DATA-DRIVEN LEARNING-BASED IDENTIFICATION AND CONTROL OF LINEAR PARAMETER-VARYING SYSTEMS

by

SYED ZEESHAN RIZVI

(Under the direction of Javad Mohammadpour)

ABSTRACT

This dissertation proposes algorithms for data-driven nonparametric identification, model reduction, and control synthesis of *linear parameter-varying* (LPV) models in the state-space form. We make use of kernelized *machine learning* (ML) and *multivariate analysis* (MVA) tools to develop model reduction techniques that reduce the scheduling dependency in LPV *state-space* (LPV-SS) models, thereby reducing exponentially, the computation complexity of LPV controller synthesis. Further, we formulate a regularized *least squares support vector machine* (LS-SVM)-based algorithm to identify LPV-SS models using inputs, outputs, states, and scheduling variables measurements. The technique is further extended to an *instrumental variable support vector machine* (IV-SVM) approach that is able to identify LPV-SS models under generic noise conditions, including cases where noise not only is colored, but is also correlated with the scheduling variables. The proposed technique seeks to obtain an unbiased estimator of the scheduling dependency functions in the face of noise-induced bias. In case the state variables are not directly measurable, a kernelized *canonical correlation analysis* (CCA)-based routine is proposed to estimate the states from past and future inputs, outputs, and scheduling variables measurements. This provides a non-unique estimate of the states up to a similarity transform. Once the states are estimated, previously-proposed LS-SVM-based algorithm is used in tandem with the estimated states to identify an LPV-SS model. All techniques proposed in this work exploit the use of the so-called

kernel trick, giving extra degrees of freedom to choose different kernel functions and tune their associated hyper-parameters. Different applications including a robotic manipulator and chemical process models are used to verify the different algorithms developed as part of this dissertation.

INDEX WORDS: System Identification; Linear Parameter-varying Models; Kernelized Machine Learning; Kernel trick; Least-squares Support Vector Machine; Model Reduction.

DATA-DRIVEN LEARNING-BASED IDENTIFICATION AND CONTROL OF LINEAR PARAMETER-VARYING SYSTEMS

by

SYED ZEESHAN RIZVI

B.E., N.E.D. University of Engg. & Tech., Pakistan, 2006

M. S., King Fahd Univeristy of Petroleum & Minerals, Saudi Arabia, 2008

A Dissertation Submitted to the Graduate Faculty

of The University of Georgia in Partial Fulfillment

of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2016

DATA-DRIVEN LEARNING-BASED IDENTIFICATION AND CONTROL OF LINEAR PARAMETER-VARYING SYSTEMS

by

SYED ZEESHAN RIZVI

Approved:

Major Professor: Javad Mohammadpour

Committee: Changying Li
Takoi Hamrita
Ardalan Vahidi

Electronic Version Approved:

Suzanne Barbour
Dean of the Graduate School
The University of Georgia
December 2016

DEDICATION

To my late grandfather, *Dr. Qayam ul Haque, Ph.D.*; he inspired me to pursue doctoral research.

ACKNOWLEDGMENTS

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

I am forever thankful to my major professor, Dr. Javad Mohammadpour for his relentless effort, guidance, mentoring, and his patience with me during the course of this work. He has taught me a lot and has shaped my research career to where it is today. I am also thankful to my dissertation committee members: Dr. Changying Li; Dr. Takoi Hamrita; and Dr. Ardalan Vahidi from Clemson University for serving on my committee, and for providing constructive comments on my work.

My utmost gesture of gratitude goes to Dr. Roland Tóth of Eindhoven University of Technology, Eindhoven, The Netherlands. As a leading researcher in LPV identification and an investigator on my Ph.D. project, this work has benefited immensely from his deep insight, his meticulous reviews, and his detailed suggestions and comments to make the mathematical background of the work more sound. His suggestions, especially those pertaining to state estimation in LPV state-space models have helped me make this work a solid theoretical contribution in the area of LPV state-space identification. Credit also goes to Dr. Nader Meskin of Qatar University for his contribution and insights on this work.

