# Predictor-Based Tensor Regression (PBTR) for LPV subspace identification

Günes, B.; van Wingerden, Jan Willem; Verhaegen, Michel

**DOI**
10.1016/j.automatica.2017.01.039

**Publication date**
2017

**Document Version**
Accepted author manuscript

**Published in**
Automatica

Link to formal publication Automatica (Elsevier): http://doi.org/10.1016/j.automatica.2017.01.039

# Predictor-Based Tensor Regression (PBTR) for LPV subspace identification ⋆

Bilal Gunes ⋆⋆, Jan-Willem van Wingerden, Michel Verhaegen

*Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628CD Delft, The Netherlands*

**Abstract**

The major bottleneck in state-of-the-art Linear Parameter Varying (LPV) subspace methods is the curse-of-dimensionality during the first regression step. In this paper, the origin of the curse-of-dimensionality is pinpointed and subsequently a novel method is proposed which does not suffer from this bottleneck. The problem is related to the LPV sub-Markov parameters. These have inherent structure and are dependent on each other. But state-of-the-art LPV subspace methods parametrize the LPV sub-Markov parameters independently. This means the inherent structure is not preserved in the parametrization. In turn this leads to a superfluous parametrization with the curse-of-dimensionality. The solution lies in using parametrizations which preserve the inherent structure sufficiently to avoid the curse-of-dimensionality. In this paper a novel method based on tensor regression is proposed. This novel method is named the Predictor-Based Tensor Regression method (PBTR). This method preserves the inherent structure sufficiently to avoid the curse-of-dimensionality. Simulation results show that PBTR has superior performance with respect to both state-of-the-art LPV subspace techniques and also non-convex techniques.

*Key words:* Identification; Subspace Methods; Closed-loop identification; LPV systems; Tensor regression.

## 1 Introduction

Identification problems can be seen as inverse problems. Given some observations and a model structure, they try to infer the values of the parameters characterizing the system. Better results can be obtained both by better observations and richer model structures. One way to obtain richer model structures, is to incorporate more structure of the underlying problem. For some systems this can be achieved by starting to use Linear Parameter Varying (LPV) model structures [11]. In this paper, we develop novel methods for Linear Parameter Varying (LPV) subspace identification.

LPV systems are a very useful subclass of non-linear systems. They are time-varying systems, but their dependence on time is strictly through a scheduling sequence. This system description is very useful for applications for which the scheduling sequence is known. Some application examples are wind turbines [3] , [9], aircraft applications [1], batteries [25] and compressors [11]. Unlike descriptions of systems that are completely non-linear, there are control methodologies available for LPV systems which can guarantee stability and performance in the face of uncertainties [26]. These control methodologies of course require models of the system, which can be obtained from first principles approaches or from identification.

Our focus is on the development of novel LPV identification methods. More specifically, we allow for arbitrarily varying scheduling sequence. This class of systems also encompasses bilinear systems, where the scheduling sequence equals the inputs. Models can be obtained from experimental data by using identification methods. LPV Identification can be divided into global and local approaches. Global approaches perform only one identification experiment, while local approaches perform several experiments at fixed scheduling parameters and then interpolate. Therefore, they perform differently depending on the application [6] , [22] , [27]. In this paper only global approaches will be discussed. There are two major approaches to (global) LPV identification: the subspace approach and the Prediction Error (PE) approach. Both have received considerable attention in literature [2] , [29] , [31]. The advantage of subspace methods is that they produce state-space models which can

⋆ This paper was not presented at any IFAC meeting.

⋆⋆Corresponding author. Tel. +31 0 15 27 85623. Fax +31 15 278 6679.

*Email addresses:* b.gunes@tudelft.nl (Bilal Gunes), j.w.vanwingerden@tudelft.nl (Jan-Willem van Wingerden), m.verhaegen@tudelft.nl (Michel Verhaegen).

be directly used by the mainstream LPV control design methodologies [26]. This is advantageous, because transforming between input-output and state-space models in the LPV setting is non-trivial [29]. Another advantage is that subspace methods can extend naturally to Multiple Input Multiple Output (MIMO) and closed-loop systems. But they also have a major disadvantage: they suffer from the curse-of-dimensionality and yield unwieldy parameter counts [31]. There are several solutions proposed in literature. Some solutions are based on regularization, such as Tikhonov or Nuclear Norm regularization [31] , [10]. Some other solutions are tailored towards scheduling sequences which are periodic [8], white noise [7] or piecewise constant [30]. However these solutions either only partially alleviate this bottleneck or only work for specific cases.

In this paper the origin of the curse-of-dimensionality of LPV subspace methods is pinpointed. It is shown that the curse-of-dimensionality appears when structure of the LPV sub-Markov parameters is not preserved in the parametrization. More specifically, not all the structure of the LPV sub-Markov parameters need to be preserved. This will be made clearly visible by reformulating the LPV data equation using tensors. Such a reformulation will be presented using the inner product of a structured LPV sub-Markov parameter tensor and a corresponding data tensor. Based on this insight, a novel method based on *tensor regression* [34] , [24] , [16] is proposed. Tensor regression is generally used in order to deal with curse-of-dimensionality, such as Magnetic Resonance Imaging (MRI) data [34]. Tensor arise naturally in several more applications such as facial recognition [32] and gait recognition [23], and preserving that structure can be highly beneficial [28]. The novel method preserves structure just sufficiently to avoid the curse-of-dimensionality. This method is named the Predictor-Based Tensor Regression method (PBTR). Simulation results show that this method has higher performance than both state-of-the-art (LPV subspace) methods and also other non-convex methods in the sense of variance accounted for.

In previous work, we presented variants of PBTR for both for LTI [14] , [15] and LPV [13] systems. The novel LPV parametrization presented here, has been presented before in [13] using unnecessarily complicated matrices. In this paper we now present everything explicitly using tensor forms in order to greatly improve clarity on the true system, the parametrizations of several methods and the cause of the curse-of-dimensionality. These tensor forms are the inherent form for tensor regression. This also allows for a clear discussion of the design choices made with PBTR. Furthermore, we compare PBTR with existing non-convex methods, and present simulation results which show that PBTR can outperforms them in terms of variance. We do remark that the formal proof to generalize these simulation results remains an open issue. This paper provides a complete and concise investigation of tensor regression for LPV subspace identification.

The outline of this paper is as follows. The basics of LPV subspace identification are discussed in the next section. Afterwards in Section 3, PBTR is presented together with its motivations. Simulations results are presented in Section 4. Finally the conclusions are presented.

## 2 LPV subspace identification

In this section, LPV subspace identification is reviewed. The focus will be on the work of [31] and [33].

An LPV system can be described by a discrete LPV state-space equation:

$$x_{k+1} = A(\mu_k)x_k + B(\mu_k)u_k + w_k \tag{1a}$$
$$y_k = C(\mu_k)x_k + D(\mu_k)u_k + v_k, \tag{1b}$$

where $x \in \mathcal{R}^n$, $u \in \mathcal{R}^r$ and $y \in \mathcal{R}^l$ are the state, input and output vector variables. This description takes into account both process noise $w$ and measurement noise $v$. The subscript $k$ indicates the sample number. The matrices $A$, $B$, $C$ and $D$ are the appropriately dimensioned state-space matrices. The scheduling sequence $\mu_k$ is time-varying and affects the state-space matrices. We assume that the relation of the scheduling sequence to the state-space matrices is affine:

$$A(\mu_k) = \sum_{i=1}^{m} \mu_k^{(i)} A^{(i)}, \tag{2}$$

and similarly for the other state-space matrices. The scalar $\mu_k^{(i)}$ is defined as the $k$-th sample of the $i$-th scheduling parameter, and $\mu_k^{(1)} = 1$. Additionally we assume in this paper that:

**Assumption 2.1** *The scheduling sequence $\mu_k$ is known.*

For presentation reasons we also restrict ourselves to LPV systems whose output equations are independent of the scheduling sequence. We also omit $D$ for presentation purposes. The extension to include $D$ is straightforward.

