A Kernel-based PCA Approach to Model Reduction of Linear Parameter-varying Systems

Syed Z. Rizvi, *Student Member, IEEE,* Javad Mohammadpour, *Member, IEEE,* Roland Tóth, *Member, IEEE,* Nader Meskin, *Member, IEEE*

***Abstract*—This paper presents a model reduction method for linear parameter-varying (LPV) systems using kernel-based principal component analysis (PCA). For state space LPV models that are affine or rational in the scheduling variables, and the variation of these variables is confined in a polytope, controller synthesis can be elegantly realized by solving the synthesis problem only at the vertices of the polytope. To exploit the computational simplicity of this approach, it is highly desirable to obtain LPV models of systems of interest in an affine or rational form. In this respect, kernel PCA allows to extract principal components of a given data set of scheduling variables in a high-dimensional feature space, reducing complicated coefficient dependencies that otherwise might not be easily reducible in a linear subspace; this gives kernel PCA an advantage over its linear PCA counterpart. We show that high dimensional scheduling variables can be mapped into a set of low dimensional variables through a nonlinear kernel PCA-based mapping. Since the kernel PCA mapping is nonlinear, finding the inverse mapping in order to represent the original scheduling variables requires solving a nonlinear optimization problem; consequently, the reduced LPV model is no longer affine in the reduced scheduling variables. To address this, we formulate an optimization problem to obtain a reduced model that is either affine or rational in the reduced scheduling variables. We apply the proposed model reduction method on a robotic manipulator system and use the reduced LPV model to design a gain-scheduled controller that satisfies an induced $\mathcal{L}_2$ gain performance. Numerical simulations are used to demonstrate the performance of the resulting LPV controller on the nonlinear manipulator model. The achieved performance of the LPV controller with the kernel PCA-based reduced model is also compared with its linear PCA-based counterpart.**

***Index Terms*—Kernel Principal Component Analysis, model reduction, linear parameter-varying systems.**

I. Introduction

Linear parameter-varying (LPV) systems are a class of dynamic systems, in which nonlinear models can be described or approximated using a linear dynamic relationship between the inputs and outputs of the system. This linear signal relation is considered to be dependent on another set of measurable signals, the so-called scheduling variables, which represent the varying operating conditions of the system. The ability of LPV models to capture nonlinearities in the system dynamics by using linear dynamical relationships that are dependent on time-varying measurable signals makes it possible to apply linear optimal control techniques to nonlinear systems represented by LPV models. However, the number of scheduling variables in an LPV model has a significant impact on the LPV controller design process, often leading to increased computational complexity, conservatism, and overbounding in the scheduling region [1]. In polytopic LPV systems, the complexity of controller synthesis has an *exponential* dependence on the number of scheduling variables, directly resulting in a high computational complexity for controller synthesis as the number of scheduling variables increases. A common objective for deriving an LPV model is, therefore, to limit the number of scheduling variables to a few [2], [3]. This elevates the problem of LPV model reduction to a significant one. Model reduction in the LPV case refers to both a reduction in the number of state variables (model order), as well as reduction of scheduling dependency. These two aspects of complexity are strongly related [3]. In particular, reduction of model order can result in an increased complexity of the dependence, while reduction of dependency is often available via the introduction of extra state variables [3]. Here, we address the problem of reduction in the complexity of the dependency while the model order is preserved. To this end, we employ multivariate data analysis techniques that can capture those components based on a data set and synthesize a simplified scheduling dependency at the cost of a minimal loss of model accuracy.

Principal component analysis (PCA) is a mathematical tool that extracts a set of linearly uncorrelated variables from an observation of possibly correlated variables using orthogonal transformations. The extracted variables are sorted with respect to their variance in the data, making it then possible to extract the components that contain the *principal information*, measured by their corresponding eigenvalues. This means that the data components with very small variance can be neglected without losing much useful information, making PCA a viable mathematical tool for dimensionality reduction [4]. The ability of PCA to reduce the data dimension makes it ideal for reducing the number of scheduling variables, and consequently, the scheduling dependency in LPV models. This is evident from a few papers that have successfully demonstrated the use of PCA for reducing the scheduling variables dimension to represent the same underlying dynamical behavior (see [1], [5]).

A new generation of data processing techniques has appeared in the literature with the emergence of *kernel-based* methods. Kernels are nonlinear functions that enable to perform linear operations in a high-dimensional feature space,

Syed Z. Rizvi and Javad Mohammadpour are with the Complex Systems Control Laboratory (CSCL), College of Engineering, The University of Georgia, Athens, GA 30602, USA. Corresponding author email: `javadm@uga.edu`

Roland Tóth is with the Dept. of Electrical Engineering, Eindhoven University of Technology, The Netherlands.

Nader Meskin is with the Dept. of Electrical Engineering, Qatar University, Qatar.

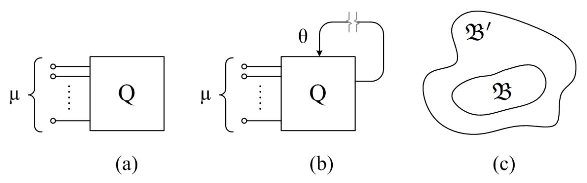

Fig. 1: (a) Original nonlinear system representation. (b) LPV model representation. (c) Resulting behaviors (set of signal trajectories as the solution of the corresponding dynamical model).

where it is much simpler to separate components in the data. Employing what is now widely known as the *kernel trick*, kernel functions perform extractions in the feature space without mapping the original data to the feature space, making component extraction much more efficient and realizable [4]. The variety in choosing kernel functions has further increased the flexibility of exploring different regions of the data more thoroughly. The main contribution of the present paper lies in the use of kernel-based PCA for attaining efficient LPV models with lower dimensional scheduling variables. We examine the advantages of kernel-based PCA over linear PCA and discuss the difficulties associated with kernel PCA in obtaining a pre-image of the reduced variables in the feature space.

The paper is organized as follows: Section II gives an introduction to LPV systems and formulates the problem of interest. In Section III, we describe the use of kernel-based PCA to determine a reduced scheduling dependency and its advantages and pitfalls. An example of an LPV model of a robotic manipulator is considered for model reduction and LPV controller design in Section IV in order to demonstrate the utilization of the proposed method. Throughout this paper, unless otherwise specified, for a given vector $\beta_i \in \mathbb{R}^n$, we use the notation $\beta_{i,j}$ to denote the j^{th} entry of β_i.