Special thanks to Farshid Abbasi for valuable technical discussions and insights into this work. We shared long working hours and sometimes long nights of meeting deadlines in the laboratory together, easing the work-load with insights on the work and lots of coffee.

Last, but not the least, my penultimate gratitude to the amazing people in my family. To my parents, Mumtaz Rizvi and Shad Rizvi, for their patience with my long career path and for the strength of their love and support. The values they have inculcated in me, and their relentless support for all my choices has helped me get to this point. To my sister; her indirect effects on

me are numerous. A special thank you to my wife Nausheen Fatima; she was the crucial support that made this journey pleasant, and even feasible. She managed things in the limited means of a graduate student's life as pleasantly as possible, and made sure that my research efforts did not get affected by problems outside of it. To my toddler daughter, Sarah Rizvi; she has been the joy of my life and the source of happiness that made these years extremely pleasant.

"*Do not worry about your difficulties in Mathematics. I can assure you mine are still greater.*"

Albert Einstein

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION AND LITERATURE SURVEY

1.1 INTRODUCTION TO LINEAR PARAMETER-VARYING MODELS

Linear parameter-varying (LPV) systems are a class of dynamical systems, in which nonlinear models can be described using a linear dynamic relation between the inputs and the outputs. This relation is dependent on measurable time-varying signals, 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.

Consider a continuous-time nonlinear system Q shown in Figure 1.1. The system describes (possibly) nonlinear dynamic relation between the measurable signals $\mu : \mathbb{R} \to \mathbb{R}_\mu,\ \mathbb{R}_\mu \subseteq \mathbb{R}^s$. Let $\mathfrak{B}$ be the set of all trajectories of μ compatible with Q. By introducing an auxiliary variable θ, one can reformulate the system representation of Q as shown in Figure 1.1(b), where given the trajectory of $\theta : \mathbb{R} \to \mathbb{R}_\theta,\ \mathbb{R}_\theta \subseteq \mathbb{R}^l$, all relations between μ are linear. By applying this reformulation of a disconnected θ and assuming that all trajectories of θ are allowed independently from μ, the reformulated system will have a solution set $\mathfrak{B}' = \{(\theta, \mu) | \theta : \mathbb{R} \to \mathbb{R}_\theta, \mu : \mathbb{R} \to \mathbb{R}_\mu\}$ and the pair (θ, μ) satisfies the reformulated model equations. The behavior $\mathfrak{B}'$ contains $\mathfrak{B}$, giving a linear, but θ-dependent description of Q. The reformulated system represents an LPV system. 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.

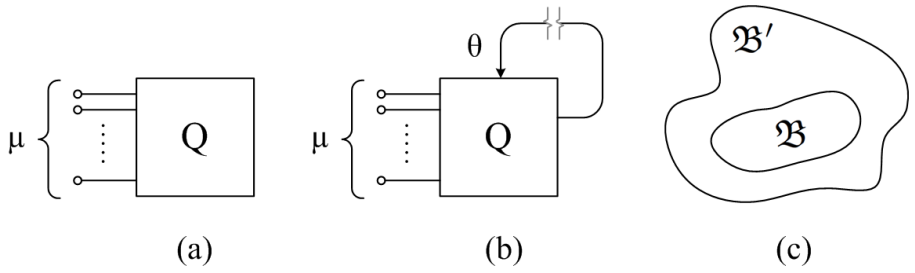

Figure 1.1: (a) Original nonlinear system representation. (b) LPV model representation. (c) Resulting behaviors (set of signal trajectories as the solution of the corresponding dynamical model).

Variables θ might even be free variables with respect to Q; however, often the auxiliary variables depend on other signals. In such a case, the resulting system is often referred to as a quasi-LPV system [1], but, in principle, the LPV framework and control synthesis problem remains invariant; only the representation of the nonlinear behavior becomes conservative, as indicated in Figure 1.1.