For identification purposes, the innovation representation is commonly used [21]. This representation uses the innovation term $e$ to describe the system. For our LPV system, the resulting expression becomes:

$$x_{k+1} = \sum_{i=1}^{m} \mu_k^{(i)} \left(A^{(i)}x_k + B^{(i)}u_k + K^{(i)}e_k\right) \tag{3a}$$
$$y_k = Cx_k + e_k \tag{3b}$$

This representation uses the properties of the Kalman filter.

If the system is closed-loop, then the inputs and noise are correlated due to the feedback. This causes open-loop identification methods to produce biased estimates. The state-of-the-art (LPV subspace) method presented in this section is a closed-loop method, and deals with the correlation between the input and noise by using a predictor-based representation of (3):

$$\hat{x}_{k+1} = \sum_{i=1}^{m} \mu_k^{(i)} \big( \tilde{A}^{(i)} x_k + \bar{B}^{(i)} \begin{bmatrix} u_k \\ y_k \end{bmatrix} \big) \tag{4a}$$

$$y_k = C x_k + e_k, \tag{4b}$$

where $\tilde{A}^{(i)} = A^{(i)} - K^{(i)} C$ and $\bar{B}^{(i)} = [B^{(i)}, K^{(i)}]$. This representation has two nice properties. First, notice that now the innovation only appears at the output equation. Second, the equations now describe the observer error states and use the corresponding observer error dynamics $\tilde{A}^{(i)} = A^{(i)} - K^{(i)} C$. The observer error dynamics can be assumed to be uniformly exponentially stable [31] , [5] , [17], and hence the influence of the states at time $k$ will decay with time. This can be exploited. It appears that if the LPV description (4) is (uniformly) exponentially stable, then it can be approximated arbitrarily well under the assumption that the effect of an initial state is exactly zero after some $p$ time steps [18]. In other words: the current state can be arbitrarily well approximated by using the $p$ past inputs and outputs without any (initial) states:

$$\hat{x}_{k+p} \approx \mathcal{K}^p \mathcal{Z}_{k+p}, \tag{5}$$

where $\mathcal{K}^p$ contains the (LPV sub-Markov) parameters and $\mathcal{Z}_k$ the effective (past input and output) data. In this factorization [31], the scheduling sequence is absorbed into $\mathcal{Z}$ and $\mathcal{K}^p$ is independent of the scheduling sequence. These two matrices will be defined explicitly later in this section. The output equation follows directly:

$$y_{k+p} \approx C \mathcal{K}^p \mathcal{Z}_{k+p} + e_{k+p}, \tag{6}$$

which is very useful for the first identification step because it directly allows for linear regression. Next, for completeness we present the parameter matrix $C\mathcal{K}^p$ and effective data matrix $\mathcal{Z}_k$.

The matrix $C\mathcal{K}^p$ contains the sub-Markov parameters and is independent of the scheduling sequence. Recall that the scheduling sequence is absorbed into $\mathcal{Z}_k$. The matrix $C\mathcal{K}^p$ is a function of the predictor-based state-space matrices:

$$C\mathcal{K}^p = [\mathcal{L}_p, \ldots, \mathcal{L}_1], \tag{7}$$

where $\mathcal{L}_j$ contains all possible routes from inputs to outputs of length $(j+1)$:

$$\mathcal{C}_1 = [\bar{B}^{(1)}, \ldots, \bar{B}^{(m)}] \tag{8a}$$

$$\mathcal{C}_2 = [\tilde{A}^{(1)} \mathcal{C}_1, \ldots, \tilde{A}^{(m)} \mathcal{C}_1] \tag{8b}$$

$$\mathcal{C}_{i+1} = [\tilde{A}^{(1)} \mathcal{C}_i, \ldots, \tilde{A}^{(m)} \mathcal{C}_i] \tag{8c}$$

$$\mathcal{L}_j = C \mathcal{C}_j \tag{8d}$$

Notice that this definition is slightly different from [31] in the sense that we absorb the $C$ matrix into $\mathcal{L}$.

During the first regression step of predictor-based methods, the matrix $C\mathcal{K}^p$ has to be estimated. But unlike in the LTI case, $C\mathcal{K}^p$ now has a very large number of elements:

$$q = l(l+r) \sum_{j=1}^{p} m^j \tag{9}$$

This creates a problem for linear regression because linear regression uses as much parameters as there are elements in $C\mathcal{K}^p$, namely $q$. More specifically, state-of-the-art LPV subspace methods suffer from the *curse-of-dimensionality*:

**Definition 2.1** *Identification methods suffer from the curse-of-dimensionality if their number of parameters scales exponentially with the past window $p$.*

The main contribution of this paper is a novel method which does not suffer from the curse-of-dimensionality and has good numeric properties.

The effective data matrix $\mathcal{Z}_k$ is:

$$\mathcal{Z}_k = N_{k-p}^p Z_k, \tag{10}$$

where $N_{k-p}^p$ contains the scheduling sequence and $Z_k$ the input-output data relevant to $y_k$. The matrix $N_k^p$ is:

$$N_k^p = \begin{bmatrix} P_{p|k} & 0 & \cdots & 0 \\ 0 & P_{p-1|k+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & P_{1|k+p-1} \end{bmatrix}, \tag{11a}$$

$$P_{p|k} = \mu_{k+p-1} \otimes \cdots \otimes \mu_k \otimes I_{r+l}, \tag{11b}$$

where $\otimes$ is defined as the Kronecker product [4]. The matrix $Z_k$ is:

$$Z_k = \begin{bmatrix} z_{k-p} \\ \vdots \\ z_{k-1} \end{bmatrix}, z_k = \begin{bmatrix} u_k \\ y_k \end{bmatrix} \tag{12}$$

The general estimation procedure is as follows. First, the matrix $C\mathcal{K}^p$ is estimated using the data-equation (6). Afterwards the estimate of $C\mathcal{K}^p$ is used in order to choose a model order and obtain an estimate of the state sequence. Together with that estimate, the state-space matrices can be readily estimated [31].

The state-of-the-art (LPV subspace) method [31] follows these same steps, but additionally deploys a dual (or kernel) approach with Tikhonov regularization in the first step. Its regularization parameter is chosen using Generalized Cross Validation [12]. This reduces computational complexity and improves the quality of the estimate in most cases.

Alternatively, non-convex methods exist such as [20] and [33]. These methods directly parametrize the state-space matrices (3). The resulting parametrizations are polynomial and very sensitive to local minima. In order to somewhat ease this issue, most of these methods assume $K = 0$. In this paper PBTR is compared among others with the output-error method of [33].

In the next section, the novel representation and PBTR are presented.

