An integral architecture for identification of continuous-time state-space LPV models

Manas Mejari *, **Bojan Mavkov** *, **Marco Forgione** *, **Dario Piga** *

* *Dalle Molle Institute for Artificial Intelligence, IDSIA USI-SUPSI, Via la Santa 1, CH-6962 Lugano-Viganello. E-mail: {manas.mejari, bojan.mavkov, marco.forgione, dario.piga}@supsi.ch*

Abstract: This paper presents an *integral* architecture for direct identification of continuous-time *linear parameter-varying* (LPV) state-space models. The main building block of the proposed architecture consist of an LPV model followed by an integral block, which is used to approximate the continuous-time state map of an LPV representation. The unknown LPV model matrices are estimated along with the state sequence by minimizing a properly constructed dual-objective criterion. A coordinate descent algorithm is employed to optimize the desired objective, which alternates between computing the unknown LPV matrices and estimating the state sequence. Thanks to the linear parametric structure induced by the LPV models, the unknown parameters within each coordinate descent step can be computed analytically via ordinary least squares. The effectiveness of the proposed methodology is assessed via a numerical example.

Keywords: Linear Parameter-Varying models, continuous-time identification, state-space identification.

1. INTRODUCTION

Direct identification of *continuous-time* (CT) dynamical systems from sampled data is a mature research topic in the systems and control community. As discussed in Garnier and Young (2014) and Garnier (2015), direct CT identification has multiple advantages over the discrete-time case. Essentially, the majority of physical systems are naturally modelled in continuous-time, and thus, the estimated parameters of CT models usually have a physical interpretation. Direct CT identification methods can also deal with non-uniformly sampled data, while discrete-time models implicitly rely on a fixed sampling time. Moreover, CT identification methods are generally more robust to numerical issues that may arise when using discrete-time methods in the case of high-frequency sampled data. Successful applications and complete reviews of direct CT identification methods can be found in the book Garnier and Wang (2008), and in the works Garnier et al. (2003); Garnier (2015); Lataire et al. (2017); Padilla et al. (2019); Piga (2020) and the references therein. However, most of the available algorithms for CT identification are restricted to the *linear time-invariant* (LTI) model class.

A natural extension to the LTI framework is represented by *linear parameter-varying* (LPV) models. In an LPV model, the property of linearity between input and output signals is preserved in the dynamic relation. However, this relation can change over time according to a measurable time-varying *scheduling* signal. In this way, the behaviour of many nonlinear and time-varying systems can be accurately described via LPV models, while retaining most of the simplicity of the LTI modelling framework.

Most of the identification methods for LPV models, either in input-output form Bamieh and Giarré (2002); Butcher et al. (2008); Laurain et al. (2010); Cerone et al. (2013); Piga et al. (2015); Mejari et al. (2018, 2020); Laurain et al. (2020) or in state-space representations Cox and Tóth (2021); Mejari and Petreczky (2019); Verdult and Verhaegen (2002, 2005), have been developed for discrete-time LPV models. Continuous-time identification of LPV systems has been addressed only in a limited number of contributions, *e.g.,* Mercère et al. (2011); Laurain et al. (2011), which identify LPV models in an input-output form.

From a control perspective, the majority of the controller synthesis approaches require continuous-time LPV state-space models, see *e.g.,* Scherer (1996). This is due to the fact that stability analysis can be done more conveniently with CT state-space representations Zhou and Doyle (1998). However, only few works have addressed the CT identification of LPV state-space models Gáspár et al. (2005); Bergamasco and Lovera (2012); Goos and Pintelon (2016). In Gáspár et al. (2005), CT identification of gray-box quasi-LPV models with an observer-based identification scheme is proposed. In Bergamasco and Lovera (2012) a local approach in the framework of subspace identification is developed. A frequency domain approach is proposed in Goos and Pintelon (2016) under the assumption of a periodic variation of the scheduling signal.

In this paper, we address the problem of direct identification of continuous-time *multi-input multi-output* (MIMO) LPV systems through an *integral* architecture. This architecture consists of an LPV model followed by an integral

block, which is used to approximate the continuous state dynamics of an LPV system. The overall methodology is based on the concept recently introduced by some of the authors in Mavkov et al. (2020) for identification of nonlinear systems through neural networks. In the current contribution, we adapt and specialize this methodology for LPV system identification. The main advantage of the specialized approach w.r.t. the generic one in Mavkov et al. (2020) is the higher computational efficiency. Indeed, the most intensive steps of the training procedure presented in Mavkov et al. (2020) are accelerated in this paper – leveraging particular properties of the LPV model structure – using a coordinate descent algorithm with a closed-form formula for each coordinate descent step.

With the employed integral architecture, we circumvent the need to run time simulations by considering the state sequence as an unknown variable to be optimized along with the state and output mapping matrices, by minimizing a properly constructed cost function. The proposed coordinate-descent algorithm minimizes the cost by successively estimating the state variables and the mapping functions, one at a time. At each step of the coordinate-descent algorithm, a least-squares problem is formulated and the solution is computed analytically.

The rest of the paper is organized as follows. The overall identification setting is outlined in Section 2. The description of the integral architecture and details for the implementation of the optimization algorithm are provided in Section 3. Numerical results are reported in Section 4 to show the effectiveness of the approach. Finally, conclusions and directions for future works are discussed in Section 5.