A typical LPV *state-space* (LPV-SS) model can be described as

$$\begin{aligned} qx_t &= A(\theta_t)x_t + B(\theta_t)u_t, \\ y_t &= C(\theta_t)x_t + D(\theta_t)u_t, \end{aligned} \tag{1.1}$$

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. Operator q denotes the differential operator for continuous-time systems, and forward shift operator for discrete-time systems, respectively. The state-space matrices are functions of the time-varying parameters $\theta_t \in \mathbb{R}_\theta,\ \mathbb{R}_\theta \subseteq \mathbb{R}^l$, commonly known as the scheduling variables. The scheduling variables are in turn continuous functions of measurable signals $\mu_t \in \mathbb{R}_\mu,\ \mathbb{R}_\mu \subseteq \mathbb{R}^s$, available from the system. This dependence can be written as

$$\theta_t = p(\mu_t), \quad p:\ \mathbb{R}^s \to \mathbb{R}^l. \tag{1.2}$$

where $s, l \in \mathbb{Z}_+$. In the practice of control engineering, there are many applications that require the design of a single controller that can guarantee stability and a transient response performance

for all operating conditions of the plant. A most common approach taken in these situations is that of gain-scheduling. In the context of LPV models, one or more scheduling variables can be used to parameterize the space of operating points of the plant. For each operating point within this space, the gain-scheduling approach will give a linear system. A similar parameterized controller can then be designed that fulfills the required closed-loop objectives at each operating point and an acceptable behavior while switching between multiple operating points [2]. A survey of different LPV gain-scheduled controller synthesis methods is given in detail in [3], [4], and [5].

Owing to this attractive property of the LPV models that retains the linearity structure between the inputs and outputs of the nonlinear dynamical system and can schedule the system based on scheduling variables that parameterize the system, it is not surprising that LPV identification and control has attracted noticeable attention in the recent past, *e.g.*, among others, see [6–9]. LPV models have found a wide variety of applications in the real world, ranging from aircrafts [10], helicopter dynamics [11], after-treatment systems [12], wind turbines [13–15], automobiles [16], servo systems [17], hypersonic vehicles [18], web service systems [19], mechatronic systems [20], aerospace applications [21], high purity distillation column [22], robot control [23], chaos synchronization [24], motion platforms [25], flight control [26], compressor [27], biomedical applications [28], and chemical processes [29], among others.

Naturally, a lot of research effort geared towards developing high fidelity LPV models has been seen in the modeling and control community in the recent past, from converting first-principles physics-based nonlinear lumped-parameter models into an LPV form, to developing LPV models purely from experimental data. In the following section, we take a look at the existing state of the art in the LPV identification and modeling literature pertaining to LPV system identification and model reduction.

1.2 Literature survey on LPV system identification

LPV modeling and system identification has a rich set of open problems that has challenged the researchers in the control systems community recently. The primary motivation behind LPV mod-

eling is to represent the nonlinear dynamic system using a *linear-in-input-and-output* model that has a dependency on time-varying parameters. This gives the advantage of being able to apply linear control synthesis techniques to nonlinear systems. The implication of this is that one can extend LTI control synthesis methods to LPV systems instead of going through tedious nonlinear control design procedures. However, to be able to design an LPV controller, one needs to have a "good" LPV model. Such a model has: (a) low model order and low number of scheduling variables, and (b) good prediction capability.

Aspect (a) pertains to having a "reduced" LPV model. 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 [30]. 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 [31–33]. 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 [33]. In particular, reduction of model order can result in an increased complexity, while reduction of dependency is often available via the introduction of extra state variables [33].

In [32], the authors proposed LPV model reduction, both in the sense of model order reduction, as well as reduction of the scheduling dependencies. The authors propose an LPV Ho-Kalman algorithm via imposing a sparsity expectation on the model to be reduced. The authors demonstrate their reduction algorithm on a mass-spring-damper system model. In [30], the authors demonstrated the use of *principal component analysis* (PCA) for finding tighter scheduling regions to represent the same underlying dynamical behavior in the LPV model. In [34], a model order reduction algorithm for non-stationary LPV models is presented. The authors present their technique as an extension of balanced truncation procedure.