## 3 Predictor-based Tensor Regression

The novel method, Predictor-based Tensor Regression, is presented in this section. First some general (tensor regression) expressions are presented. Afterwards we show that the LPV subspace identification problem contains structure that can be exploited. This can be done using tensor regression in order to avoid the curse-of-dimensionality. We show how the LPV sub-Markov parameters indeed form a parameter tensor. Then the parametrization of PBTR and its algorithm are presented.

### *3.1 General tensor regression expressions*

We first present some general (tensor regression) expressions.

Define $[M]_{i,j}$ as the entry of $M$ at row $i$ and column $j$. Let $[M]_{:,j}$ and $[M]_{i,:}$ respectively be a column and row vector. For a two-by-two matrix $M$:

$$M = \begin{bmatrix} [M]_{1,1} & [M]_{1,2} \\ [M]_{2,1} & [M]_{2,2} \end{bmatrix} = \begin{bmatrix} [M]_{1,:} \\ [M]_{2,:} \end{bmatrix} = \begin{bmatrix} [M]_{:,1} & [M]_{:,2} \end{bmatrix} \tag{13}$$

For both row and column vectors, define $[v]_i$ as the $i$-th entry of $v$.

We define an operator to form tensors from a set of vectors, like in [34]. Let $\beta_d$ represent a vector with size $d_i$-by-1. Then the outer product $\beta_1 \circ \beta_2 \circ \cdots \circ \beta_D$ is a tensor of size $\mathcal{R}^{d_1,d_2,\ldots,d_D}$ with entries:

$$[\beta_1 \circ \cdots \circ \beta_D]_{i_1,\ldots,i_D} = \prod_{d=1}^{D} [\beta_d]_{i_d} \tag{14}$$

A tensor can be represented in several forms. For the use of tensor regression, the most natural form is the rank-$R$ decomposition [34] (or CANonical DEComposition/PARAllel FACtors in psychometrics [19]). This decomposition decomposes a tensor into the sum of exactly $R$ outer products. For example consider a tensor $\mathcal{T}$ with $D$ dimensions. This tensor can then be rank-$R$ decomposed into:

$$\mathcal{T} = \sum_{r=1}^{R} \beta_1^{(r)} \circ \beta_2^{(r)} \circ \cdots \circ \beta_D^{(r)}, \tag{15}$$

where the $\beta_*^{(*)}$ represent vectors and $\circ$ is the outer product [34] which turns vectors into a tensor. The subscript indicates the vector group, and the superscript indicates which individual vector to take. There are in total $R$ times $D$ vectors.

In the succeeding subsections, the structure of the LPV subspace problem and PBTR are presented explicitly and in tensor form.

### *3.2 The highly-structured parameter tensor*

In this subsection we use the LPV sub-Markov parameters to build a highly-structured parameter tensor. We present this tensor in rank-$R$ decomposition form, such that tensor regression can be directly applied. This will allow for a clear view on the parametrization of PBTR and why it is sufficient to avoid the curse-of-dimensionality.

We present the structure of the LPV sub-Markov parameters explicitly in tensor form. For this purpose, consider the LPV sub-Markov parameters:

$$C\mathcal{K}^p = [\mathcal{L}_p, \mathcal{L}_{p-1}, \ldots, \mathcal{L}_1] \tag{16}$$

For presentation purposes, consider the part:

$$\mathcal{L}_2 = C[\tilde{A}^{(1)}\bar{B}^{(1)}, \tilde{A}^{(1)}\bar{B}^{(2)}, \tilde{A}^{(2)}\bar{B}^{(1)}, \cdots \tilde{A}^{(m)}\bar{B}^{(m)}]$$

Notice that there is structure present. This structure becomes more apparent if we re-organize the parameters to:

$$\tilde{\mathcal{L}}_2 = \begin{bmatrix} C\tilde{A}^{(1)}\bar{B}^{(1)} & C\tilde{A}^{(1)}\bar{B}^{(2)} & \ldots & C\tilde{A}^{(1)}\bar{B}^{(m)} \\ \vdots & \vdots & \ddots & \vdots \\ C\tilde{A}^{(m)}\bar{B}^{(1)} & C\tilde{A}^{(m)}\bar{B}^{(2)} & \ldots & C\tilde{A}^{(m)}\bar{B}^{(m)} \end{bmatrix},$$

or even more clear:

$$\tilde{\mathcal{L}}_2 = \begin{bmatrix} C\tilde{A}^{(1)} \\ C\tilde{A}^{(2)} \\ \vdots \\ C\tilde{A}^{(m)} \end{bmatrix} [\bar{B}^{(1)}, \bar{B}^{(2)}, \ldots, \bar{B}^{(m)}], \tag{17}$$

where the reorganization changed the size from $l$-by-$(l+r)m^2$ to $lm$-by-$(l+r)m$.

However, this description using matrix products is not directly suitable for tensor regression. For that reason we now move towards a rank-$R$ decomposition form, which is a sum of outer products of vectors. First we solve this problem for one output $C_1$ and one effective input $\bar{B}_1$. Equation (17) then becomes:

$$\bar{\mathcal{L}}_2^{1,1} = \begin{bmatrix} C_1\tilde{A}^{(1)} \\ C_1\tilde{A}^{(2)} \\ \vdots \\ C_1\tilde{A}^{(m)} \end{bmatrix} [\bar{B}_1^{(1)}, \ldots, \bar{B}_1^{(m)}], \tag{18}$$

Then we can rewrite the equation above by splitting out every single summation inside the matrix multiplications:

$$\bar{\mathcal{L}}_2^{1,1} = \sum_{i=1}^{n}\sum_{j=1}^{n} \begin{bmatrix} [C_1]_i[\tilde{A}^{(1)}]_{i,j} \\ [C_1]_i[\tilde{A}^{(2)}]_{i,j} \\ \vdots \\ [C_1]_i[\tilde{A}^{(m)}]_{i,j} \end{bmatrix} [[\bar{B}_1^{(1)}]_j, \ldots, [\bar{B}_1^{(m)}]_j],$$

or using the fact that $[C_1]_i$ is scalar:

$$\bar{\mathcal{L}}_2^{1,1} = \sum_{i=1}^{n}\sum_{j=1}^{n} [C_1]_i \begin{bmatrix} [\tilde{A}^{(1)}]_{i,j} \\ [\tilde{A}^{(2)}]_{i,j} \\ \vdots \\ [\tilde{A}^{(m)}]_{i,j} \end{bmatrix} [[\bar{B}_1^{(1)}]_j, \ldots, [\bar{B}_1^{(m)}]_j]$$

Now there are just products of vectors instead of products of matrices. Hence, we can set up the following rank-$R$ decomposition:

$$\bar{\mathcal{L}}_2^{1,1} = \sum_{i=1}^{n}\sum_{j=1}^{n} v_1^{(i)} \circ v_2^{(i,j)} \circ v_3^{(j)} \tag{21}$$

where we have:

$$\tilde{v}_1^{(i)} = [C]_{1,i}, v_2^{(i,j)} = \begin{bmatrix} [\tilde{A}^{(1)}]_{i,j} \\ \vdots \\ [\tilde{A}^{(m)}]_{i,j} \end{bmatrix}, \tilde{v}_3^{(j)} = \begin{bmatrix} [\bar{B}_1^{(1)}]_j \\ \vdots \\ [\bar{B}_1^{(m)}]_j \end{bmatrix},$$

where tildes have been used to indicate definitions which are only valid for the single output and single effective input case (18).