2. PROBLEM FORMULATION

2.1 Data generating system

We consider a data-generating system $\mathcal{S}$ governed by the following continuous-time MIMO state-space LPV representation:

$$\dot{\mathbf{x}}(t) = \mathcal{A}(\mathbf{p}(t))\mathbf{x}(t) + \mathcal{B}(p(t))\mathbf{u}(t), \tag{1a}$$
$$\mathbf{x}(0) = \mathbf{x}_0, \tag{1b}$$
$$\mathbf{y}^{\mathrm{o}}(t) = \mathcal{C}(\mathbf{p}(t))\mathbf{x}(t) + \mathcal{D}(\mathbf{p}(t))\mathbf{u}(t), \tag{1c}$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ and $\dot{\mathbf{x}}(t) \in \mathbb{R}^{n_x}$ are the state vector and its time derivative, respectively; $\mathbf{x}_0 \in \mathbb{R}^{n_x}$ is the initial state condition; $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ is the system input; $\mathbf{p}(t) \in \mathbb{R}^{n_p}$ is the scheduling signal and $\mathbf{y}^{\mathrm{o}}(t) \in \mathbb{R}^{n_y}$ is the (noise-free) system output at time $t \in \mathbb{R}$. The matrix functions $\mathcal{A}(\cdot), \mathcal{B}(\cdot), \mathcal{C}(\cdot), \mathcal{D}(\cdot)$ are time-varying *affine* functions of the scheduling signal $\mathbf{p}(t)$ defined as:

$$\mathcal{A}(\mathbf{p}(t)) = A_0 + \sum_{i=1}^{n_p} A_i \mathbf{p}_i(t),\ \mathcal{B}(\mathbf{p}(t)) = B_0 + \sum_{i=1}^{n_p} B_i \mathbf{p}_i(t),$$
$$\mathcal{C}(\mathbf{p}(t)) = C_0 + \sum_{i=1}^{n_p} C_i \mathbf{p}_i(t),\ \mathcal{D}(\mathbf{p}(t)) = D_0 + \sum_{i=1}^{n_p} D_i \mathbf{p}_i(t),$$

where $\mathbf{p}_i(t)$ deonotes the i-th component of the scheduling vector $\mathbf{p}(t)$ and $\{A_i, B_i, C_i, D_i\}_{i=0}^{n_p}$ are real, constant matrices of appropriate dimensions.

A training dataset $\mathcal{D}$ of length N is gathered from the LPV system $\mathcal{S}$ defined in (1), at time instants $T = \{t_0 = 0, t_1, \ldots, t_{N-1}\}$. The dataset consists of input, scheduling and *noisy* output samples: $\mathcal{D} = \{\mathbf{u}(t_k), \mathbf{p}(t_k), \mathbf{y}(t_k)\}_{k=0}^{N-1}$. The measured output samples $\mathbf{y}(t_k)$ are corrupted by a zero-mean Gaussian noise η, *i.e.*, $\mathbf{y}(t_k) = \mathbf{y}^{\mathrm{o}}(t_k) + \eta(t_k)$. Finally, we assume that the input $\mathbf{u}(t)$ and the scheduling signal $\mathbf{p}(t)$ can be reconstructed (or reasonably approximated) for all time instants $t \in [0\ \ t_{N-1}] \subset \mathbb{R}$ from the measured samples $\{\mathbf{u}(t_k), \mathbf{p}(t_k)\}_{k=0}^{N-1}$.

Given the training dataset $\mathcal{D}$, our goal is to identify a continuous-time LPV state space affine model, such that the model output matches closely with the measured system output $\mathbf{y}(t)$. To this end, we define an *integral* architecture for the identification of continuous-time LPV models in the following section.

3. CONTINUOUS-TIME IDENTIFICATION OF LPV MODELS

3.1 Integral architecture

In order to describe the continuous-time state dynamics in (1a), we define an LPV block $\mathcal{M}_x(\hat{\mathbf{x}}, \mathbf{u}, \mathbf{p})$, which is fed by the system input $\mathbf{u}(t)$, scheduling signal $\mathbf{p}(t)$ and the (estimated) state $\hat{\mathbf{x}}(t)$ at time t, and returns the estimated state time-derivative $\dot{\hat{\mathbf{x}}}(t)$, *i.e.*,

$\mathcal{M}_x(\hat{\mathbf{x}}, \mathbf{u}, \mathbf{p})$:

$$\dot{\hat{\mathbf{x}}} = \left(\hat{A}_0 + \sum_{i=1}^{n_p} \hat{A}_i \mathbf{p}_i(t)\right)\hat{\mathbf{x}}(t) + \left(\hat{B}_0 + \sum_{i=1}^{n_p} \hat{B}_i \mathbf{p}_i(t)\right)\mathbf{u}(t), \tag{2}$$

where $\hat{A}_i \in \mathbb{R}^{n_x \times n_x}$ and $\hat{B}_i \in \mathbb{R}^{n_x \times n_u}$ (for $i = 0, \ldots, n_p$) are the model matrices to be identified.