Aspect (b) pertains to obtaining LPV models that can predict the outputs of the nonlinear dynamical systems under varying operating conditions. While models based on physics can be written into an LPV form, often these models may not be available, or might be too complex to be used for LPV controller synthesis. This is where "data-driven" models have shown immense potential in their use for LPV controller design. The LPV system identification community has been actively focusing on developing efficient methods of identifying LPV models using system-generated data.

LPV model identification has attracted a lot of attention within the system identification community in the recent past. Significant progress on the identification of LPV systems with *input-output* models (LPV-IO) has been achieved. Bamieh and Giarré developed a *least mean squares* (LMS) and *recursive least-squares* (RLS)-based algorithm for identification of discrete-time LPV-IO models in [35]. More recently, Tóth *et al*. have developed: discrete-time LPV-IO identification methods using *least-squares support vector machines* (LS-SVM) in [36]; an *instrumental variables* (IV) method to cater to generic noise conditions [37]; an IV-based method for LPV identification in the closed-loop [38]; a refined-IV method for direct identification of continuous-time LPV-IO models [39]; an IV-based scheme for LPV Box-Jenkins models [40]; and an LS-SVM scheme to identify LPV-IO models with noise-corrupted scheduling variables measurements [41], among several others. While these, and several other recent efforts exist for the identification of LPV-IO models, identification of LPV-SS model remains challenging with several open problems still unresolved.

The main streams of LPV control synthesis approaches in the literature are derived from LPV-SS representations. However, the bulk of discrete-time LPV identification and modeling is often carried out under an IO structure. Therefore, a possible approach would be to transform available LPV-IO models to LPV-SS form. However, such a transformation is complicated as the conversion to equivalent SS models often results in dynamic dependence of the state-space matrices on the scheduling variables while approximative "realizations" deform the dynamical relations between the inputs and outputs, often leading to high output errors [42]. Allowing for such a dynamic

dependence increases the complexity of the transformed LPV-SS model thereby making controller synthesis more difficult or even computationally infeasible. It is for this reason that LPV state-space models directly identified from input and output data are of prime importance.

There are two approaches researchers have taken for LPV identification, including LPV-SS identification. A local approach is that in which so-called "frozen" LTI models are identified for fixed values of scheduling variables and interpolation is performed to find a model at any other measured value of the scheduling variable. Examples of local LPV-SS identification include, but are not limited to [20, 43–46]. The other, global approach, is that of identifying the functional dependency of the state-space matrices on the scheduling variables.

Broadly speaking, global LPV-SS identification methods can be categorized into parametric and nonparametric methods. In parametric identification of LPV models, the assumption is made that the scheduling dependencies of the model coefficients are known *a priori* [35]. However, in practice, selecting adequate functions to parameterize these dependencies is a non-trivial task where often one tries to include a wide array of basis functions to ensure that the process dynamics are captured. This often leads to over-parametrization of the model coefficients [47], causing a large variance in the estimates. On the other hand, an inappropriate selection of these functions causes structural bias [36].

Examples of parametric LPV-SS identification include different variants of *subspace identification methods* (SIM) extended from LTI systems to LPV and bilinear models with affine parameter dependence. Verhaegen *et al.* first proposed a method based on gradient search in the scheduling parameter space in [48]. The said method involved solving a nonlinear optimization problem; the authors claimed that using subpace methods for LPV systems could ensure starting with a viable initial condition. Later, in [49], the authors proposed a subspace-based method for multivariable LPV-SS models. This work was limited to LPV models with affine parameter dependence on the LPV-SS matrices. The method usually suffers from the so-called *curse of dimensionality* problem; the authors proposed using only dominant rows of the data matrices for identification, in order to overcome this curse. The authors later applied an extension of this work on the problem of