II. Preliminaries and Problem Statement

Consider a continuous-time nonlinear system Q shown in Figure 1. Consider that the system is connected to its environment via the signals $\mu : \mathbb{R} \to \mathbb{R}_\mu,\ \mathbb{R}_\mu \subseteq \mathbb{R}^s$, whose entries are related to each other via a (possibly) nonlinear dynamic relation. These signals can be considered as input and output ports of the system. Assume that the set $\mathfrak{B}$ denotes the set of all trajectories of μ that are compatible with Q, *i.e.*, valid solutions of the underlying dynamical equations. For a given time instant t, we introduce auxiliary variables $\theta_t = p(\mu_t), p : \mathbb{R}^s \to \mathbb{R}^l$, which reformulates the system representation of Q as shown in Figure 1(b), where given the trajectory of $\theta : \mathbb{R} \to \mathbb{R}_\theta,\ \mathbb{R}_\theta \subseteq \mathbb{R}^l$, substitution of θ into the representation of Q gives a linear dynamic system. Let us denote by Θ all possible trajectories of θ that are allowed in Q. The reformulated θ-dependent linear system will have a solution set $\mathfrak{B}' = \{(\theta,\mu) | \theta \in \Theta, \mu : \mathbb{R} \to \mathbb{R}_\mu\}$, and the behavior $\mathfrak{B}'$ contains $\mathfrak{B}$, *i.e.*, more specifically $\mathfrak{B} \subseteq \{\mu | \ \exists\ \theta \in \Theta$ s.t. the reformulated system description is satisfied$\}$, giving a linear, but θ-dependent description of Q. The reformulated system now represents an LPV system. A particular objective is to choose θ such that this embedding of $\mathfrak{B}$ is as tight as possible. In principle, the auxiliary variables θ are functions of the measurable signals μ and allow to write the nonlinear system Q as a linear dynamic, but θ-dependent, mapping between the inputs and the outputs; the entries of θ are also known as the *scheduling variables*. This enables us to use various linear control design tools formulated in the form of convex problems for the nonlinear system described by an LPV representation. In case θ is measurable in the considered system, then such a controller can be directly implemented. In real world applications, there can be several different relations between the scheduling variables θ and the measurable signals μ. Variables θ might even be free variables with respect to Q; however, often θ depends on other signals, in which case, the resulting system is often referred to as a quasi-LPV system [6]. In principle, however, the LPV framework and control synthesis problem remains invariant while only the representation of the nonlinear behavior becomes conservative, as indicated in Figure 1.

Consider an LPV system in continuous-time, described by a state space representation as

$$\begin{aligned}\dot{x}_t &= A(\theta_t)x_t + B(\theta_t)u_t,\\ y_t &= C(\theta_t)x_t + D(\theta_t)u_t,\end{aligned} \tag{1}$$

where $x_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$, and $y_t \in \mathbb{R}^{n_y}$ represent the states, inputs, and outputs at time instant t, respectively. The state space matrices are continuous functions of the scheduling variables $\theta_t \in \mathbb{R}_\theta,\ \mathbb{R}_\theta \subseteq \mathbb{R}^l$. The dependence of the scheduling variables on the measurable signals available from the system can be written as

$$\theta_t = p(\mu_t), \quad p : \mathbb{R}^s \to \mathbb{R}^l. \tag{2}$$

An LPV state space representation is considered to be *affine* in the scheduling variables if

$$Q(\theta_t) = Q_0 + \sum_{i=1}^{l} Q_i \theta_{t,i}, \tag{3}$$

where $\theta_{t,i}$ is the i^{th} entry of θ_t, and $Q(\theta_t)$ is a compact representation of the system $Q(\theta_t) = \left[\begin{smallmatrix} A(\theta_t) & B(\theta_t) \\ C(\theta_t) & D(\theta_t)\end{smallmatrix}\right]$. Next, consider the compact convex set $R_\theta \subseteq \mathbb{R}^l, \theta_t \in R_\theta, \forall t > 0$, *i.e.*, the considered scheduling region or operating region for the system, defined by the vertices $R_\theta := \text{Co}\{\theta_{v_1} \cdots \theta_{v_N}\}$, where Co denotes the minimal convex hull and $N = 2^l$ denotes the number of vertices defining the polytope of the scheduling variable $\theta_t \in \mathbb{R}^l$. Hence, θ_t at any time t can be obtained by a convex combination of the vertices as

$$\theta_t = \sum_{i=1}^{N} \beta_i \theta_{v_i}, \text{ with } \beta_i \geq 0,\ \sum_{i=1}^{N} \beta_i = 1. \tag{4}$$

Given the fact that θ_t can be obtained by a convex combination of the vertices θ_{v_i}, and that $Q(\theta)$ depends affinely on the scheduling variables, *i.e.*, conditions (3)-(4), it follows that the system can be represented by a linear combination of multiple LTI systems at the vertices. Such a representation of the system is referred to as a *polytopic* state space LPV representation [7]. The same applies to a desired reduced LPV model of (1). The affine dependence condition, along with the fact that the reduced scheduling variables vary in a polytope is important to be preserved during model reduction as this makes it possible to represent the reduced model as a convex combination of

LTI systems at the vertices of this polytope, and hence, makes controller synthesis problem computationally tractable due to the need to solve the controller design problem only at the vertices [7]. Similar tractability can be achieved when the LPV representation is rationally dependent on the scheduling variables [8]. Therefore, given the system representation (1), with measurable signals μ_t, and scheduling variables defined in (2), the problem of LPV model reduction explored in this paper can be stated as follows: Find a mapping defined by

$$\rho_t = q(\mu_t), \quad q:\ \mathbb{R}^s \to \mathbb{R}^m, \tag{5}$$

where $m < l$, such that the trajectories of the reduced model represented by the following equations

$$\begin{aligned} \dot{\hat{x}}_t &= \tilde{A}(\rho_t)\hat{x}_t + \tilde{B}(\rho_t)u_t, \\ y_t &= \tilde{C}(\rho_t)\hat{x}_t + \tilde{D}(\rho_t)u_t, \end{aligned} \tag{6}$$

can accurately follow the trajectories of (1) and $\tilde{A}(\cdot), \tilde{B}(\cdot), \tilde{C}(\cdot)$, and $\tilde{D}(\cdot)$ are either affine or rationally dependent on the reduced variables ρ_t. We use kernel PCA to provide a solution for finding lower dimensional scheduling variables $\rho_t \in R_\rho$, where $R_\rho \subseteq \mathbb{R}^m$ and $m < l$ As will be shown, this leads to an efficient tradeoff between accuracy and complexity of the reduced model.