Similarly, the output equation in (1c) is represented by another LPV block $\mathcal{M}_y(\hat{\mathbf{x}}, \mathbf{u}, \mathbf{p})$, which is fed by the estimated state $\hat{\mathbf{x}}(t)$, input $\mathbf{u}(t)$ and scheduling signal $\mathbf{p}(t)$ and returns the model output $\hat{\mathbf{y}}(t)$ at time t, *i.e.*,

$\mathcal{M}_y(\hat{\mathbf{x}}, \mathbf{u}, \mathbf{p})$:

$$\hat{\mathbf{y}}(t) = \left(\hat{C}_0 + \sum_{i=1}^{n_p} \hat{C}_i \mathbf{p}_i(t)\right)\hat{\mathbf{x}}(t) + \left(\hat{D}_0 + \sum_{i=1}^{n_p} \hat{D}_i \mathbf{p}_i(t)\right)\mathbf{u}(t), \tag{3}$$

where $\hat{C}_i \in \mathbb{R}^{n_y \times n_x}$ and $\hat{D}_i \in \mathbb{R}^{n_y \times n_u}$ (for $i = 0, \ldots, n_p$) have to be estimated from data.

To simplify the notations, we introduce the following matrices stacking the model parameters in their columns:

$$\Theta_x = \left[\hat{A}_0 \cdots \hat{A}_{n_p}\ \hat{B}_0 \cdots \hat{B}_{n_p}\right] \in \mathbb{R}^{n_x \times (n_p+1)(n_x+n_u)}$$

and

$$\Theta_y = \left[\hat{C}_0 \cdots \hat{C}_{n_p}\ \hat{D}_0 \cdots \hat{D}_{n_p}\right] \in \mathbb{R}^{n_y \times (n_p+1)(n_x+n_u)}.$$

The resulting continuous-time LPV state-space affine model is then given by:

$$\dot{\hat{\mathbf{x}}}(t) = \mathcal{M}_x(\hat{\mathbf{x}}(t), \mathbf{u}(t), \mathbf{p}(t); \Theta_x), \tag{4a}$$
$$\hat{\mathbf{x}}(0) = \mathbf{x}_0, \tag{4b}$$
$$\hat{\mathbf{y}}(t) = \mathcal{M}_y(\hat{\mathbf{x}}(t), \mathbf{u}(t), \mathbf{p}(t); \Theta_y). \tag{4c}$$

We note that, from (2) and (3), the maps $\mathcal{M}_x(\cdot)$ and $\mathcal{M}_y(\cdot)$ are *linear* functions of the parameters Θ_x and Θ_y, respectively given by:

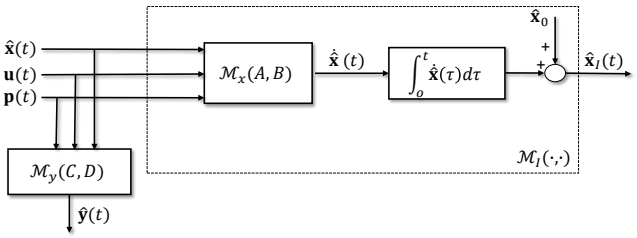

Fig. 1. Integral architecture for LPV identification.

$$\mathcal{M}_x(\hat{\mathbf{x}}(t),\mathbf{u}(t),\mathbf{p}(t);\Theta_x) = \Theta_x \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t) \end{bmatrix} \otimes \hat{\mathbf{x}}(t) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t) \end{bmatrix} \otimes \mathbf{u}(t) \end{bmatrix}, \tag{5a}$$

$$\mathcal{M}_y(\hat{\mathbf{x}}(t),\mathbf{u}(t),\mathbf{p}(t);\Theta_y) = \Theta_y \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t) \end{bmatrix} \otimes \hat{\mathbf{x}}(t) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t) \end{bmatrix} \otimes \mathbf{u}(t) \end{bmatrix}, \tag{5b}$$

where $\otimes$ denotes the Kronecker product between the two vectors.

A possible approach to fit the LPV model parameters to the training dataset $\mathcal{D}$ is to simulate (4) using a numerical ODE solution scheme, and then to minimize the *simulation error* norm penalizing the distance between measured $\mathbf{y}$ and simulated outputs $\hat{\mathbf{y}}$, with respect to the unknown parameters Θ_x and Θ_y.

In this paper, we adopt an alternative method first introduced in Mavkov et al. (2020), which exploits the *integral form* of the Cauchy problem (4a)-(4b), by defining an *integral* LPV block $\mathcal{M}_I$ as:

$$\hat{\mathbf{x}}_I(t) = \mathcal{M}_I(\hat{\mathbf{x}}(t),\mathbf{u}(t),\mathbf{p}(t);\Theta_x) \tag{6}$$

with

$$\mathcal{M}_I(\hat{\mathbf{x}}(t),\mathbf{u}(t),\mathbf{p}(t);\Theta_x) = \hat{\mathbf{x}}(0) + \int_0^t \mathcal{M}_x(\hat{\mathbf{x}}(\tau),\mathbf{u}(\tau),\mathbf{p}(\tau);\Theta_x)d\tau.$$

If the state sequence $\hat{\mathbf{x}}(t)$ feeding the LPV model block $\mathcal{M}_I(\cdot)$ is actually generated by the model given in (4), then the state $\hat{\mathbf{x}}_I(t)$ exactly matches $\hat{\mathbf{x}}(t)$, i.e.,

$$\hat{\mathbf{x}}(t) = \hat{\mathbf{x}}_I(t) \qquad \forall t \in [t_0\ t_{N-1}]. \tag{7}$$

The block diagram in Fig. 1 is a representation of (6), along with the output equation (4c) producing $\hat{\mathbf{y}}(t)$.