identifying rotor dynamics of a helicopter in [11]. Kernels were later introduced to cope with the curse of dimensionality in [50]. The authors claimed that by relying only on computations with kernel matrices, the computation complexity of the earlier-proposed subspace method does not grow exponentially with the number of rows in the data matrix. Thus, with the introduction of kernel methods, the limitation of high computational complexity was addressed, making the method more suited to higher order systems. An improved subspace-based parametric identification method for LPV-SS models was proposed in [51]; the proposed method made use of periodically varying scheduling variables during the course of the identification experiment. The proposed method emphasized on obtaining an observability matrix that is defined with respect to the same state basis. However, once identified, the model was claimed to be valid for non-periodic sequences of scheduling variables as well. Predictor-based subspace identification (PBSID) was proposed in [52]. This method was yet another extension of LPV subspace-based identification method; it introduced a factorization making it possible to form predictors based on past inputs, outputs, and scheduling variables data. A similar predictor-based method was proposed and presented for open, as well as, closed-loop identification of LPV and bilinear systems in [53]. In [54], LPV-SS model with affine dependence was formulated as a generalized recurrent neural network model first proposed in [55]. A novel indirect technique for closed-loop identification was proposed; the identified model was later trasnformed into a quasi-LPV model. Validation was performed on an arm-driven inverted pendulum. All of the above methods pertain to systems that can be modeled with affine parameter-dependence, and fall under the category of parametric identification; these are usually suitable for low-dimensional cases. For an overview of other LPV-SS identification schemes, the interested reader is referred to [56].

An alternative approach with an attractive bias-variance trade-off is to obtain a nonparametric reconstruction of the scheduling dependencies in LPV models. Kernel-based nonparametric identification techniques have demonstrated encouraging results for LPV-IO models in [36, 41, 47], among others; however, very few nonparametric methods for state-space model structures have been reported. A mixed parametric method based on LS-SVM was proposed recently in [57]. In

this work, the state matrix A is described by a parametric model, while the state-readout matrix C is described by a nonparametric one. The problem of selecting basis functions therefore is solved only partially. Additionally, the work [57] focuses only on *single-input single-output* (SISO) LPV-SS models.

1.3 Dissertation aims and organization

The aim of this dissertation is to develop new algorithms, using tools from *machine learning* (ML), to address the problems of model reduction and identification in LPV-SS models. Specifically, we look at the problem of reducing the scheduling space in LPV-SS models in an efficient manner using kernelized ML as well as *multivariate analysis* (MVA) tools. We also compare the proposed methods with the existing method of model reduction that seeks to reduce the scheduling space. We further propose in this dissertation, nonparametric algorithms for identification of discrete-time LPV-SS models using kernelized LS-SVM. The proposed methods aim to identify LPV-SS models directly from input, output, and scheduling variables data, both, with and without, the availability of states measurements. We propose a convex optimization problem that depends only on tuning kernel hyper-parameters to unlock the functional dependencies of the elements of LPV-SS matrices on the scheduling variables. The use of MVA for state estimation is also explored. The dissertation is organized as follows: Chapter 2 proposed a method for model reduction in LPV-SS models using a kernelized PCA approach. The proposed method develops an iterative scheme to estimate the pre-image of the reduced scheduling variable in order to estimate the original LPV model from the reduced model. Chapter 3 proposes a method to overcome the need for solving an iterative nonlinear optimization procedure as proposed in the previous chapter, and proposes a much more efficient model reduction technique that makes use of kernelized MVA tools, yet gives us a reduced *affine-in-parameters* LPV-SS model. This makes the model ideal for LPV controller synthesis purpose. Chapter 4 introduces an LS-SVM-based scheme for identification of discrete-time LPV-SS models and considers the case study of identifying the model of a chemical process using the proposed method. In Chapter 5, the identification scheme of the previous chapter is

extended to a more generic noise setting, giving us the ability to handle colored noise in LPV-SS identification. Chapter 6 extends the proposed identification algorithm to the case where state measurements are not directly available. This chapter proposes an algorithm for state estimation in LPV-SS models when only the measurements of inputs, outputs, and scheduling variables data are available. Finally, concluding remarks about the contribution of the dissertation and possible future tracks of research are laid out in Chapter 7.

CHAPTER 2

PARAMETER SET-MAPPING USING KERNEL-BASED PCA FOR LINEAR PARAMETER-VARYING SYSTEMS [1]

[1] Syed Z. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin: Parameter Set-mapping using Kernel-based PCA for Linear Parameter-Varying Systems. 2014. *Proc. of the 13th European Control Conference*, Strasbourg, France: 2744-2749.