III. Kernel PCA for LPV Model Reduction

In order to perform model reduction on LPV scheduling variables, one first needs to collect data from measurements or simulations. Given the LPV model (1), the scheduling variables $\theta_t \in \mathbb{R}^l$ are computed and collected as $\Theta = \begin{bmatrix} \theta_1 \cdots \theta_n \end{bmatrix} = \begin{bmatrix} p(\mu_1) \cdots p(\mu_n) \end{bmatrix} \in \mathbb{R}^{l\times n}$, where n denotes the number of collected samples and $n \geq l$. For linear PCA, a covariance matrix of the scheduling variation is then calculated as $\bar{C} = \frac{1}{n}\Theta\Theta^\top$ after centering the data around zero mean [9]. We then solve an eigenvalue problem that gives the new lower-dimensional scheduling variables $\rho_t \in \mathbb{R}^m, m < l$, such that the resulting reduced model state space matrices are affine in ρ_t. Details on the use of linear PCA for LPV model reduction are reported in [1] and are not repeated here for brevity.

Kernel-based PCA, more simply known as kernel PCA, is an extension of the traditional linear PCA approach, in which the linear dot product operation is performed in a higher dimensional feature space [4]. Kernel PCA first maps the data into a possibly high-dimensional feature space F via a usually nonlinear map $\Phi : \mathbb{R}^N \to F$, and then takes the dot product there [10]. Due to the high dimension of this feature space, separation of features or components in the data is much easily realizable. The effectiveness of kernel-based methods primarily lies in the now-famous *kernel trick*, which allows performing linear operations in the feature space without explicitly mapping the parameters into the feature space. This has led to various applications of kernel PCA such as feature extraction in facial recognition [11] and denoising [10], among several others. For LPV modeling, the use of kernel PCA can lead to reduced complexity of the models/system representations by reducing the number of scheduling variables.

Let us assume that the scheduling variables of the considered LPV description (1) are mapped into the feature space as $\Phi(\theta_1), \Phi(\theta_2), \cdots, \Phi(\theta_n)$, where n is the number of samples. At this point, we also assume that the mapped parameters are centered, *i.e.*, $\sum_{j=1}^n \Phi(\theta_j) = 0$. To perform traditional PCA on this data, we derive the covariance matrix as

$$\bar{C} = \frac{1}{n}\sum_{j=1}^{n} \Phi(\theta_j)\Phi^\top(\theta_j). \tag{7}$$

In order to select the principal components, the relation $\lambda v = \bar{C}v$ should hold. We can, therefore, deduce the following

$$\lambda(v \cdot \Phi(\theta_i)) = (\bar{C}v \cdot \Phi(\theta_i)), \quad \forall\, i = 1, \cdots, n, \tag{8}$$

where $(a \cdot b) = a^\top b$ denotes dot product. Note that (7) implies that there exist coefficients α_w for $w = 1, \cdots, n$, such that the eigenvectors of $\bar{C}$ belong to the span of $\Phi(\theta_j)$ for all j. Substituting $v = \sum_{w=1}^n \alpha_w \Phi(\theta_w)$ and (7) in (8), we get [4]

$$\begin{aligned} &\lambda \sum_{w=1}^{n} \alpha_w(\Phi(\theta_w)\cdot\Phi(\theta_i)) = \\ &\qquad \frac{1}{n}\sum_{j=1}^{n}\sum_{w=1}^{n} \alpha_w\left(\Phi(\theta_j)\cdot\Phi(\theta_w)\right)\left(\Phi(\theta_j)\cdot\Phi(\theta_i)\right), \end{aligned} \tag{9}$$

for $i = 1, \cdots, n$. Next, we define a Gram or kernel matrix $K \in \mathbb{R}^{n\times n}$ as

$$K_{ij} = (\Phi(\theta_i)\cdot\Phi(\theta_j)) = k(\theta_i, \theta_j), \tag{10}$$

where $k(\cdot,\cdot)$ is a nonlinear kernel function; later, we shall elaborate on the choice of these kernels. Substituting (10) in (9) and writing it in matrix form, we obtain $n\lambda\alpha = K\alpha$ for non-zero eigenvalues, where $\alpha = [\alpha_1 \cdots \alpha_n]^\top$. Each solution α_r corresponding to the non-zero eigenvalue λ_r is needed to be normalized. We skip the details here for brevity and refer the interested reader to [4]. Lastly, for principal component extraction, we compute the projections of the image of a test point θ_t onto the eigenvector v_r in the feature space as

$$\begin{aligned} \rho_{t,r} &= (v_r \cdot \Phi(\theta_t)) = \sum_{j=1}^{n} \alpha_{r,j}(\Phi(\theta_j)\cdot\Phi(\theta_t)) \\ &= \sum_{j=1}^{n} \alpha_{r,j}\ k(\theta_j, \theta_t) = \sum_{j=1}^{n} \alpha_{r,j}\ k(\theta_j, p(\mu_t)) = q(\mu_t), \end{aligned} \tag{11}$$

where $\rho_{t,r}$ is the projection of $\Phi(\theta_t)$ on v_r, and is the r^{th} entry of ρ_t. It is noteworthy to mention that the above equation does not explicitly require the computation of feature space map $\Phi(\theta_j)$, but requires only the characterization of the dot product in the feature space which can be defined using a kernel function k. Kernel functions can be chosen from a variety of different functions like *polynomial kernels* $k(\theta_i, \theta_j) = ((\theta_i \cdot \theta_j)+1)^d$, *radial basis function* $k(\theta_i, \theta_j) = \exp\left(-\frac{\|\theta_i - \theta_j\|^2}{\sigma^2}\right)$, and *sigmoid kernels* $k(\theta_i, \theta_j) = \tanh(\kappa(\theta_i \cdot \theta_j) + b)$. Parameters d, σ κ, and b denote the degree of the polynomial, the spread of the radial basis function, and sigmoid kernel parameters; these are essentially tuning parameters chosen by the user [12]. We shall further discuss choosing of appropriate kernel functions in the case study presented in the next section.

We recall that the data was initially assumed to be centered in the feature space, which is not always true. Since one cannot, in general, center the data because of the unavailability of feature maps, the centered Gram matrix can be calculated by replacing $\Phi(\theta_i)$ with $\tilde{\Phi}(\theta_i) := \Phi(\theta_i) - \frac{1}{n}\sum_{i=1}^n \Phi(\theta_i)$ and deriving the Gram matrix again as detailed in [13]; here, we reproduce the centered Gram matrix as

$$\tilde{K} = K - 1_n K - K 1_n + 1_n K 1_n, \tag{12}$$

where $1_n \in \mathbb{R}^{n \times n}$ with its each entry being $1/n$.