Now we can extend to the case of a full $C$ matrix. This is done by redefining $v_1^{(i)}$. While previously it was a scalar, we now turn it into a vector. The resulting outer product becomes three-dimensional. Simply define:

$$v_1^{(i)} = [C]_{:,i} \tag{22}$$

Now we can extend to the case of a full $\bar{B}$ matrix as well. This is more involved, because we already had a vector group devoted to $\bar{B}_1$. The solution is to devote two vector groups to $\bar{B}$. First, we define $v_3$ to incorporate the entire $\bar{B}$ matrix. This requires another superscript index $\bar{\kappa}$:

$$v_3^{(j,\bar{\kappa})} = \begin{bmatrix} [\bar{B}^{(1)}]_{j,\bar{\kappa}} \\ \vdots \\ [\bar{B}^{(m)}]_{j,\bar{\kappa}} \end{bmatrix} \tag{23}$$

Notice that previously $\bar{\kappa}$ was fixed at one. Also notice that $j$ cycles the states, and $\bar{\kappa}$ cycles the width of $\bar{B}^{(\#)}$. We want to map this added complexity on a new dimension. For that purpose, we define $v_4$ as:

$$v_4^{(\bar{\kappa})} = \begin{bmatrix} \bar{0}_{\bar{\kappa}-1} \\ 1 \\ \bar{0}_{(l+r)-\bar{\kappa}} \end{bmatrix} \tag{24}$$

If we then use:

$$\bar{\mathcal{L}}_2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,j)} \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})}, \tag{25}$$

we obtain a $\mathcal{R}^{l,m,m,l+r}$ tensor where the last dimension cycles over the width of $\bar{B}^{(\#)}$. That is, $[\bar{\mathcal{L}}_2]_{:,:,:,i}$ corresponds to the $i$-th column of all $\bar{B}^{(\#)}$. Notice that $\bar{\mathcal{L}}_2$ and $\mathcal{L}_2$ have the same entries, but in different positions. Basically, we reorganized $\mathcal{L}_2$ into $\bar{\mathcal{L}}_2$ in order to make it suitable for rank-$R$ decomposition and tensor regression.

The resulting expression (25) is valid for any LPV system, but describes only the LPV sub-Markov parameters in $\mathcal{L}_2$. Hence, we need to extend this formulation

to capture all the LPV sub-Markov parameters. Before searching for an expression containing all the LPV sub-Markov parameters, we first investigate an expression containing the LPV sub-Markov parameters of $\mathcal{L}_p$. Its added complexity is the appearance of multiple $\tilde{A}$ within every product. We accommodate this complexity by using multiple $v_2$, one for each appearance of $\tilde{A}$ within every product:

$$\begin{aligned}\bar{\mathcal{L}}_2 &= \sum_{i,j}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,j)} \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})} \\ \bar{\mathcal{L}}_p &= \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,\delta_1)} \circ v_2^{(\delta_1,\delta_2)} \circ \ldots \\ &\quad \circ v_2^{(\delta_{p-2},j)} \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})},\end{aligned}$$

where $\bar{\mathcal{L}}_p$ is a $\mathcal{R}^{l,m,m,\ldots,m,l+r}$ tensor. Notice that the superscripts continue to form a chain link $(i,\delta_1)$, $(\delta_1,\delta_2)$ etc. This is a result of the underlying matrix multiplications, as was explained for (20).

The following step is to define a single tensor which contains all the LPV sub-Markov parameters. This requires the stacking of all $\bar{\mathcal{L}}_*$. These will be stacked over a new dimension, which runs from 1 to $p$ and has index $\bar{p}$. So $\bar{\mathcal{L}}_1$ will be at the first index and so forth. However, the tensors $\bar{\mathcal{L}}_*$ do not have equal size. The solution is to make all these tensors the same size. For this purpose, consider the first product of the first $\mathcal{L}_*$:

$$\begin{aligned}\mathcal{L}_1 &= [C\bar{B}^{(1)}, \ldots] && \text{(27a)}\\ \mathcal{L}_2 &= [C\tilde{A}^{(1)}\bar{B}^{(1)}, \ldots] && \text{(27b)}\\ \mathcal{L}_3 &= [C\tilde{A}^{(1)}\tilde{A}^{(1)}\bar{B}^{(1)}, \ldots] && \text{(27c)}\\ &\vdots\end{aligned}$$

The dimension mismatch of different $\bar{\mathcal{L}}_{\#}$ appears because they have a different number of terms in their products. This can be easily solved by adding identity matrices:

$$\begin{aligned}\mathcal{L}_1 &= [CI_nI_n\bar{B}^{(1)}, \ldots] && \text{(28a)}\\ \mathcal{L}_2 &= [C\tilde{A}^{(1)}I_n\bar{B}^{(1)}, \ldots] && \text{(28b)}\\ \mathcal{L}_3 &= [C\tilde{A}^{(1)}\tilde{A}^{(1)}\bar{B}^{(1)}, \ldots] && \text{(28c)}\\ &\vdots\end{aligned}$$

Now every $\mathcal{L}_*$ has the same number of terms inside their products, and we can redefine:

**Definition 3.1** *The tensors $\bar{\mathcal{L}}_j$ Redefine the tensors $\bar{\mathcal{L}}_j$ as:*

$$\begin{aligned}\bar{\mathcal{L}}_p &= \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,\delta_1)} \circ v_2^{(\delta_1,\delta_2)} \circ \ldots \\ &\quad \circ v_2^{(\delta_{p-2},j)} \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})} \\ \bar{\mathcal{L}}_{p-1} &= \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,\delta_1)} \circ v_2^{(\delta_1,\delta_2)} \circ \ldots \\ &\quad \circ [I]_{\delta_{p-2},j}\bar{1}_m \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})} \\ &\vdots \\ \bar{\mathcal{L}}_2 &= \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ v_2^{(i,\delta_1)} \circ [I]_{\delta_1,\delta_1}\bar{1}_m \circ \ldots \\ &\quad \circ [I]_{\delta_{p-2},j}\bar{1}_m \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})} \\ \bar{\mathcal{L}}_1 &= \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)} \circ [I]_{i,\delta_1}\bar{1}_m \circ [I]_{\delta_1,\delta_2}\bar{1}_m \circ \ldots \\ &\quad \circ [I]_{\delta_{p-2},j}\bar{1}_m \circ v_3^{(j,\bar{\kappa})} \circ v_4^{(\bar{\kappa})},\end{aligned}$$

*such that all $\bar{\mathcal{L}}_i$ have the same dimensions which is $\mathcal{R}^{l,m,m,\ldots,m,m,l+r}$. The tensor $\bar{\mathcal{L}}_i$ contains all entries of $\mathcal{L}_i$ and their transformation back and forth is one-to-one. Basically the smaller tensors are padded with themselves until they have the appropriate size.*

Now we can safely stack $\bar{\mathcal{L}}_*$ over a new dimension to obtain the parameter tensor. Define:

**Definition 3.2** *Define the tensor $\bar{\mathcal{L}} \in \mathcal{R}^{l,m,\ldots,m,l+r,p}$ as:*

$$\bar{\mathcal{L}} = \sum_{\bar{p}}^{p} \bar{\mathcal{L}}_{\bar{p}} \circ \begin{bmatrix} \bar{0}_{\bar{p}-1} \\ 1 \\ \bar{0}_{p-\bar{p}} \end{bmatrix}, \qquad (30)$$

*using Definition 3.1,* (22), (22), (23) *and* (24)*. Notice that $\bar{\mathcal{L}}$ and $C\mathcal{K}^p$ have the same entries (with some duplicates), but in different positions. Basically, we reorganized $C\mathcal{K}^p$ into $\bar{\mathcal{L}}$ in order to make it suitable for rank-R decomposition and tensor regression.*

We have finished deriving the highly-structured parameter tensor $\bar{\mathcal{L}}$. Its expression is different from the parameter matrix $C\mathcal{K}^p$, but contains the same LPV sub-Markov parameters. The new expression will be useful for clarifying which pieces of structure are discarded in state-of-the-art methods and why the curse-of-dimensionality appears. This relates strongly to the chosen parametrizations. In the next section, we present the parametrizations of both the state-of-the-art method and PBTR and investigate the resulting parameter counts.