ABSTRACT

This paper proposes a method for reduction of scheduling dependency in linear parameter-varying (LPV) systems. In particular, both the dimension of the scheduling variable and the corresponding scheduling region are shrunk using kernel-based principal component analysis (PCA). Kernel PCA corresponds to linear PCA that is performed in a high-dimensional feature space, allowing the extension of linear PCA to nonlinear dimensionality reduction. It, hence, enables the reduction of complicated coefficient dependencies which cannot be simplified in a linear subspace, giving kernel PCA an advantage over other linear techniques. This corresponds to mapping the original scheduling variables to a set of lower dimensional variables via a nonlinear mapping. However, to recover the original coefficient functions of the model, this nonlinear mapping is needed to be inverted. Such an inversion is not straightforward. The reduced scheduling variables are a nonlinear expansion of the original scheduling variables into a high-dimensional feature space, an inverse mapping for which is not available. Therefore, we cannot generally assert that such an expansion has a "pre-image" in the original scheduling region. While certain pre-image approximation algorithms are found in the literature for Gaussian kernel-based PCA, we aim to generalize the pre-image estimation algorithm to other commonly used kernels, and formulate an iterative pre-image estimation rule. Finally, we consider the case study of a physical system described by an LPV model and compare the performance of linear and kernel PCA-based LPV model reduction.

2.1 INTRODUCTION

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. PCA is an eigenvector-based multivariate data analysis technique. The extracted variables are called principal components and are arranged in descending order of their variance in the data, measured by their corresponding eigenvalues. This means that the data components with very small variance can be neglected without losing much useful information, a property which makes PCA an extremely attractive option for dimensionality reduction purpose [58].

The ability of PCA to reduce the data dimension makes it ideal for tightening the scheduling region in linear parameter-varying (LPV) models. LPV systems are a class of systems, in which nonlinear models can be represented as a linear dynamic relation of the input and output variables, where this relation is dependent on a measurable time-varying signal, the so called scheduling variable, which expresses the varying operating conditions of the system. This allows one to apply linear optimal control techniques to nonlinear systems represented by LPV models. However, in practice, LPV controller design often encounters significant difficulties due to the high number of scheduling variables and conservatism and overbouding in the scheduling region [30]. For instance, in polytopic LPV systems, the complexity of controller synthesis has an *exponential* dependence on the number of scheduling variables. Hence, a large number of scheduling signals results in a high computational complexity for controller synthesis. Due to this, one of the objectives in deriving an LPV model for even a highly complex system is to limit the number of scheduling parameters to a very few as described in [31, 33, 59]. PCA has proved to be an effective tool for solving this problem and a handful of papers have successfully demonstrated the use of PCA for finding tighter scheduling regions for LPV systems (see [30], [60]).

With the emergence of kernel-based techniques, a new avenue of data processing appeared in the last decade. Kernels are nonlinear functions that enable us to perform linear operations in high-dimensional feature spaces, where it is easier to separate components in the data. These nonlinear functions operate in the original data space and do not require mapping of the original data to

the feature space, where the actual extraction takes place. This so called *kernel trick* has made component extraction much more efficient and realizable [61]. The choice of various kernels has further increased the flexibility of exploring different regions of the data more thoroughly. It is thus natural to investigate the use of kernel-based PCA for attaining efficient LPV models with reduced scheduling region.