A. Accuracy of the estimated LPV model

The accuracy of the estimated model can be gauged from the fraction of total data variation calculated as

$$a(m) = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{\bar{m}} \lambda_i}, \tag{13}$$

where m is the number of reduced variables, and λ_i denotes the i^{th} eigenvalue of the kernel matrix $\tilde{K}$ in (12); m can be chosen by the user based on significant eigenvalues. Variable $\bar{m}$ is equal to n and l for kernel and linear PCA, respectively. The rationale for using this accuracy measure comes from [13], which states that the first m principal components, *i.e.*, projections on eigenvectors, carry more variance than any other m orthogonal directions. It is, therefore, logical to measure accuracy in terms of the variance (energy) represented by the corresponding m eigenvalues.

B. The pre-imaging problem

For linear PCA, the reduced LPV model is affine in the new scheduling variables ρ_t, as shown in [1]. Kernel PCA, however, suffers from two problems when it comes to reconstructing the original scheduling variables. Firstly, given the reduced variables ρ_t, there is no systematic way of reconstructing the original variables θ_t, and the problem of finding the "pre-image" of the reduced scheduling variable in the input space is unsolvable in the general case. Schölkopf *et al.* argued in [10] that instead, it is often feasible to find an estimate of the pre-image $\tilde{\theta}_t$. Secondly, to find the estimate $\tilde{\theta}_t$, one has to solve a nonlinear optimization problem [10] whose convergence depends highly on factors such as initial conditions and the choice of kernel function [14]. Moreover, running an optimization problem to find $\tilde{\theta}_t$ at each sampling instant t in real-time is not practical for online control implementation due to heavy computation power needed for such an approach.

Since in most cases, $\tilde{\theta}_t$ denotes an estimate of θ_t, we can write the following for the kernel PCA-based reduced model

$$\tilde{Q}(\rho_t) = \begin{bmatrix} \tilde{A}(\rho_t) & \tilde{B}(\rho_t) \\ \tilde{C}(\rho_t) & \tilde{D}(\rho_t) \end{bmatrix} \approx \begin{bmatrix} A(\tilde{\theta}_t) & B(\tilde{\theta}_t) \\ C(\tilde{\theta}_t) & D(\tilde{\theta}_t) \end{bmatrix} = Q(\tilde{\theta}_t). \tag{14}$$

Using (3) and (14), we can relate the reduced model to the full-order LPV model as follows:

$$\tilde{Q}(\rho_t) \approx Q(\tilde{\theta}_t) = Q_0 + \sum_{i=1}^{l} Q_i \tilde{\theta}_{t,i} = Q_0 + \sum_{i=1}^{l} Q_i f_i\left(\rho_t, \tilde{\theta}_t\right), \tag{15}$$

where $f(\rho_t, \tilde{\theta}_t)$ is a nonlinear pre-image function as derived in [14].

C. Optimizing the reduced LPV model for controller synthesis

The reduced model in (15) depends on ρ_t through a nonlinear function. In our problem statement in Section II, we aimed at finding a reduced model with affine or rational dependence on ρ_t; this was desired for ease of controller synthesis detailed in the next section. Now, suppose that the kernel PCA-based reduced model is represented by state space matrices $\breve{A}(\rho_t)$, $\breve{B}(\rho_t)$, $\breve{C}(\rho_t)$, and $\breve{D}(\rho_t)$ that have affine dependence on ρ_t. According to Apkarian et al. [7], the vertex property implies that for such an LPV representation, the state space matrices for any ρ_t belong to a matrix polytope as

$$\begin{bmatrix} \breve{A}(\rho_t) & \breve{B}(\rho_t) \\ \breve{C}(\rho_t) & \breve{D}(\rho_t) \end{bmatrix} \in \mathcal{P} := \text{Co}\left\{ \begin{bmatrix} \breve{A}_{v_i} & \breve{B}_{v_i} \\ \breve{C}_{v_i} & \breve{D}_{v_i} \end{bmatrix}, i = 1, \cdots, 2^m \right\},$$

where m is the number of scheduling variables, and $\breve{A}_{v_i}$ denotes the matrix corresponding to $\breve{A}(\rho_t)$ at the i^{th} vertex of the polytope. Therefore, for any given ρ_t, the system matrices can be described as a convex combination of the matrices at the vertices of the polytope. For a polytopic LPV system with affine dependence, we only need to solve the controller synthesis problem, detailed in the next section, with respect to the vertices of the polytope of scheduling region to ensure closed-loop system stability and quadratic $\mathcal{H}_\infty$ performance for all variations of ρ_t within the polytope. Scherer showed in [8] that a similar property for LPV controller synthesis holds if the defining state space matrices of the LPV model depend rationally on ρ_t. In such a case, a linear fractional representation is admitted by the system. In both cases, the controller synthesis problem needs to be solved with respect to the vertices of the polytope and stability and quadratic $\mathcal{H}_\infty$ performance satisfied at each vertex means that it is satisfied for any ρ_t within the polytope. The same holds true for the performance and stability of the closed-loop system. If, on the other hand, the dependence is not affine or rational, then performance is not guaranteed for any ρ_t; our best bet in that case is to solve the controller synthesis problem over a fine grid across the polytope. A coarse grid will lead to degraded performance while a fine grid will lead to a massive increase in computational complexity. By performing kernel PCA-based model reduction, we have reduced the number of vertices of the polytope from 2^l to 2^m; henceforth, we need to make sure that the dependence is affine or rational in order to exploit this reduction. This will enable us to solve the controller synthesis problem only at these reduced number of vertices. To ensure affine or rational dependence on ρ_t, we avoid pre-imaging by recasting the projection w.r.t. the system matrices as

$$\min_{R_i, P_i, S_i \; i=0,1,\cdots,m} \quad \frac{1}{n} \sum_{t=1}^{n} \| Q(\theta_t) - R(\rho_t) \|_{\text{F}}^2, \tag{16}$$

where $Q(\theta_t)$ is as defined in (3), $\|\cdot\|_{\text{F}}$ represents the Frobenius norm, m is the number of reduced scheduling variables, n is the total number of time samples, and $R(\rho_t) = R_0 + \sum_{i=1}^{m} R_i \rho_{t,i} = \left[\begin{smallmatrix} \breve{A}(\rho_t) & \breve{B}(\rho_t) \\ \breve{C}(\rho_t) & \breve{D}(\rho_t) \end{smallmatrix}\right]$ for an affine approximation, or $R(\rho_t) = \left\{ P_0 + \sum_{i=1}^{m} P_i \left(\rho_{t,i}\right)^j \right\}^{-1} \left\{ S_0 + \sum_{i=1}^{m} S_i \left(\rho_{t,i}\right)^j \right\} = \left[\begin{smallmatrix} \breve{A}(\rho_t) & \breve{B}(\rho_t) \\ \breve{C}(\rho_t) & \breve{D}(\rho_t) \end{smallmatrix}\right]$ for a rational approximation, where $j \in \mathbb{Z}_+$ is a non-negative integer. This gives the state space matrices $\breve{A}$, $\breve{B}$, $\breve{C}$, and $\breve{D}$ of the reduced LPV model. It is important to note that this optimization problem is solved offline. If the considered scheduling trajectories in defining the data-driven mapping have sufficient information content about all the operating regions of the modeled system, then the reduced