3.2 Fitting criterion

In the proposed methodology, the LPV model matrices Θ_x, Θ_y and the state signal $\hat{\mathbf{x}}(t), t \in [t_0\ \ t_{N-1}]$, are *free* optimization parameters. They are jointly optimized according to a *dual* objective constructed according to the following rationale. First, the estimated model output $\hat{\mathbf{y}}$ should match the output measurements in training dataset $\mathcal{D}$. This objective is achieved by introducing a *fitting term* in the cost function which penalizes the distance between the model output $\hat{\mathbf{y}}(t_k)$ and the measured output $\mathbf{y}(t_k)$, $k = 0, 1, \ldots, N-1$. Second, the state signal $\hat{\mathbf{x}}$ should be compatible with the model dynamics (4). This can be achieved through an additional *regularization term* which penalizes the distance between $\hat{\mathbf{x}}_I(t)$ and $\hat{\mathbf{x}}(t)$, where $\hat{\mathbf{x}}_I$ is defined as in (6).

The following minimization problem is thus formulated:

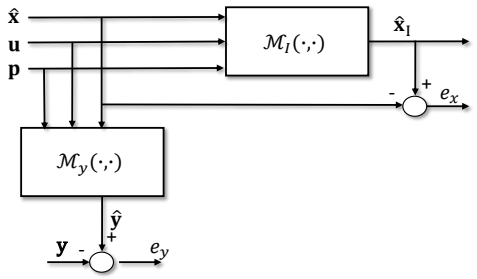

Fig. 2. Block diagram representing computation of the cost function for the proposed integral LPV architecture.

$$\min_{\hat{\mathbf{x}}(\cdot),\Theta_x,\Theta_y} J(\hat{\mathbf{x}}(\cdot),\Theta_x,\Theta_y), \tag{8a}$$

where

$$J = \underbrace{\sum_{k=0}^{N-1} \|\overbrace{\hat{\mathbf{y}}(t_k) - \mathbf{y}(t_k)}^{\mathbf{e}_y}\|^2}_{J_{\mathbf{y}}} + \alpha \underbrace{\int_{t_0}^{t_{N-1}} \|\overbrace{\hat{\mathbf{x}}_I(\tau) - \hat{\mathbf{x}}(\tau)}^{\mathbf{e}_x}\|^2\, d\tau}_{J_{\mathbf{x}}}, \tag{8b}$$

with

$$\hat{\mathbf{y}}(t_k) = \mathcal{M}_y(\hat{\mathbf{x}}(t_k),\mathbf{u}(t_k),\mathbf{p}(t_k);\ \Theta_y), \tag{8c}$$

$$\hat{\mathbf{x}}_I(t) = \hat{\mathbf{x}}(0) + \int_0^t \mathcal{M}_x(\hat{\mathbf{x}}(\tau),\mathbf{u}(\tau),\mathbf{p}(\tau);\ \Theta_x)\, d\tau. \tag{8d}$$

The hyper-parameter $\alpha > 0$ acts as a tuning knob balancing the relative importance of the fitting cost term J_y and the regularization cost term J_x. The operations required to compute the cost J in (8b) are depicted in Fig. 2.

3.3 Optimization algorithm

It is important to note that the optimization problem (8) is *infinite-dimensional* and thus computationally intractable. Indeed, the continuous-time state signal $\hat{\mathbf{x}}(t) \in \mathbb{R}^{n_x},\ t \in [t_0\ t_{N-1}]$ is one of the problem's decision variables.

Following the rationale in Mavkov et al. (2020), we employ numerical techniques in order to transform (8) into a finite-dimensional problem amenable for software implementation. In particular, the state signal $\hat{\mathbf{x}}(t)$ is approximated using a finite-dimensional parametrization. For the sake of simplicity of exposition, we represent the state signal with a *piecewise constant* parametrization, where $\hat{\mathbf{x}}(t)$ is constant in the intervals $[t_k\ t_{k+1}]$, $k = 0, 1, \ldots, N-1$. Such parameterization approximates the signal reasonably well for short sampling intervals.

We remark that, in general, more complex parametrizations for $\hat{\mathbf{x}}$ such as piecewise linear or polynomial could be used. Moreover, the intervals for the piecewise constant approximation of $\hat{\mathbf{x}}$ may not necessarily correspond to the ones induced by the measurements in $\mathcal{D}$.

Furthermore, we approximate the integrals in (8b) and (8d) by applying a numerical integration scheme. For simplicity, in this work we apply the classical *rectangular approximation* rule for the numerical integration of both integrals. In general, other quadrature rules such as the

trapezoidal rule or *Gaussian quadrature* Quarteroni et al. (2010) could be considered.

Overall, the *piecewise constant* parametrization of the signals $\hat{\mathbf{x}}(t)$, $\mathbf{u}(t)$, $\mathbf{p}(t)$ with the *rectangular quadrature* of the integrals leads to:

$$\int_{t_0}^{t_{N-1}} \|\hat{\mathbf{x}}_I(\tau) - \hat{\mathbf{x}}(\tau)\|^2 d\tau \approx \sum_{k=1}^{N-1} \|\hat{\mathbf{x}}_I(t_k) - \hat{\mathbf{x}}(t_k)\|^2 \Delta t_k, \tag{9}$$

where $\Delta t_k = t_k - t_{k-1}$, and (8d) can be approximated with the following Riemann sum as:

$$\hat{\mathbf{x}}_I(t_k) \approx \hat{\mathbf{x}}(0) + \sum_{j=0}^{k-1} \Delta t_{j+1} \mathcal{M}_x(\hat{\mathbf{x}}(t_j), \mathbf{u}(t_j), \mathbf{p}(t_j);\ \Theta_x). \tag{10}$$