In this paper, we explore the use of kernel-based PCA for LPV model reduction. It should be noted that model reduction in the LPV case refers to both reduction of the number of state variables (model order) and scheduling dependency as these facets of complexity are strongly related [33]. In particular, reduction of state order often results in an increased complexity, while reduction of dependency is often available via the introduction of extra state variables. Here, we address the problem of reduction of the complexity of the dependency with a proposed kernel PCA approach while the current state order is preserved. We analyze the advantages of kernel-based PCA over linear PCA, as well as the difficulties associated with it in obtaining a pre-image of the reduced parameters in the feature space. The paper is arranged as follows: Section 2.2 gives an introduction to LPV systems and formulates the problem of interest. Section 2.3 revisits the use of PCA for LPV systems. In Section 2.4, kernel-based PCA is applied to determine a tighter LPV scheduling region, and the mathematical advantages and pitfalls are explored. Section 2.5 considers a practical example of a mechanical system approximated by kernel-based PCA reduced LPV model. Some concluding remarks are made in Section 2.6. Here, we find it important to mention the use of subscripts and superscripts for variable notation used in this paper. Unless otherwise specified, for any given vector $\beta \in \mathbb{R}^n$, we use β_i^j to denote the j^{th} component of the i^{th} measurement vector β_i.

2.2 Preliminaries and Problem Formulation

Consider a nonlinear system Q shown in Figure 2.1. The system describes (possibly) nonlinear dynamic relation between measurable signals $\mu : \mathbb{R} \to \mathbb{R}^s$. Let $\mathfrak{B}$ be the set of all trajectories of μ compatible with Q. By introducing an auxiliary variable θ, one can reformulate the system representation of Q as shown in Figure 2.1 (b), where given the trajectory of θ, all relations between

μ are linear. By applying this reformulation of a disconnected θ and assuming that all trajectories of θ are allowed, the reformulated system will form a possible behavior $\mathfrak{B}'$. This behavior $\mathfrak{B}'$ will contain $\mathfrak{B}$, giving us a linear, but θ-dependent description of Q. The reformulated system represents an LPV system. This enables us to use several linear control techniques and convex controller synthesis for the nonlinear system described by an LPV representation. There can be several diffferent relations between the scheduling variable θ and the original variables μ. Variabale θ might be a free variable w.r.t. Q, However, often the auxillary variable depends on other signals. In such a case, the resulting system is often referred as a quasi parameter-varying system [1].

LPV systems in continuous-time are often described by a state-space representation as follows:

$$\begin{aligned} \dot{x}(t) &= A(\theta(t))x(t) + B(\theta(t))u(t), \\ y(t) &= C(\theta(t))x(t) + D(\theta(t))u(t), \end{aligned} \tag{2.1}$$

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 state, input, and output variables at time t, respectively. The system matrix, as well as the input, output and feedthrough matrices are continuous functions of time-varying parameters $\theta(t) \in \mathbb{R}^l$, commonly known as the scheduling variables. The scheduling parameters are in turn a continuous function of measurable signals $\mu(t) \in \mathbb{R}^s$, available from the system. This dependence can be written as

$$\theta(t) = p(\mu(t)), \quad p: \mathbb{R}^s \to \mathbb{R}^l. \tag{2.2}$$

The LPV system is considered affine in the scheduling parameters if

$$Q(\theta) = \sum_{i=1}^{l} \theta^i Q_i, \tag{2.3}$$

where $Q(\theta)$ is a more compact representation of the system given by

$$Q(\theta) = \begin{bmatrix} A(\theta) & B(\theta) \\ C(\theta) & D(\theta) \end{bmatrix}. \tag{2.4}$$

Now, consider the compact set $R_\theta \subset \mathbb{R}^l : \theta(t) \in R_\theta, \forall t > 0$ defined by vertices

$$R_\theta := \mathrm{Co}\{\theta_{v1} \cdots \theta_{vN}\}, \tag{2.5}$$

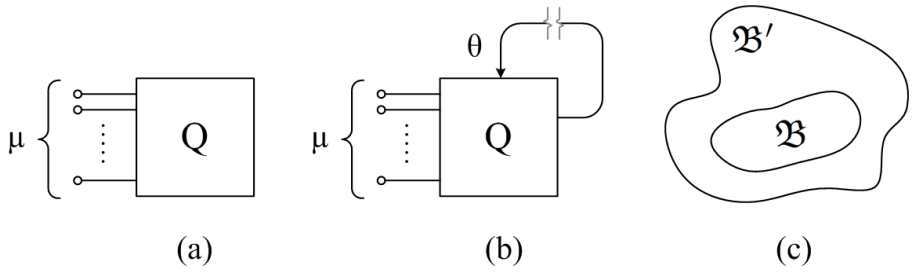

Figure 2.1: (a) Original nonlinear system representation. (b) LPV representation. (c) Resulting beaviors

where $N = 2^l$ defines the number of vertices in the scheduling region and Co denotes minimal convex hull. Given the fact that θ can be obtained by a convex combination of vertices θ_{vi}, and that $Q(\theta)$ depends affinely on the scheduling parameters, it follows that the system can be represented by a linear combination of multiple LTI systems at the vertices. Such a representation/system is referred to as a polytopic LPV system.