Algorithm 1 Applying kernel PCA for LPV model reduction

```
Step 1: Obtain a set of measurable signals using measurements or simulations, covering the desired range of operation.
Step 2: Compute the trajectories of the corresponding scheduling variables $\theta_t$ for $t = 1, \cdots, n$.
Step 3: Compute the kernel matrix $K$ using (10).
Step 4: Center the data in the feature space to find $\tilde{K}$ using (12).
Step 5: Diagonalizing $\tilde{K}$, normalize eigenvectors $\alpha$, and save $\alpha$ for projecting scheduling variables $\theta_t$ and extracting $\rho_t$.
Step 7: Compute $\rho_t$ for $t = 1, \cdots, n$ using (11).
Step 8: Solve the optimization problem (16) and obtain a reduced LPV state space model.
```

model is expected to provide a good estimate of the actual model with $m < l$. The proposed model reduction method is summarized in Algorithm 1.

IV. Robotic Manipulator Example

The dynamic model of a two-link planar robotic manipulator is considered here with its configuration shown in Figure 2. Detailed nonlinear model and the associated parameters have been taken from [15]. The lower arm of the robot is known as the shoulder, while the upper part is simply known as the arm, to which the actuator is attached. The two joints are connected via a gear servo mechanism. Following the ideas in [1], [15], a state space LPV model is derived. For the sake of brevity, we denote the scheduling variables θ_t and ρ_t simply as θ and ρ, dropping the time index. The state space matrices of the derived LPV model are given as

$$A(\theta) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ cd\theta_3 & -be\theta_4 & \theta_5 & b\theta_6 \\ -bd\theta_7 & ae\theta_8 & \theta_9 & \theta_{10} \end{bmatrix}, \ C = I_{4\times4},$$

$$B(\theta) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ cnk_m\theta_1 & -bnk_m\theta_2 \\ -bnk_m\theta_2 & ank_m\theta_1 \end{bmatrix}, \ D = O_{4\times2}, \tag{17}$$

where variables $\theta = [\theta_1 \ \cdots \ \theta_{10}]^\top$ represent the time-varying scheduling variables and k_m is the motor constant linking current to torque and is taken as unity. For details about the model constants and the variables $\theta_1, \cdots, \theta_{10}$, see [1], [15]. The state vector is given by $x = [q_1 \ q_2 \ \dot{q}_1 \ \dot{q}_2]^\top$. Angles of the two links with respect to the vertical reference frame give the first two states, while the angular velocities make up the other two states. Motor torques for the two joints make up the two control inputs u for this plant. This LPV model has a total of $l = 10$ scheduling variables. The objective here is to reduce the number of scheduling variables in order to reduce the over-parametrization in the given LPV model. The measurable signals μ in this case are the states, and hence, $s = 4$. Using PCA, we seek to reduce the number of variables to m, with $m < l$, such that the reduced LPV model can achieve an efficient trade-off between accuracy and complexity.

A set of trajectories is generated with open-loop simulation of the LPV model using sinusoidal inputs u and the resulting scheduling variables are computed using the states. We apply linear and kernel PCA on the scheduling variable data and then compare the accuracy criterion (13) of the approximated LPV models as a function of the dimension of reduced variables.

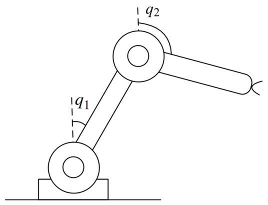

Fig. 2: Configuration of the 2-DoF robotic manipulator

A. Choosing the kernel function

The general question of how to choose a kernel function is still an open problem [16]. Jolliffe *et al.* argued in [9] that finding the first eigenvector *w.r.t.* the centered data, in this case, $\Theta_c = \mathcal{C}(\Theta) = \Theta - \theta_{\text{mean}}$, can also be formulated as finding the direction which exhibits the most variance *w.r.t.* the data. The next eigenvectors form an orthonormal basis where each eigenvector satisfies a similar property *w.r.t.* the remaining subspace. Schölkopf argued in [4] that the same reasoning can be applied in case of kernel PCA. Owing to this, we measure the direction with maximum variance using the corresponding eigenvalues formulating the accuracy measure (13). Different kernel functions are used on the collected data. The accuracy measure obtained as a function of number of components extracted shows that polynomial kernels provide the maximum measure of accuracy for $m = 1$ or $m = 2$ components. Different values of kernel parameters are searched over a fine grid. Polynomial kernel of degree 2 gives an accuracy of 65.5%; degree 7 gives an accuracy of around 94%, after which, increasing the degree of the polynomial kernel shows little improvement. Radial basis function (RBF) gives a maximum accuracy of 67% after tuning the kernel parameter σ over a fine grid. Sigmoid kernel provides around 71% accuracy after fine-tuning its parameters. Detailed results for various kernels with the tuned parameters are tabulated in Table I. Accuracy results for polynomial, RBF, and sigmoid kernels as functions of extracted component m are shown in Figure 3, where it can be seen that the polynomial kernel is the only kernel that provides better than 90% accuracy for $m = 1$; this occurs for polynomial kernel of order $d = 5$ and higher. We finally choose a 7th order polynomial kernel for model reduction. We once again recall that there is not a single way of "optimally" choosing a kernel function, and the problem of kernel selection remains an open and most commonly, application-specific problem [16]. The results obtained for kernel PCA with the chosen degree 7 polynomial kernel are shown in comparison with linear PCA in Figure 4(a). Higher accuracy is clearly observed in case of kernel PCA for $m \leq 5$.

