selection are shown in Fig. 3 and 4 for the dLPV-RNN and S-RNN, respectively. Both figures show the BFR of all candidate model structures on the validation dataset. For visualization purposes results for

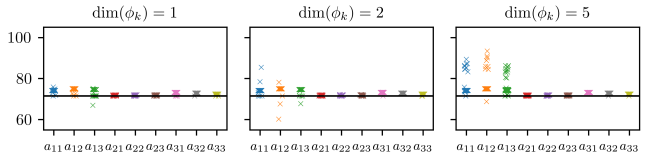

Fig. 3: BFR in % of dLPV-RNN on the validation dataset after initialization (12) depending on non-zero coefficients in $\boldsymbol{A}_1$ and the dimension of the bottleneck layer $\dim(\boldsymbol{\phi}_k)$. Horizontal line indicates linear model fit.

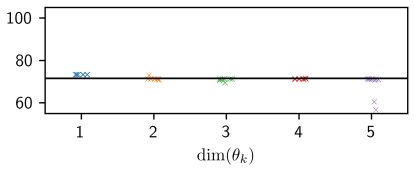

Fig. 4: BFR in % of S-RNN (6) on the validation dataset after initialization (12) depending on $\dim(\boldsymbol{\theta}_k)$. Horizontal line indicates linear model fit.

the dLPV-RNN were only differentiated by the dimension of the bottleneck layer $\dim(\boldsymbol{\phi}_k)$ and the non-zero entries in $\boldsymbol{A}_1$. From the results it can be seen, that none of the S-RNN candidate structures performed significantly better than the initial linear model, indicating that this model structure is not suited to approximate the system at hand. As discussed above, each time-varying parameter is assumed to be a RBF centered at the origin. Hence, increasing $\dim(\boldsymbol{\phi}_k)$ does not increase the models approximative capabilities significantly. The dLPV-RNNs performance is seriously hampered by the bottleneck. However, significant performance gains can be made without a bottleneck layer and with non-zero entries a_{11}, a_{12} and a_{13}, indicating where the time invariant linear model can be significantly enhanced by additional nonlinear terms. Of those candidates who achieved above-average performances, all have tanh activation functions. The ten best candidates of each model were subsequently used as initializations for the original recurrent optimization problem (3). The results are shown in Fig. 5. From Fig. 5 it can be seen, that the proposed dLPV-RNN produces the best results, on average and absolute. The RBF-RNN approach

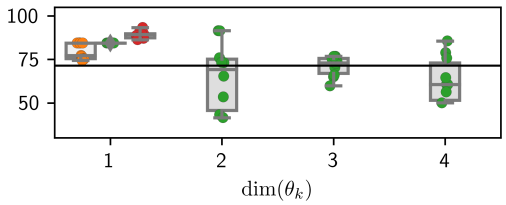

Fig. 5: BFR in % of RBF-RNN (●), S-RNN (●) and dLPV-RNN (●) on the test dataset after NOE optimization (3) depending on $\dim(\boldsymbol{\theta}_k)$. Horizontal line indicates linear model fit.

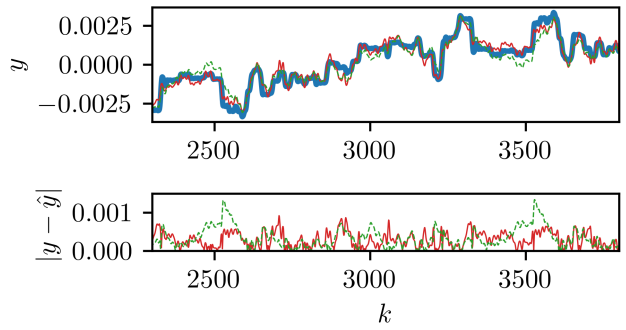

Fig. 6: Simulation (above) and simulation error (below) of RBF-RNN (BFR 91.5 %) and dLPV-RNN (BFR 93.4 %) on the test dataset.