Note that, for computational convenience, the sum in (10) can be constructed recursively as:

$$\hat{\mathbf{x}}_I(t_k+1) = \hat{\mathbf{x}}_I(t_k) + \overbrace{\Delta t_{k+1} \mathcal{M}_x(\hat{\mathbf{x}}(t_k), \mathbf{u}(t_k), \mathbf{p}(t_k)); \Theta_x)}^{\Delta \mathbf{x}_k}. \tag{11}$$

From the linear dependence of the map $\mathcal{M}_x(\cdot)$ on Θ_x in (5a), equation (11) can be re-written as:

$$\hat{\mathbf{x}}_I(t_k+1) \ = \ \hat{\mathbf{x}}_I(t_k) + \Delta t_{k+1} \Theta_x \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t_k) \end{bmatrix} \otimes \hat{\mathbf{x}}(t_k) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t_k) \end{bmatrix} \otimes \mathbf{u}(t_k) \end{bmatrix}. \tag{12}$$

Similarly, as the map $\mathcal{M}_y(\cdot)$ is a linear function of Θ_y, substituting (5b) in (8c) yields:

$$\hat{\mathbf{y}}(t_k) = \Theta_y \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t_k) \end{bmatrix} \otimes \hat{\mathbf{x}}(t_k) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t_k) \end{bmatrix} \otimes \mathbf{u}(t_k) \end{bmatrix}. \tag{13}$$

With the above considerations, the cost function J in (8a) is minimized w.r.t. the parameters $\{\hat{\mathbf{x}}, \Theta_x, \Theta_y\}$ by using the coordinate-descent approach described in Algorithm 1. With a slight abuse of notation, the optimization variable $\hat{\mathbf{x}}$ in Algorithm 1 denotes the finite-dimensional representation of the state signal $\hat{\mathbf{x}}$, i.e., $\hat{\mathbf{x}} = \{\hat{\mathbf{x}}(t_0), \ldots, \hat{\mathbf{x}}(t_{N-1})\}$.

Given an initial guess $\hat{\mathbf{x}}^{(0)}$ of the state sequence, at each iteration $n \geq 1$, Algorithm 1 alternates between two steps: Step 1.1 and Step 1.2. In particular, at Step 1.1, model parameters Θ_x, Θ_y are computed by solving (8a) for a *fixed* state sequence $\hat{\mathbf{x}}^{(n-1)}$ obtained at the iteration $(n-1)$. Subsequently, at Step 1.2, the state sequence $\hat{\mathbf{x}}^{(n)}$ is estimated by minimizing the cost (8a) for *fixed* model parameters $\Theta_x^{(n)}$ and $\Theta_y^{(n)}$ obtained from Step 1.1 at the n-th iteration. The procedure continues until a maximum number of iterations is reached, or a certain convergence criterion is met (Step 2).

We stress that, the solutions at Step 1.1 and 1.2 of Algorithm 1 are computed *analytically* through linear least-squares. For example, at Step 1.1, given $\hat{\mathbf{x}}$, the model parameters Θ_x and Θ_y can be computed from the solution of the following least-squares problems:

$$\Theta_x' = (\mathbf{\Phi}_x' \mathbf{\Phi}_x)^{-1} \mathbf{\Phi}_x' X, \quad \Theta_y' = (\mathbf{\Phi}_y' \mathbf{\Phi}_y)^{-1} \mathbf{\Phi}_y' Y, \tag{14}$$

with $X = \begin{bmatrix} (\hat{\mathbf{x}}(t_1) - \hat{\mathbf{x}}(t_0)) & \ldots & (\hat{\mathbf{x}}(t_{N-1}) - \hat{\mathbf{x}}(t_0)) \end{bmatrix}'$ and $Y = \begin{bmatrix} (\hat{\mathbf{y}}(t_0))' & \ldots & (\hat{\mathbf{y}}(t_N))' \end{bmatrix}'$. The matrices $\mathbf{\Phi}_x$ and $\mathbf{\Phi}_y$ are constructed such that their k-th *block row* is given by:

Algorithm 1 Coordinate descent for the estimation of states $\hat{\mathbf{x}}$ and model parameter matrices Θ_x, Θ_y; tolerance $\epsilon > 0$; maximum number of iterations $n_{\max}$

Input: Training dataset $\mathcal{D} = \{\mathbf{u}(t_k), \mathbf{y}(t_k), \mathbf{p}(t_k)\}_{k=0}^{N-1}$; initial guess $\hat{\mathbf{x}}^{(0)}$; tuning parameter α; tolerance ϵ, maximum number of iteration $n_{\max}$.