Once the reduced scheduling variables are obtained, cost function (16) is minimized to obtain an affine reduced model. As we will see later, for the given robotic manipulator example, the resulting affine reduced model provides a high accuracy in terms of predicting open-loop model output and controller performance. Therefore, we do not attempt to fit a rational reduced model. Once the model reduction is performed, the reduced models are then simulated using a fresh set of sinusoidal torque input signals, different from the signals used for training. Figure 4(b) shows the states of the open-loop robotic manipulator over a period of 10 seconds. It

TABLE I: Accuracy measure for different kernels as a function of number of scheduling variables m.

kernel fcn	*parameters*	accuracy (%)									
		$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$	$m = 8$	$m = 9$	$m = 10$
	$d = 2$	65.55	82.09	86.57	90.79	92.88	94.84	96.42	97.88	98.85	99.32
	$d = 4$	87.89	92.74	95.98	97.50	98.10	98.61	98.93	99.20	99.39	99.51
	$d = 5$	91.41	95.03	97.70	98.82	99.20	99.44	99.55	99.66	99.74	99.79
polynomial	$d = 6$	93.04	97.01	98.37	99.34	99.64	99.74	99.82	99.86	99.89	99.91
	$d = 7$	94.19	98.57	99.33	99.69	99.90	99.94	99.96	99.97	99.98	99.98
	$d = 8$	94.36	98.88	99.57	99.76	99.93	99.97	99.98	99.99	99.99	99.99
	$d = 9$	94.44	99.06	99.70	99.85	99.95	99.98	99.99	99.99	100	100
	$\sigma = 5$	49.96	74.49	83.77	89.20	93.08	95.24	96.70	97.70	98.49	98.94
RBF	$\sigma = 10$	63.93	84.12	91.66	95.23	97.21	98.22	98.91	99.38	99.63	99.77
	$\sigma = 15$	67.17	86.03	93.21	96.44	98.06	98.80	99.34	99.66	99.82	99.89
	$\kappa = 0.05$ $b = 0.01$	70.09	89.01	96.40	99.63	99.98	100	100	100	100	100
sigmoid	$\kappa = 0.05$ $b = 0.05$	70.55	89.30	96.69	99.92	99.99	99.99	99.99	100	100	100
	$\kappa = 0.05$ $b = 0.1$	71.14	89.66	97.06	99.99	99.99	100	100	100	100	100
	$\kappa = 0.1$ $b = 0.1$	71.43	91.76	99.82	100	100	100	100	100	100	100

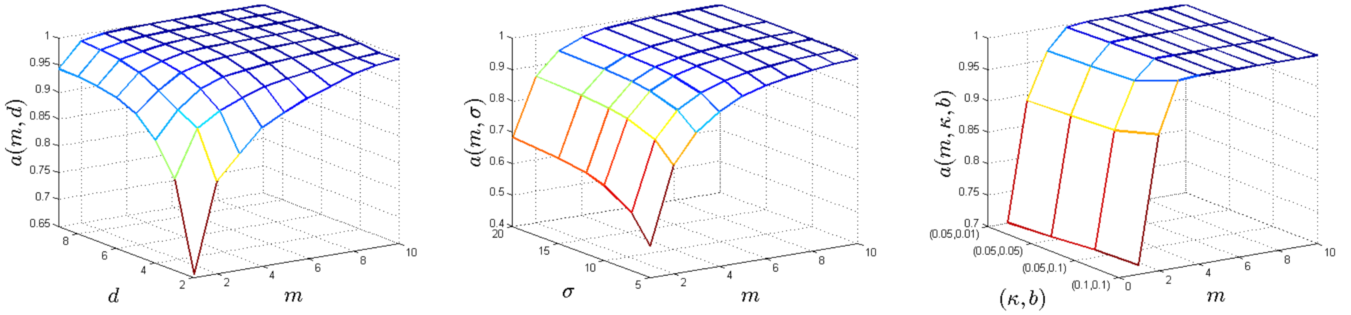

Fig. 3: From left to right: Accuracy plots as functions of number of scheduling variables m for (a) polynomial kernel (b) radial basis function kernel and (c) sigmoid kernel. In each case, the y-axis shows the kernel parameters.

should be noted here that by a "full-order" LPV model, we refer to the LPV model with all the original 10 scheduling variables. The output of the full-order LPV model (solid blue line), linear PCA-based reduced LPV model (red dashed line), and the kernel PCA-based reduced LPV model (black dotted line) are compared. All three models were excited with the same sinusoidal inputs and both reduced LPV models use $m = 1$ scheduling variable. These results illustrate that the time response of the open loop kernel PCA-based reduced model with a single scheduling variable mimics the response of the full order LPV model. The output of the linear PCA-based reduced model diverges from the full order model output after a few seconds. We define the *Best Fit Ratio* criterion as

$$\text{BFR} := 100\% \cdot \max\left(\frac{\|x-\hat{x}\|_2}{\|x-\bar{x}\|_2}, 0\right), \tag{18}$$

where $\hat{x}$ represents the simulated states of the approximated model while $\bar{x}$ is the mean value of the states of the original system denoted by x, and $\|\cdot\|_2$ denotes $\mathcal{L}_2$ norm, respectively. Monte-Carlo simulations are run for 50 different trajectories of the input torque with randomly generated magnitudes and frequencies, different from the signals used for data collection and training; mean BFR value of each state is computed over the 50 runs. The mean BFR values for the states are then averaged over the four states and tabulated in Table II, in which kPCA and lPCA denote kernel and linear PCA, respectively.

To examine the closed-loop performance of LPV controllers designed based on the full-order, as well as reduced LPV models, we explore LPV controller design in the next subsection.

B. LPV controller design

The LPV controller design configuration is illustrated in Figure 5 (a). The polytopic gain-scheduled controller $K_c(\rho)$ is designed based on the reduced LPV model $G(\rho)$. Variables z, w, and y denote controlled outputs, external disturbance, and measurements. We design $K_c(\rho)$ such that it satisfies an induced $\mathcal{L}_2$ gain performance and is described as

$$K_c(\rho) : \begin{cases} \dot{\zeta}_t = A_K(\rho)\zeta_t + B_K(\rho)y_t, \\ u_t = C_K(\rho)\zeta_t + D_K(\rho)y_t. \end{cases} \tag{19}$$

This control methodology is specific to LPV models that have affine dependence on the scheduling variables ρ with the scheduling variables ranging within a fixed polytope and available for measurement [7]. We define the induced $\mathcal{L}_2$ gain and the induced $\mathcal{L}_2$ gain performance as follows.

Definition 1 (Induced $\mathcal{L}_2$ gain for LPV systems) [7]: For the closed-loop LPV system shown in Figure 5 (a), the energy-to-energy gain, or induced $\mathcal{L}_2$ gain, from external disturbance w to controlled outputs z is defined as

$$\|T_{wz}\|_{i,2} = \sup_{\rho} \sup_{w \neq 0} \frac{\|z\|_{\mathcal{L}_2}}{\|w\|_{\mathcal{L}_2}}, \tag{20}$$

where i denotes that the norm is "induced". This gain indicates the worst-case output energy $\|z\|_{\mathcal{L}_2}$ over all bounded energy disturbances $\|w\|_{\mathcal{L}_2}$ for all admissible values of the scheduling variables ρ.