### 3.3 Parametrizations

In this subsection, we present the parametrization of PBTR and compare it to the parametrization of state-of-the-art methods. We show which pieces of structure are ignored where, and what causes the curse-of-dimensionality to appear.

Consider the LPV predictor-based data equation:

$$y_k \approx C\mathcal{K}^p \mathcal{Z}_k + e_k \tag{31}$$

This equation is parametrized by state-of-the-art LPV subspace methods *element-wise* as:

$$\hat{y}_k(\theta) = [C\mathcal{K}^p](\theta)\mathcal{Z}_k \tag{32}$$

As a result, the parameter count of state-of-the-art LPV subspace methods is equal to the number of entries in $C\mathcal{K}^p$. Because this number scales exponentially with the past window $p$, so these methods suffer from the curse-of-dimensionality.

The PBTR is a tensor regression method, therefore we present a novel rewritten LPV data equation which is more suitable for tensor regression. This data equation uses the inner product of the parameter tensor $\bar{\mathcal{L}}$ (Definition 3.2) and and appropriate data tensor $\bar{\mathcal{Z}}_k$. This appropriate data tensor is a reorganization of $\mathcal{Z}_k$ which matches the reorganization of $C\mathcal{K}^p$ into $\bar{\mathcal{L}}$, where the data corresponding to duplicate parameters are scaled down. This tensor-form LPV data equation is:

$$y_k \approx < \bar{\mathcal{L}}, \bar{\mathcal{Z}}_k > \ + e_k, \tag{33}$$

where the inner product is redefined in order to deal with multiple outputs:

**Definition 3.3** *Consider the inner product of two tensors:* $< \mathcal{T}, \mathcal{U} >$. *Normally this requires* $\mathcal{T}$ *and* $\mathcal{U}$ *to have equal size. But in order to deal with multiple outputs, we extend the definition of this operator as follows. Let* $\mathcal{T} \in \mathcal{R}^{l,d_1,\ldots,d_N}$ *and* $\mathcal{U} \in \mathcal{R}^{d_1,\ldots,d_N}$. *Then their inner product exists and equals:*

$$< \mathcal{T}, \mathcal{U} >= \begin{bmatrix} < \mathcal{T}_1, \mathcal{U} > \\ \vdots \\ < \mathcal{T}_l, \mathcal{U} > \end{bmatrix}, \tag{34}$$

*where* $\mathcal{T}_i \in \mathcal{R}^{d_1,\ldots,d_N}$ *is an appropriate part of* $\mathcal{T}$.

The parametrization of PBTR is a tensor regression parametrization, and as a result multi-linear in nature. Additionally, the parametrization revolves around the predictor-based state-space matrices. It can then be written as:

$$\bar{\mathcal{L}}(\theta) = \sum_{\bar{p}}^{p} \bar{\mathcal{L}}_{\bar{p}}(\theta) \circ \begin{bmatrix} \bar{0}_{\bar{p}-1} \\ 1 \\ \bar{0}_{p-\bar{p}} \end{bmatrix} \tag{35a}$$

$$\bar{\mathcal{L}}_p(\theta) = \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)}(\theta_1) \circ v_2^{(i,\delta_1)}(\theta_{2,1}) \circ \ldots \circ v_2^{(\delta_{p-2},j)}(\theta_{2,p-1}) \circ v_3^{(j,\bar{\kappa})}(\theta_3) \circ v_4^{(\bar{\kappa})} \tag{35b}$$

$$\bar{\mathcal{L}}_{p-1}(\theta) = \sum_{i,j,\delta_1,\ldots,\delta_{p-2}}^{n} \sum_{\bar{\kappa}=1}^{l+r} v_1^{(i)}(\theta_1) \circ v_2^{(i,\delta_1)}(\theta_{2,1}) \circ \ldots \circ [I]_{\delta_{p-2},j} \bar{1}_m \circ v_3^{(j,\bar{\kappa})}(\theta_3) \circ v_4^{(\bar{\kappa})} \tag{35c}$$

where each individual vector is parametrized element-wise. The sizes of the parameter group are as follows. The $\theta_1$ has $ln$ parameters, $\theta_{2,i}$ has $n^2m$ parameters for all $i \in \{1,\ldots,p-1\}$, and $\theta_3$ has $(l+r)nm$ parameters. Notice that this parametrization is not a direct parametrization in the LPV predictor-based state-space matrices (4), because the $\tilde{A}^{(i)}$ are spuriously parametrized in order to obtain a multi-linear parametrization. Notice that PBTR exploits more structure than the state-of-the-art LPV subspace methods do and does not suffer from the curse-of-dimensionality.

It is also possible to use the polynomial non-convex method [33], which enforces available structure by directly parametrizing the regular state-space matrices. Notice that this method does not have a second estimation step. This method also does not suffer from the curse-of-dimensionality, and the surplus enforced structure slightly further reduces the parameter count. Notice that both PBTR and the polynomial non-convex method have a non-convex parametrization. Therefore they require an initial estimate (including a model order). This initial estimate can be obtained from state-of-the-art (LPV subspace) methods. This places PBTR and the polynomial non-convex method as refinement methods for LPV subspace methods.

The parameter counts are summarized in Table 1. Notice that the parameter counts of the evaluated methods scale differently. The state-of-the-art LPV subspace methods suffer from the curse-of-dimensionality, because their parameter count scales exponentially with $p$. Usage of kernels (dual approaches) changes the parameter count to scale with data instead, but have the disadvantage that they result in ill-conditioned problems. Regularization can only partially solve this problem [31]. Both PBTR and the polynomial method do not suffer from the curse-of-dimensionality.

This concludes the evaluation of the parameter counts

Table 1
**Comparison of the parameter counts for the (first) estimation step**

| Method | Parameter count |
|---|---|
| LPV-PBSID$_{opt}$ (primal) | $l(r+l)\sum_{j=1}^{p} m^j$ |
| LPV-PBSID$_{opt}$ (dual) | $l(N-2p)$ |
| PBTR (with free parametrization) | $nl + n(l+r)m + n^2m(p-1)$ |
| Polynomial method | $nl + nrm + n^2m$ |

of PBTR and state-of-the-art methods. In the next subsection, the full PBTR algorithm is presented.