1. **Iterate for** $n = 1, \ldots$
   1.1. $\Theta_x^{(n)}, \Theta_y^{(n)} \leftarrow \arg\min_{\Theta_x, \Theta_y} J(\hat{\mathbf{x}}^{(n-1)}, \Theta_x, \Theta_y)$
   1.2. $\hat{\mathbf{x}}^{(n-1)} \leftarrow \operatorname{argmin}_{\hat{\mathbf{x}}} \ J(\hat{\mathbf{x}}, \Theta_x^{(n)}, \Theta_y^{(n)})$
2. **Until** $\|\hat{\mathbf{x}}^{(n)} - \hat{\mathbf{x}}^{(n-1)}\| \leq \epsilon$ or $n = n_{\max}$

Output: Estimates $\{\hat{\mathbf{x}}(t_k)\}_{k=0}^{N-1}$ and Θ_x, Θ_y.

$$\mathbf{\Phi}_x[k,:] = \sum_{j=0}^{k-1} \Delta t_{j+1} \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t_j) \end{bmatrix} \otimes \hat{\mathbf{x}}(t_j) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t_j) \end{bmatrix} \otimes \mathbf{u}(t_j) \end{bmatrix},$$

$$\mathbf{\Phi}_y[k,:] = \sum_{j=0}^{k-1} \begin{bmatrix} \begin{bmatrix} 1 \\ \mathbf{p}(t_j) \end{bmatrix} \otimes \hat{\mathbf{x}}(t_j) \\ \begin{bmatrix} 1 \\ \mathbf{p}(t_j) \end{bmatrix} \otimes \mathbf{u}(t_j) \end{bmatrix},$$

where we introduce the notation $\mathbf{\Phi}_x[k,:]$ and $\mathbf{\Phi}_y[k,:]$ to refer to the k-th *block row* of respective matrices. Similarly, at Step 1.2, given the model parameters Θ_x and Θ_y, the state sequence $\hat{\mathbf{x}}$, can be computed via ordinary least-squares.

4. NUMERICAL EXAMPLE

In this section, we demonstrate the effectiveness of the proposed method via a numerical example. All computations are carried out on an i7 1.9-GHz Intel core processor with 32 GB of RAM running MATLAB R2019a.

We consider a benchmark MIMO LPV state-space system considered in Verdult and Verhaegen (2005); Cox and Tóth (2021), modified to the continuous-time setting. The benchmark LPV system has input dimension $n_u = 2$, scheduling signal dimension $n_p = 2$, state dimension $n_x = 2$, and output dimension $n_y = 2$, with the following system matrices:

$$[\, A_0 \ A_1 \ A_2 \,] = \left[\begin{array}{cc|cc|cc} 0 & 0.5 & 0.4 & 0 & 0.2 & 0 \\ -0.5 & 0 & 0 & 0.3 & 0 & 0 \end{array}\right],$$

$$B_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C_0 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, D_0 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}.$$

The data are generated by simulating [1] the continuous-time LPV system (1) and sampling the outputs, inputs and scheduling signals with a sampling time of 0.01 s. The input trajectories vary in the domain $\mathbf{u}(t) \in [-2, 2] \times [-2, 2]$ and the scheduling signal varies within the box $\mathbf{p}(t) \in [-1, 1] \times [-1, 1]$. The output measurements are corrupted by a zero-mean white Gaussian noise η with distribution $\eta(t) \sim \mathcal{N}(0, \Sigma)$ where Σ is diagonal. In order to analyze the statistical properties of the proposed identification algorithm, a Monte-Carlo study with 50 runs is performed. At each Monte-Carlo run, a different realization of input $\mathbf{u}(t)$, scheduling signal $\mathbf{p}(t)$ and noise $\eta(t)$ is considered. We carry out two simulation studies

[1] The trajectories of the data-generating system $\mathcal{S}$ in (1) are evaluated using MATLAB's `ode45` function which employs 4-th order Runge-Kutta method with an adaptive step size.

involving different noise conditions. In particular, the noise covariance matrix Σ is chosen such that for each Monte-Carlo run, the *signal-to-noise* ratio (SNR)

$$\mathrm{SNR}_{\mathbf{y}}^{[i]} = 10 \log \frac{\sum_{t_k=0}^{N-1} \left(\mathbf{y}_i(t_k) - \eta_i(t_k)\right)^2}{\sum_{t_k=0}^{N-1} \left(\eta_i(t_k)\right)^2},$$

is set to $\mathrm{SNR}_{\mathbf{y}}^{[i]} = \{15, 25\}$ dB for all output channels $i = 1, \ldots, n_y$. Here, the subscript i denotes the i-th element of the vector signal, and $\mathrm{SNR}_{\mathbf{y}}^{[i]}$ corresponds to the SNR of the output channel $\mathbf{y}_i$. For each Monte-Carlo run, a training dataset of $N = 800$ samples and a *noise-free* validation dataset of $N_{\mathrm{val}} = 500$ samples is gathered. The quality of the estimated LPV model is assessed via *Best Fit Rate* (BFR) index defined as

$$\mathrm{BFR}_{\mathbf{y}}^{[i]} = \max \left\{ 1 - \sqrt{\frac{\sum_{t_k=0}^{N_{\mathrm{val}}-1} (\mathbf{y}_i(t_k) - \hat{\mathbf{y}}_i(t_k))^2}{\sum_{t_k=0}^{N_{\mathrm{val}}-1} (\mathbf{y}_i(t_k) - \bar{\mathbf{y}}_i)^2}}, 0 \right\} \times 100\%,$$

with $\hat{\mathbf{y}}_i$ being the simulated model output and $\bar{\mathbf{y}}_i$ is the sample mean of the output over the validation set.