TABLE II: Open-loop simulation Monte-Carlo statistics for linear and kernel PCA

	$m=1$		$m=2$		$m=3$		$m=4$		$m=5$		$m=6$	
	kPCA	lPCA	kPCA	lPCA	kPCA	lPCA	kPCA	lPCA	kPCA	lPCA	kPCA	lPCA
BFR (%)	84.13	62.31	89.08	71.53	93.71	78.41	96.06	85.15	97.67	91.22	98.72	96.01

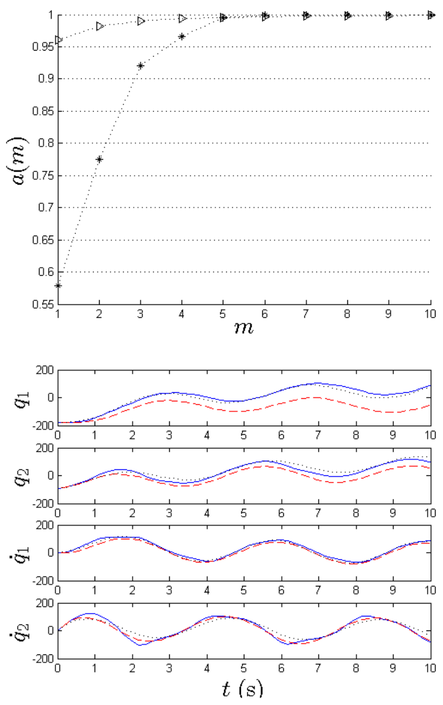

Fig. 4: (Top) Accuracy of the approximation (13) as a function of the number of scheduling variables m for linear PCA (*) and polynomial-based kernel PCA ($\triangleright$). (Bottom) States of the robotic manipulator full order LPV model (solid blue line), linear PCA-based reduced LPV model (red dashed line), and kernel PCA-based reduced LPV model (black dotted line) for $m = 1$.

Definition 2 (Induced $\mathcal{L}_2$ gain performance) [7]: The closed-loop LPV system of Figure 5 (a) has an induced $\mathcal{L}_2$ gain performance less than γ if there exists a symmetric positive-definite matrix X such that

$$\begin{bmatrix} A_{cl}^{\top}(\rho)X + XA_{cl}(\rho) & XB_{cl}(\rho) & C_{cl}^{\top}(\rho) \\ \star & -\gamma I & D_{cl}^{\top}(\rho) \\ \star & \star & -\gamma I \end{bmatrix} \prec 0, \qquad (21)$$

for all admissible trajectories of ρ, where A_{cl}, B_{cl}, C_{cl}, and D_{cl} are the closed-loop state space matrices. This holds true for systems with fixed values of ρ. In the case of a polytopic LPV system, the matrix X in the inequality (21) is found by solving a finite number of *linear matrix inequalities* (LMIs). The vertex property implies that for polytopic LPV representation with affine dependence on ρ, the inequality (21) holds for all trajectories of ρ within the polytope, if it holds true at the vertices (for proof, see [7]). Therefore, we can achieve a closed-loop $\mathcal{L}_2$ gain performance γ if (21) holds true at all vertices ρ_{v_i} of the polytope of scheduling variables. One can see the benefit of model reduction at this point; for a reduced LPV model with a lower number of scheduling variables and affine dependence, the number of LMIs that need to be solved in order to design an LPV controller based on a reduced model decreases exponentially. In case of the reduced robot model, we need to solve only two LMIs.

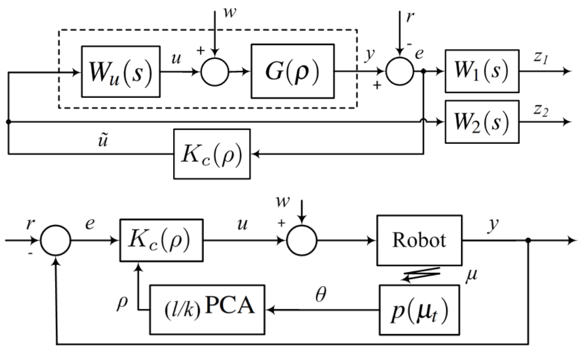

Fig. 5: (Top) Generalized configuration of closed-loop system composed of reduced LPV model, LPV controller, and design weights. (Bottom) Control of the robotic manipulator using an LPV controller $K_c(\rho)$ based on the reduced LPV model.

The design objective is for the measured output y, which consists of the first two states, *i.e.*, the two joint positions, to track the desired reference trajectories given by r. For controller synthesis based on polytopic LPV models, the plant input and output matrices, B and C, need to be independent of the scheduling variables (for details, see [7]). This is not the case for the manipulator example considered here; both the full order LPV model (17) and the reduced model have an input matrix $\breve{B}(\rho)$ that is a function of ρ. This restriction can be worked around as shown in Figure 5 (top), by augmenting the plant with a low pass filter $W_u(s)$ having sufficiently large bandwidth [7]. A first order filter is selected for this purpose. Filters $W_1(s)$ and $W_2(s)$ are loop-shaping filters, where $W_1(s)$ is selected to be a first order low pass filter with a pole close to the origin in order to minimize the steady-state tracking error, and $W_2(s)$ is chosen as a static gain. These filters are selected and tuned by trial and error, seeking the minimization of the induced $\mathcal{L}_2$ gain from the external disturbance $w_t = \begin{bmatrix} r_1 & r_2 & w_1 & w_2 \end{bmatrix}^{\top}$ to the controlled outputs $z_t = \begin{bmatrix} z_1 & z_2 \end{bmatrix}^{\top}$ in order to enforce the performance requirement $\|T_{wz}\|_{i,2} < \gamma$.

Using the kernel PCA-based reduced LPV model with $m = 1$ scheduling variable, the LPV controller matrices at the vertices of the polytope are obtained using the MATLAB robust control toolbox command `hinfgs`. An 8th order controller is designed and a minimum value of $\gamma = 0.11$ is obtained. The designed controller is then placed in the control loop (see Figure 5 (bottom)) to control the robotic manipulator model. Reduced scheduling variables are obtained using (11) in order to schedule the controller $K_c(\rho)$. Process noise w is added to the system input such that a *signal-to-noise ratio* (SNR) of 20dB is maintained. A saturation limit of $|u_i| < 30$ N-m, for $i = 1, 2$, is imposed on the controller outputs in order to mimic the physical constraints on the motor torques at the two joints.