### *3.4 Algorithm*

**Algorithm 3.1** *The PBTR*

*Define the cost function of PBTR as:*

$$V_N(\theta) = \sum_{k=1}^{N} (y_k - \hat{y}_k(\theta))^T (y_k - \hat{y}_k(\theta)), \tag{36a}$$

$$\hat{y}_k(\theta) = < \bar{\mathcal{L}}(\theta), \bar{\mathcal{Z}}_k > \tag{36b}$$

*where $\hat{y}_k$ is the model output, $\bar{\mathcal{L}}(\theta)$ is defined in* (35) *and $\bar{\mathcal{Z}}_k$ is defined in the previous subsection. This is a multi-linear parametrization in the predictor-based state-space matrices with additional structure. It is possible to obtain a consistent estimate of all $\theta$ using multi-linear optimization [34], for cases where equation* (6) *is not an approximation but an equality. This can be done using Alternating Least Squares [34] or MATLAB's 'fmin' command. Notice that the only indeterminacy is modulo global state-coordinate transformation, which is common. After obtaining an estimate of $\bar{\mathcal{L}}$, an estimate of $C\mathcal{K}^p$ can be directly and one-to-one constructed. The succeeding steps follow the same methodology as other predictor-based subspace methods as presented in Section 2.*

This concludes the section on PBTR. In the next section, the simulation results are presented.

## 4 Simulations

In this section simulation results are presented in order to compare PBTR with state-of-the-art methods for several cases in terms of bias, variance and parameter count.

### *4.1 Simulation settings*

In this subsection, the general simulation settings and some definitions are presented.

The results presented in this paper are based on 100 Monte Carlo simulations. For every Monte Carlo simulation different realizations of both the input and the innovation vector were used. During every Monte Carlo simulation, first estimates of the state-space matrices are obtained from both the unregularized and the regularized variant of the LPV subspace method [31]. Then the estimate of the regularized variant is used as an initial estimate for the non-convex methods: PBTR and the method of [33]. It is worth noting that we do not consider the prediction error variant of the method of [33], because the authors indicated that that variant performs badly. The variant that we do use, fixes the parameters of $K$ to zero in order to somewhat relieve its involved parametrization. All methods are provided with the system order $n$, which is assumed to be known, and the information that $D = 0$ and $C$ is LTI.

We also present the following settings which are the same for every case. The matrix $K^{(i)}$ for $i = \{1, \ldots, m\}$ is obtained from the Discrete Algebraic Ricatti Equation (DARE) with $A^{(i)}$, $C$ and identity covariance of the concatenated process and measurement noise. Every signal of the input vector $u_k$ and innovation vector $e_k$ is white noise. The data size $N$ is chosen as 200, and both the past window $p$ and the future window $f$ are 6.

The quality of the estimates is evaluated by investigating the Variance Accounted For (VAF) on a validation data set *different* from the one used for identification, in the sense that different realizations of both the input and the innovation vector are used. The VAF for single-output systems is defined as [31]:

$$VAF(\bar{y}_k, \hat{y}_k) = max\left\{1 - \frac{var(\bar{y}_k - \hat{y}_k)}{var(\bar{y}_k)}, 0\right\} * 100\%$$

Notice that the noise-free simulated output of the system can be used when evaluating the VAF, because the data is obtained from simulations. This allows the VAF to reach 100% when the model is equal to the true system modulo global state-coordinate transformations, such that the analysis becomes more clear. The noise-free (simulated) output of the system is denoted as $\bar{y}_k$. Similarly, $\hat{y}_k$ is used for the noise-free (simulated) model output. The $var(*)$ operator denotes the variance.

In the case that a non-convex method produces an estimate with an identification data VAF less than half the identification data VAF of the initial estimate, the refined estimate is rejected and substituted directly by the initial estimate. This is possible, because the identification data is available during estimation. Notice that this does not prevent local minima, but merely serves to reject poor optimization results.

The cases and their results are presented in the following subsection. A parameter count investigation is per-

formed in the last subsection.

### *4.2 Simulation results Case 1*

This case uses the following LPV state-space system (3):

$$[A^{(1)}, A^{(2)}] = \left[\begin{array}{cc|cc} \frac{4}{15} & \frac{1}{15} & \frac{3}{20} & -\frac{1}{60} \\ -\frac{1}{6} & \frac{1}{30} & -\frac{1}{60} & \frac{3}{20} \end{array}\right],$$
$$[B^{(1)}, B^{(2)}] = \left[\begin{array}{c|c} 1 & 0.2 \\ 0 & 0.2 \end{array}\right], C = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

and the Signal-to-Noise Ratio (SNR) is 1. The remaining settings are as described in Subsection 4.1. The system is evaluated at two different affine scheduling sequences with:

$$\mu_k^{(2)} = cos(2\pi k \frac{\Pi}{N})/2 + 0.2,$$

where $\Pi = 20$ for the first and $\Pi = 4$ for the second scheduling sequence.

Table 2
**Mean VAF for different methods for Case 1**

| Scheduling | Method | VAF |
|---|---|---|
| $\Pi = 20$ | LPV-PBSID$_{opt}$ (kernel) | 95.8 |
| | Reg. LPV-PBSID$_{opt}$ (kernel) | 96.6 |
| | PBTR | 98.0 |
| | Polynomial non-convex method | 96.5 |
| $\Pi = 4$ | LPV-PBSID$_{opt}$ (kernel) | 23 |
| | Reg. LPV-PBSID$_{opt}$ (kernel) | 97.0 |
| | PBTR | 98.1 |
| | Polynomial non-convex method | 96.7 |

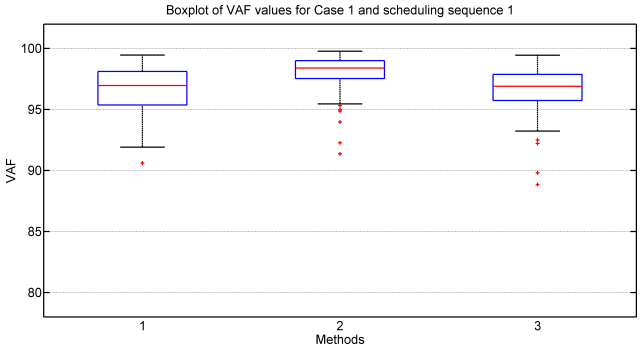


Fig. 1. Boxplots of the VAF results of Case 1 for the evaluated methods at scheduling sequence 1. The methods are: 1. Reg. LPV-PBSID$_{opt}$, 2. PBTR, 3. Polynomial method.

From the results of Table 2 it can be seen that PBTR has superior VAF in comparison to the other methods. The other (polynomial) non-convex method has comparable VAF to state-of-the-art LPV subspace method. Furthermore, the VAF of the unregularized variant of the state-of-the-art LPV subspace method appears to

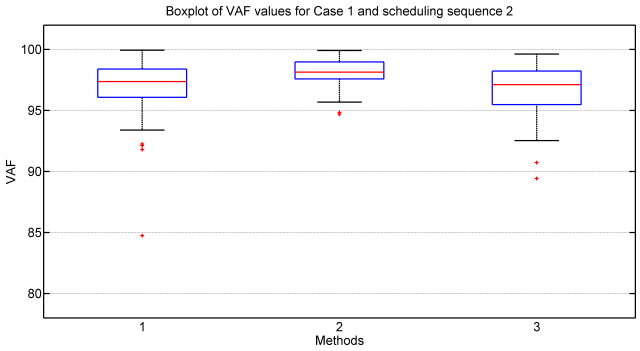


Fig. 2. Boxplots of the VAF results of Case 1 for the evaluated methods at the scheduling sequence 2. The methods are: 1. Reg. LPV-PBSID$_{opt}$, 2. PBTR, 3. Polynomial method.

vary considerably with the scheduling sequence. These results are further supported by Fig. 1 and 2.