For identification, we consider an LPV state-space affine model structure (4) with state dimension set to the true system dimension $n_x = 2$. The LPV model matrices Θ_x, Θ_y and the state sequence $\hat{\mathbf{x}}$ are estimated by running the coordinate descent Algorithm 1 for $n_{\max} = 30$ iterations, with initial guess $\hat{\mathbf{x}}^{(0)}$ of the state sequence chosen randomly from a uniform distribution $\mathcal{U}(-0.1, 0.1)$. The regularization parameter α can be chosen via cross-validation. In this example, this parameter is set to $\alpha = 1$. The average computation time for each iteration is 10.8 ms, out of which, only 0.32 ms are required to compute the model parameters Θ_x, Θ_y (Step 1.1) and 10.5 ms are spent to compute the state sequence $\hat{\mathbf{x}}$ (Step 1.2). We remark that, the difference in the computation time between the two steps of the coordinate descent algorithm is expected. This is due to the fact that the LS problem at Step 1.2 involves a significantly higher number of optimization variables $\hat{\mathbf{x}}$, namely Nn_x, than the number of optimization variables at Step 1.1, since typically $N \gg (n_x, n_p, n_u, n_y)$. More efficient ways to compute $\hat{\mathbf{x}}$ at Step 1.2 are currently under investigation.

The mean and standard deviations of the cost J is plotted in Fig. 3 against the iterations of the coordinate descent algorithm for 50 Monte-Carlo runs. The estimated model outputs over the validation dataset are plotted in Fig. 4, for the case of 15 dB SNR. For the sake of better visualization, only a subset of the simulated output has been plotted. It can be observed that convergence of the coordinate descent algorithm is achieved in about 30 iterations for all runs, and the reconstructed outputs match closely with the true one.

Furthermore, the boxplots of the achieved BFR indices over different Monte-Carlo runs, for varying SNR are plotted in the Fig. 5. For both of the noise scenarios under considerations, *i.e.*, with SNR equal to 15 dB and 25 dB, the estimated models achieve a good performance in terms of BFR index for most of the Monte-Carlo runs, with only a few outliers. Overall, satisfactory performance has been achieved by the proposed algorithm with a good computational efficiency.

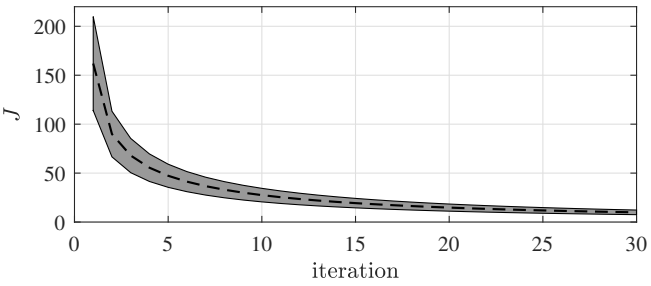

Fig. 3. Training: Mean (dashed black) and std deviation (shaded gray area) of the cost *vs* number of iterations over the Monte Carlo runs.

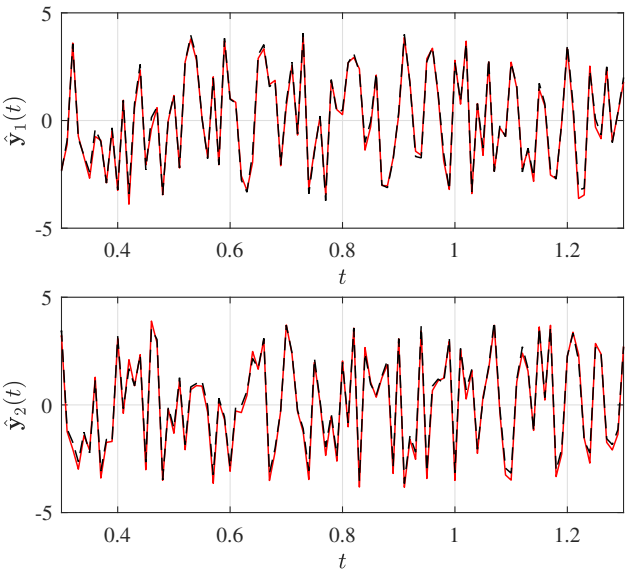

Fig. 4. Validation: True output (red) *vs* estimated outputs (dashed black) for output channel 1 (Top panel) and output channel 2 (Bottom panel).

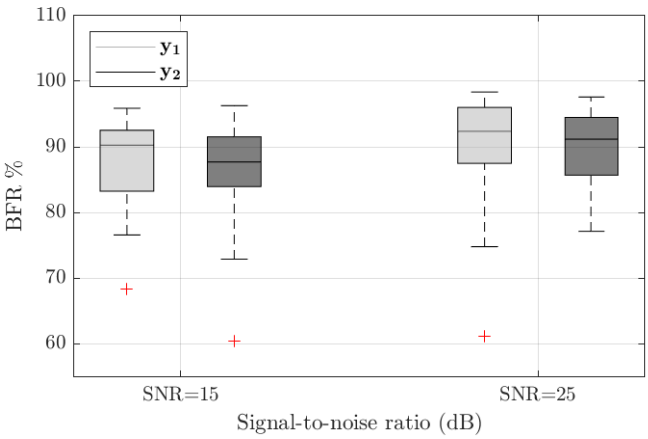

Fig. 5. Monte-Carlo analysis: boxplots of achived BFR indices for output channels $\mathbf{y}_1$ and $\mathbf{y}_2$ with SNR levels 15, 25 dB.

5. CONCLUSION

We have presented an integral architecture for continuous-time identification of LPV models in state-space form. A coordinate descent algorithm tailored for the proposed

LPV model architecture is presented, which alternates between computation of the model parameters and state estimation. The main advantage of the proposed algorithm is the high computational efficiency, as the solution to the sub-problems iteratively constructed at each step of the coordinate descent algorithm can be obtained in closed form. The numerical example demonstrates the effectiveness of the proposed approach in reproducing the true system outputs. Future research directions will be focused on developing a more efficient methods to optimize the state estimation step in the proposed algorithm.