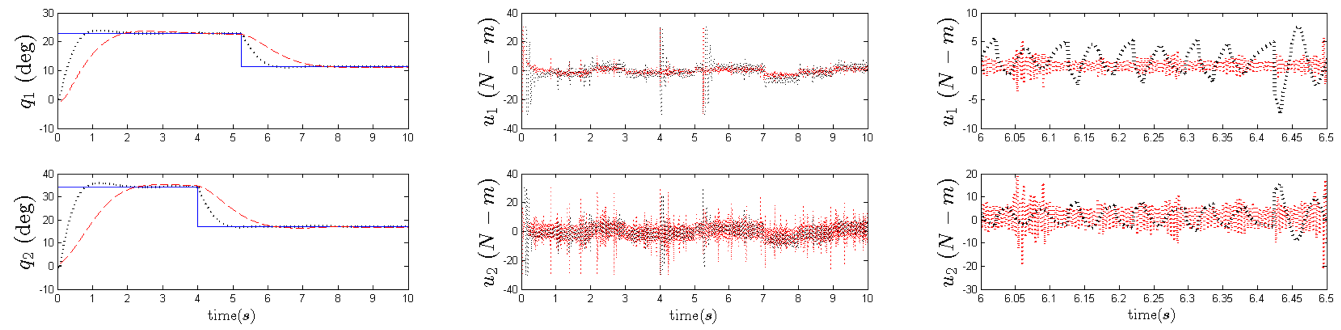

Fig. 6: Closed-loop control results from left to right: (a) Reference trajectory (blue solid line) in comparison with controlled outputs based on linear PCA-based reduced model (red dashed line) and kernel PCA-based reduced model (black dotted line); (b) Outputs of the gain scheduled controllers designed using linear PCA-based reduced model (red dashed line) and kernel PCA-based reduced model (black dotted line); (c) Close-up view of controller outputs.

For the sake of comparison, a similar controller is designed based on linear PCA-based reduced model with $m = 1$. An optimal value of $\gamma = 0.18$ is achieved after tuning the filters. Figure 6(a) shows the reference tracking results. Reference trajectories are chosen to be different from the trajectories used in the data-driven kernel PCA reduction. These reference trajectories are represented by blue solid line; linear and kernel PCA-based controlled outputs are shown by red dashed and black dotted lines, respectively. Figure 6(b) shows the controller outputs for the linear and kernel PCA cases. Figure 6(c) shows a magnified view of the linear and kernel-PCA based controller outputs using red dashed and black solid lines, respectively. The results demonstrate efficient reference tracking achieved by the controller designed using kernel PCA-based reduced model, and shows improvement over its linear PCA counterpart.

V. Concluding Remarks

In this paper, we have explored the use of kernel-based PCA for dimensionality reduction of the scheduling variables in LPV models. Reducing the number of scheduling variables directly results in reduction of computational complexity for LPV controller design and implementation. Kernel PCA has been known to be efficient in extracting components of data because of its ability to perform extraction in a high dimensional feature space; it does so using nonlinear kernel functions. This makes the reduced LPV model nonlinear in the reduced scheduling variables. We overcome this problem by solving an optimization problem and obtaining an affine or rational representation with respect to the reduced scheduling variables. We infer that the reduced model is suitable for controller design purpose. In the case of the robotic manipulator example considered in this paper, kernel PCA has been able to reduce the number of scheduling variables from $l = 10$ to $m = 1$, thereby reducing significantly the number of LMIs to be solved for LPV controller synthesis, as well as the controller implementation time. Closed-loop simulations show promising reference tracking, disturbance attenuation and improved settling time. Results in this paper provide encouraging insights into the use of kernel PCA for LPV dependency reduction.

Acknowledgment

This work was made possible by the NPRP grant # NPRP 5-574-2-233 from Qatar National Research Fund (QNRF), a member of Qatar Foundation. The statements made here are solely the responsibility of the authors.

References

[1] A. Kwiatkowski and H. Werner, "PCA-based parameter set mappings for LPV models with fewer parameters and less overbounding," *IEEE Trans. Control Syst. Tech.*, vol. 16, no. 4, pp. 781–788, 2008.

[2] M. Siraj, R. Tóth, and S. Weiland, "Joint order and dependency reduction for LPV state-space models," in *51st IEEE Conf. Decision and Control, 2012*, 2012, pp. 6291–6296.

[3] R. Tóth, H. Abbas, and H. Werner, "On the state-space realization of LPV input-output models: practical approaches," *IEEE Trans. Control Syst. Tech.*, vol. 20, no. 1, pp. 139–153, 2012.

[4] B. Schölkopf and A. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2002.

[5] M. Meisami-Azad, J. Mohammadpour, K. Grigoriadis, M. Harold, and M. Franchek, "LPV gain-scheduled control of SCR aftertreatment systems," *Int. J. Control*, vol. 85, no. 1, pp. 114–133, 2012.

[6] R. Tóth, J. Willems, P. Heurberger, and P. V. den Hof, "The behavioral approach to linear parameter-varying systems," *IEEE Trans. Auto. Control*, vol. 65, no. 11, pp. 2499–2514, 2011.

[7] P. Apkarian, P. Gahinet, and G. Becker, "Self-scheduled $\mathcal{H}_\infty$ control of linear parameter-varying systems: a design example," *Automatica*, vol. 31, no. 9, pp. 1251–1261, 1995.

[8] C. Scherer, "LPV control and full block multipliers," *Automatica*, vol. 37, no. 3, pp. 361–375, 2001.

[9] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[10] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K. Müller, G. Ratsch, and A. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1000–1016, 1999.

[11] Q. Wang, "Kernel principal component analysis and its applications in face recognition and active shape models," *ArXiv e-prints*, 2012, eprint no. 1207.3538.

[12] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Fifth ACM Annual Workshop on Computational Learning Theory*, Pittsburg, PA, 1992, pp. 144–152.

[13] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[14] S. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin, "Parameter set-mapping using kernel-based PCA for linear parameter-varying systems," in *13th European Control Conf.*, Strasbourg, France, 2014, pp. 2744–2749.

[15] Z. Yu, H. Chen, and P. Woo, "Gain scheduled LPV $\mathcal{H}_\infty$ control based on LMI approach for a robotic manipulator," *J. Rob. Syst.*, vol. 19, no. 12, pp. 585–593, 2002.

[16] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Artificial Neural Networks ICANN'97*. Springer, 1997, pp. 583–588.