### *4.3 Simulation results Case 2*

This case uses the following LPV state-space system (3):

$$[A^{(1)}, A^{(2)}, A^{(3)}] = \left[\begin{array}{cc|cc|cc} \frac{4}{15} & \frac{1}{15} & \frac{3}{20} & -\frac{1}{60} & \frac{29}{405} & \frac{2}{81} \\ -\frac{1}{6} & \frac{1}{30} & -\frac{1}{60} & \frac{3}{20} & \frac{1}{81} & \frac{52}{405} \end{array}\right],$$
$$[B^{(1)}, B^{(2)}, B^{(3)}] = \left[\begin{array}{c|c|c} 1 & 0.2 & 0.2 \\ 0 & 0.2 & -0.2 \end{array}\right], C = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

and the Signal-to-Noise Ratio (SNR) is 2. The remaining settings are as described in Subsection 4.1. The system is evaluated at two different affine scheduling sequences with:

$$\mu_k^{(2)} = cos(2\pi k \frac{\Pi}{N})/2 + 0.2$$
$$\mu_k^{(3)} = cos(2\pi k \frac{\Pi}{2.5N} + 0.5\pi)/3,$$

where $\Pi = 20$ for the first and $\Pi = 4$ for the second scheduling sequence.

Table 3
**Mean VAF for different methods for Case 2**

| Scheduling | Method | VAF |
|---|---|---|
| $\Pi = 20$ | LPV-PBSID$_{opt}$ (kernel) | 81 |
| | Reg. LPV-PBSID$_{opt}$ (kernel) | 97.4 |
| | PBTR | 98.4 |
| | Polynomial non-convex method | 97.6 |
| $\Pi = 4$ | LPV-PBSID$_{opt}$ (kernel) | 7.3 |
| | Reg. LPV-PBSID$_{opt}$ (kernel) | 97.7 |
| | PBTR | 98.4 |
| | Polynomial non-convex method | 97.6 |

From the results of Table 3 it can be seen that PBTR has superior VAF in comparison to the other methods.

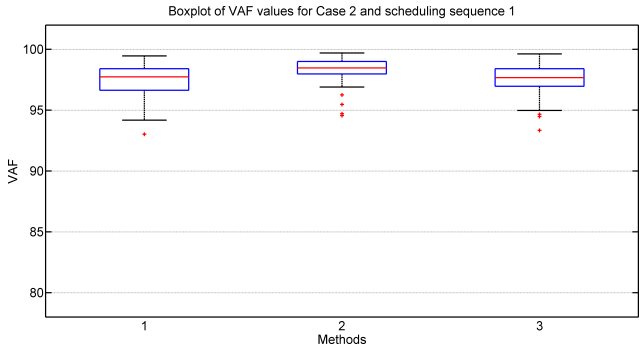


Fig. 3. Boxplots of the VAF results of Case 2 for the evaluated methods at scheduling sequence 1. The methods are: 1. Reg. LPV-PBSID$_{opt}$, 2. PBTR, 3. Polynomial method.

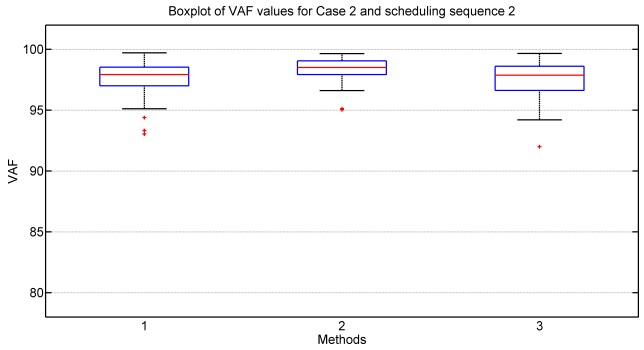


Fig. 4. Boxplots of the VAF results of Case 2 for the evaluated methods at the scheduling sequence 2. The methods are: 1. Reg. LPV-PBSID$_{opt}$, 2. PBTR, 3. Polynomial method.

The other (polynomial) non-convex method has comparable VAF to state-of-the-art LPV subspace method. Furthermore, the VAF of the unregularized variant of the state-of-the-art LPV subspace method appears to vary considerably with the scheduling sequence. These results are further supported by Fig. 3 and 4.

### *4.4 Simulation results Case 3*

This case uses the following LPV state-space system (3):

$$[A^{(1)}, A^{(2)}] = \left[\begin{array}{cccc|cccc} \frac{-1}{300} & \frac{1}{30} & \frac{11}{75} & \frac{8}{75} & \frac{-3}{10} & \frac{1}{6} & \frac{11}{30} & \frac{11}{30} \\ \frac{3}{20} & \frac{1}{20} & \frac{-3}{20} & \frac{-3}{20} & \frac{1}{20} & \frac{7}{60} & \frac{-1}{20} & \frac{-1}{20} \\ \frac{-29}{100} & \frac{1}{10} & \frac{32}{75} & \frac{7}{25} & \frac{-9}{20} & \frac{3}{20} & \frac{31}{60} & \frac{9}{20} \\ \frac{11}{300} & \frac{-1}{60} & \frac{-3}{100} & \frac{23}{300} & \frac{-1}{30} & \frac{1}{30} & \frac{1}{30} & \frac{1}{10} \end{array}\right],$$

$$[B^{(1)}, B^{(2)}] = \left[\begin{array}{c|c} 1 & 0.2 \\ 0 & 0.2 \\ 0 & 0.2 \\ 0 & 0.2 \end{array}\right], C = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}\right],$$

and the Signal-to-Noise Ratio (SNR) is 0.5. The remaining settings are as described in Subsection 4.1. The system is evaluated at the affine scheduling sequence with:

$$\mu_k^{(2)} = cos(2\pi k \frac{20}{N})/2 + 0.2,$$

Table 4
**Mean VAF for different methods for Case 3**

| Method | VAF |
|---|---|
| LPV-PBSID$_{opt}$ (kernel) | 81 |
| Reg. LPV-PBSID$_{opt}$ (kernel) | 85.1 |
| PBTR | 90.8 |
| Polynomial non-convex method | 80.9 |

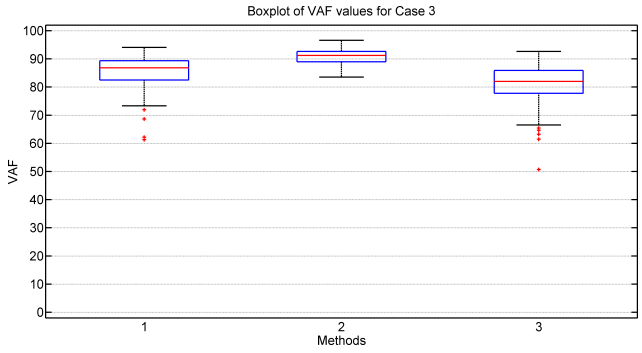


Fig. 5. Boxplots of the VAF results of Case 3 for the evaluated methods. The methods are: 1. Reg. LPV-PBSID$_{opt}$, 2. PBTR, 3. Polynomial method.

From the results of Table 4 it can be seen that PBTR has superior VAF in comparison to the other methods. The other (polynomial) non-convex method however fails to refine the initial estimates supplied by the state-of-the-art LPV subspace method. These results are supported by Fig. 5.