yields the second best model but needs two time-varying parameters. The large variance of the RBF-RNN models performances in contrast to the dLPV-RNN indicates that the employed initialization procedure yields better starting points than the standard procedure for local linear models. The S-RNNs performance increases slightly due to the recurrent optimization, but does not match that of the other models. Fig. 6 shows the simulation and simulation error of the best dLPV-RNN and RBF-RNN. The RBF-RNN shows large deviations from the true output signal, whenever the mass remains in one position and then suddenly breaks loose, e.g. at $k \approx 2500$ and $k \approx 3500$. This indicates, that the velocity-dependent nonlinearity (14) has been better captured by the dLPV-RNN. Hence it has been successfully shown that modeling the time variation with a DNN instead of RBFs can lead to a better model with less time-varying parameters, which will be beneficial for the subsequent controller synthesis step.

VI. Conclusion & Outlook

It has been shown, that existing blackbox model structures for qLPV system identification employ RBFs to model the parameter variation. Since the dimension of the time-varying parameter vector is the limiting factor when using an affine qLPV model for polytopic controller synthesis, a RNN with an affine qLPV structure was proposed. The proposed dLPB-RNN uses a DNN to approximate the parameter variation with the intention to reduce the dimension of the time-varying parameter vector requiered for a good model fit. On a nonlinear system identification task, the proposed model achieved the best fit with only one time-varying parameter. Besides evaluating the proposed model on other identification problems and conducting a thorough empirical study regarding the choice of the activation functions, other initialization procedures should also be considered or developed. Optimizing a recurrent model as a one-step-predictor can still yield an unstable NOE-model, although this has not happened in this case study. Instead, it might be beneficial to capitalize on the structure of the proposed model, which constrains the nonlinear part to act only on select parts of the linear model. It might be possible to formulate the initialization problem as a quadratic or nonlinear program, that when solved guarantees a stable model.

Acknowledgment

Funded by the federal state of Hesse and the European Regional Development Fund (ERDF 2014-2020), Project: Digital Twin of Injection Molding (DIM) FKZ: 0107/20007409

References

[1] V. Verdult. “Nonlinear System Identification: A State-Space Approach”. PhD thesis. University of Twente, 2002.

[2] N. Lachhab et al. “A Neural-Network Based Technique for Modelling and LPV Control of an Arm-Driven Inverted Pendulum”. In: *Proc. of the 47th IEEE Conference on Decision and Control*. 2008, pp. 3860–3865.

[3] A. Marconato et al. “Improved initialization for nonlinear state-space modeling”. In: *IEEE Transactions on instrumentation and Measurement* 63.4 (2013), pp. 972–980.

[4] C. Hoffmann and H. Werner. “A Survey of Linear Parameter-Varying Control Applications Validated by Experiments or High-Fidelity Simulations”. In: *IEEE Transactions on Control Systems Technology* 23 (2015), pp. 416–434.

[5] C. Hoffmann. “Linear Parameter-Varying Control of Systems of High Complexity”. PhD thesis. Technische Universitat Hamburg-Harburg, 2016.

[6] J. Martens. “Second-order optimization for neural networks”. PhD thesis. 2016.

[7] R. Tóth. *Modeling and identification of linear parameter-varying systems*. Vol. 403. Springer, 2010.

[8] P. Gil et al. “On state-space neural networks for systems identification: Stability and complexity”. In: *2006 IEEE Conference on Cybernetics and Intelligent Systems*. IEEE. 2006, pp. 1–5.

[9] P.J.W. Koelewijn and R. Tóth. “Scheduling Dimension Reduction of LPV Models-A Deep Neural Network Approach”. In: *2020 American Control Conference (ACC)*. IEEE. 2020, pp. 1111–1117.

[10] O. Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes*. Berlin Heidelberg, Germany: Springer, 2020.

[11] H. Larochelle et al. “Exploring strategies for training deep neural networks.” In: *Journal of machine learning research* 10.1 (2009).

[12] J. A. E. Andersson et al. “CasADi – A software framework for nonlinear optimization and optimal control”. In: *Mathematical Programming Computation* (2018).

[13] A. Wächter and L. T. Biegler. “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming”. In: *Mathematical programming* 106.1 (2006), pp. 25–57.