REFERENCES

Bamieh, B.A. and Giarré, L. (2002). Identification of linear parameter-varying models. *International Journal of Robust Nonlinear Control*, 12(9), 841–853.

Bergamasco, M. and Lovera, M. (2012). Subspace identification of continuous-time state-space LPV models. *Linear Parameter-Varying System Identification*, 201–230.

Butcher, M., Karimi, A., and Longchamp, R. (2008). On the consistency of certain identification methods for linear parameter varying systems. In *Proc. of the 17th IFAC World Congress*, 4018–4023. Seoul, South Korea.

Cerone, V., Piga, D., and Regruto, D. (2013). A convex relaxation approach to set-membership identification of LPV systems. *Automatica*, 49(9), 2853–2859.

Cox, P.B. and Tóth, R. (2021). Linear parameter-varying subspace identification: A unified framework. *Automatica*, 123, 109296.

Garnier, H. (2015). Direct continuous-time approaches to system identification. overview and benefits for practical applications. *European Journal of control*, 24, 50–62.

Garnier, H., Mensler, M., and Richard, A. (2003). Continuous-time model identification from sampled data: implementation issues and performance evaluation. *International journal of Control*, 76(13), 1337–1357.

Garnier, H. and Wang, L. (2008). *Identification of Continuous-time Models from Sampled Data*. Springer Publishing Company, Incorporated, 1st edition.

Garnier, H. and Young, P.C. (2014). The advantages of directly identifying continuous-time transfer function models in practical applications. *International Journal of Control*, 87(7), 1319–1338. doi: 10.1080/00207179.2013.840053.

Gáspár, P., Szabó, Z., and Bokor, J. (2005). Gray-box continuous-time parameter identification for lpv models with vehicle dynamics applications. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, 393–398. IEEE.

Goos, J. and Pintelon, R. (2016). Continuous-time identification of periodically parameter-varying state space models. *Automatica*, 71, 254 – 263.

Lataire, J., Pintelon, R., Piga, D., and Tóth, R. (2017). Continuous-time linear time-varying system identification with a frequency-domain kernel-based estimator. *IET Control Theory Applications*, 11(4), 457–465.

Laurain, V., Gilson, M., Tóth, R., and Garnier, H. (2010). Refined instrumental variable methods for identification of LPV Box–Jenkins models. *Automatica*, 46(6), 959–967.

Laurain, V., Tóth, R., Gilson, M., and Garnier, H. (2011). Direct identification of continuous-time linear parameter-varying input/output models. *IET Control Theory & Applications*, 5(7), 878–888.

Laurain, V., Tóth, R., Piga, D., and Darwish, M.A.H. (2020). Sparse RKHS estimation via globally convex optimization and its application in LPV-IO identification. *Automatica*, 115.

Mavkov, B., Forgione, M., and Piga, D. (2020). Integrated neural networks for nonlinear continuous-time system identification. *IEEE Control Systems Letters*, 4(4), 851–856.

Mejari, M., Naik, V., Piga, D., and Bemporad, A. (2020). Identification of hybrid and linear parameter-varying models via piecewise affine regression using mixed integer programming. *International Journal of Robust Nonlinear Control*, 30, 5802–5819.

Mejari, M. and Petreczky, M. (2019). Realization and identification algorithm for stochastic LPV state-space models with exogenous inputs. *3rd IFAC Workshop on Linear Parameter Varying Systems LPVS 2019*, 52(28), 13 – 19.

Mejari, M., Piga, D., and Bemporad, A. (2018). A bias-correction method for closed-loop identification of Linear Parameter-Varying systems. *Automatica*, 87, 128–141.

Mercère, G., Palsson, H., and Poinot, T. (2011). Continuous-time linear parameter-varying identification of a cross flow heat exchanger: a local approach. *IEEE Transactions on Control Systems Technology*, 19(1), 64–76.

Padilla, A., Garnier, H., Young, P.C., Chen, F., and Yuz, J.I. (2019). Identification of continuous-time models with slowly time-varying parameters. *Control Engineering Practice*, 93, 104165.

Piga, D. (2020). Finite-horizon integration for continuous-time identification: bias analysis and application to variable stiffness actuators. *International Journal of Control*, 93(10), 2378–2391.

Piga, D., Cox, P., Tóth, R., and Laurain, V. (2015). LPV system identification under noise corrupted scheduling and output signal observations. *Automatica*, 53, 329–338.

Quarteroni, A., Sacco, R., and Saleri, F. (2010). *Numerical mathematics*, volume 37. Springer Science & Business Media.

Scherer, C.W. (1996). Mixed $\mathcal{H}_2/\mathcal{H}_\infty$ control for time-varying and linear parametrically-varying systems. *International Journal of Robust and Nonlinear Control*, 6(9-10), 929–952.

Verdult, V. and Verhaegen, M. (2002). Subspace identification of multivariable linear parameter-varying systems. *Automatica*, 38(5), 805–814.

Verdult, V. and Verhaegen, M. (2005). Kernel methods for subspace identification of multivariable LPV and bilinear systems. *Automatica*, 41(9), 1557–1565.

Zhou, K. and Doyle, J.C. (1998). *Essentials of Robust Control*. Prentice-Hall.