### *4.5 Parameter counts*

The parameter counts of the evaluated methods for Cases 1 and 2 are presented in Table 5. It is visible that the PBTR has a parameter count roughly in between the state-of-the-art LPV subspace methods and the polynomial non-convex method. The PBTR does not suffer from the curse-of-dimensionality, while also having a superior performance in terms of VAF as shown in the previous subsections.

## 5 Conclusions

In this paper a novel method for LPV identification was presented, which is named PBTR. The benefit of PBTR over state-of-the-art LPV subspace methods is that it does not suffer from the curse-of-dimensionality. This was achieved by first pinpointing the origin of the curse-of-dimensionality, which appeared to be the ignoring of inherent structure of the LPV sub-Markov parameters,

Table 5
**Comparison of the parameter counts for the (first) estimation step for some cases**

| Method | Case 1 | Case 2 |
|---|---|---|
| LPV-PBSID$_{opt}$ (primal) | 252 | 2184 |
| LPV-PBSID$_{opt}$ (dual) | 188 | 188 |
| PBTR (with free parametrization) | 50 | 74 |
| Polynomial method | 18 | 26 |

and then using tensor regression to prevent it. This does make PBTR a non-convex method. The difference of PBTR with other non-convex methods is that it uses tensor regression to exploit only the structure necessary to avoid the curse-of-dimensionality. Though the formal proof remains an open issue, simulation results show that PBTR has better performance with respect to state-of-the-art LPV subspace techniques and non-convex techniques by looking at the variance.

## References


[1] Gary J Balas. Linear, parameter-varying control and its application to aerospace systems. In *ICAS Congress Proceedings*, 2002.

[2] Bassam Bamieh and Laura Giarre. Identification of linear parameter varying models. *International journal of robust and nonlinear control*, 12(9):841–853, 2002.

[3] Fernando D Bianchi, Ricardo J Mantz, and Carlos F Christiansen. Gain scheduling control of variable-speed wind energy conversion systems using quasi-LPV models. *Control Engineering Practice*, 13(2):247–255, 2005.

[4] John Brewer. Kronecker products and matrix calculus in system theory. *IEEE Transactions on circuits and systems*, 25(9):772–781, 1978.

[5] Alessandro Chiuso. The role of vector autoregressive modeling in predictor-based subspace identification. *Automatica*, 43(6):1034–1048, 2007.

[6] Jan De Caigny, JF Camino, Bart Paijmans, and Jan Swevers. An application of interpolating gain-scheduling control. In *Proc. 3rd IFAC Symp. Syst., Struct. and Control (SSSC07)*, 2007.

[7] Wouter Favoreel, Bart De Moor, and Peter Van Overschee. Subspace identification of bilinear systems subject to white inputs. *Automatic Control, IEEE Transactions on*, 44(6):1157–1165, 1999.

[8] Federico Felici, Jan-Willem Van Wingerden, and Michel Verhaegen. Subspace identification of mimo lpv systems using a periodic scheduling sequence. *Automatica*, 43(10):1684–1697, 2007.

[9] Pieter MO Gebraad, Jan-Willem van Wingerden, Paul A Fleming, and Alan D Wright. Lpv identification of wind turbine rotor vibrational dynamics using periodic disturbance basis functions. *Control Systems Technology, IEEE Transactions on*, 21(4):1183–1190, 2013.

[10] Pieter MO Gebraad, Jan-Willem van Wingerden, Gijs J van der Veen, and Michel Verhaegen. LPV subspace identification using a novel nuclear norm regularization method. In *American Control Conference (ACC)*, 2011.

[11] Laura Giarré, Dario Bauso, Paola Falugi, and Bassam Bamieh. LPV model identification for gain scheduling control: An application to rotating stall and surge control problem. *Control Engineering Practice*, 14(4):351–361, 2006.

[12] Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[13] Bilal Gunes, Jan-Willem van Wingerden, and Michel Verhaegen. Tensor regression for lpv subspace identification. *Accepted for SYmposium on System IDentification (SYSID)*, 2015.

[14] Bilal Gunes, Jan-Willem van Wingerden, and Michel Verhaegen. Tensor regression for subspace identification. *American Control Conference*, 2015.

[15] Bilal Gunes, Jan-Willem van Wingerden, and Michel Verhaegen. Tensor regression for subspace identification: free parameterizations. *Accepted for SYmposium on System IDentification (SYSID)*, 2015.

[16] Weiwei Guo, Irene Kotsia, and Ioannis Patras. Tensor learning for regression. *Image Processing, IEEE Transactions on*, 21(2):816–827, 2012.

[17] Magnus Jansson. A new subspace identification method for open and closed loop data. *Proceedings of the 16th IFAC world congress*, 2005.

[18] Torben Knudsen. Consistency analysis of subspace identification methods based on a linear regression approach. *Automatica*, 37(1):81–89, 2001.

[19] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[20] Lawton H Lee and Kameshwar Poolla. Identification of linear parameter-varying systems using nonlinear programming. *Journal of dynamic systems, measurement, and control*, 121(1):71–78, 1999.

[21] Lennart Ljung. *System identification (2nd ed.): theory for the user*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999.

[22] Marco Lovera and Guillaume Mercere. Identification for gain-scheduling: a balanced subspace approach. In *American Control Conference 2007, ACC'07*, page CDROM, 2007.

[23] Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. MPCA: Multilinear principal component analysis of tensor objects. *Neural Networks, IEEE Transactions on*, 19(1):18–39, 2008.

[24] Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. A survey of multilinear subspace learning for tensor data. *Pattern Recog-*

*nition*, 44(7):1540–1551, 2011.

[25] Jurgen Remmlinger, Michael Buchholz, and Klaus Dietmayer. Identification of a bilinear and parameter-varying model for lithium-ion batteries by subspace methods. In *American Control Conference (ACC), 2013*, pages 2268–2273. IEEE, 2013.

[26] Carsten W Scherer. Lpv control and full block multipliers. *Automatica*, 37(3):361–375, 2001.

[27] Jeff S Shamma and Michael Alhans. Gain scheduling: Potential hazards and possible remedies. 1992.

[28] Marco Signoretto, Lieven De Lathauwer, and Johan AK Suykens. Nuclear norms for tensors and their use for convex multilinear estimation. *Submitted to Linear Algebra and Its Applications*, 43, 2010.

[29] Roland Tóth, Hossam Seddik Abbas, and Herbert Werner. On the state-space realization of LPV input-output models: Practical approaches. *Control Systems Technology, IEEE Transactions on*, 20(1):139–153, 2012.

[30] Jan-Willem van Wingerden, Federico Felici, and Michel Verhaegen. Subspace identification of MIMO LPV systems using a piecewise constant scheduling sequence with hard/soft switching. In *Control Conference (ECC), 2007 European*, pages 927–934. IEEE, 2007.

[31] Jan-Willem van Wingerden and Michel Verhaegen. Subspace identification of bilinear and LPV systems for open- and closed-loop data. *Automatica*, 45(2):372 – 381, 2009.

[32] M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *Computer Vision–ECCV 2002*, pages 447–460. Springer, 2002.

[33] Vincent Verdult, Niek Bergboer, and Michel Verhaegen. Identification of fully parameterized linear and nonlinear state-space systems by projected gradient search. In *Proceedings of the 13th IFAC Symposium on System Identification, Rotterdam*, 2003.

[34] Hua Zhou, Lexin Li, and Hongtu Zhu. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552, 2013.