Given the system (2.1), the problem of LPV model reduction can be described as follows: Find a mapping defined by

$$\rho(t) = q(\mu(t)), \quad q : \mathbb{R}^s \to \mathbb{R}^m, \tag{2.6}$$

where $m \leq l$, such that the system matrices in

$$\begin{aligned} \dot{x}(t) &= \tilde{A}(\rho(t))x(t) + \tilde{B}(\rho(t))u(t), \\ y(t) &= \tilde{C}(\rho(t))x(t) + \tilde{D}(\rho(t))u(t), \end{aligned} \tag{2.7}$$

approximate (2.1) sufficiently well. We use kernel-based PCA to find a tighter scheduling region, and seek to find a relatively small m, that can approximate the original system with the one in (2.7), without incurring a significant loss of useful information in the data.

2.3 PCA FOR LPV MODELING

Before formulating a kernel-based PCA algorithm for parameter reduction, we quickly revisit the standard linear PCA as applied to the LPV model reduction problem. Basic literature on PCA can be found in detail in the literature in several manuscripts. Here, we refer the reader to [62]. To apply PCA to LPV scheduling variable data, one first needs to generate and collect data by means of measurement or simulation, such that the data covers all regions of operation within the operating range. Given the LPV system (2.1) and assuming that the scheduling signals have been sampled at time instants $t = 1, 2, \cdots, n$, scheduling parameters $\theta_i \in \mathbb{R}^l$ for $i = 1, 2, \cdots, n$ are computed and represented by the following $l \times n$ matrix:

$$\begin{aligned} \Theta &= \begin{bmatrix} \theta(1) & \cdots & \theta(n) \end{bmatrix}, \\ &= \begin{bmatrix} p(\mu(1)) & \cdots & p(\mu(n)) \end{bmatrix}, \end{aligned} \tag{2.8}$$

PCA can be applied either by performing a singular value decomposition (SVD) on the data matrix Θ, or by solving an eigenvalue problem for the covariance matrix $\Theta^{\mathrm{T}}\Theta$. Here, we describe the procedure based on eigenvalue decomposition of the covariance matrix in order to maintain uniformity with the kernel version of PCA in the next section. The covariance matrix is given by

$$C = \Theta_c^{\top}\Theta_c,$$

where $\Theta_c = \mathcal{N}(\Theta) = \Theta - \Theta_{\mathrm{mean}}$ is the data centered around the origin. We then solve an eigenvalue problem for the covariance matrix C, such that $Cv_i = \lambda_i v_i$, where λ_i and v_i are the i^{th} eigenvalue and eigenvector, respectively. The eigenvectors are then sorted in descending order of their corresponding non-zero eigenvalues, and the m principal components for any test point $\theta(t)$ at a given time t are extracted using

$$\rho(t) = q(\mu(t)) = V_m^{\top} p(\mu(t)) = V_m^{\top}\theta(t), \tag{2.9}$$

where V_m denotes an $l \times m$ matrix whose columns contain the first m significant eigenvectors associated with the m significant eigenvalues. The approximation of the actual parameter $\hat{\theta}(t)$

corresponding to $\rho(t)$ can be easily computed as

$$\hat{\theta}(t) = \mathcal{N}^{-1}(V_m \rho(t)), \tag{2.10}$$

where $\mathcal{N}^{-1}(v_m \rho(t)) = v_m \rho(t) + \Theta_{mean}$ denotes scaling back of the data by adding the mean value of Θ along each dimension.

2.4 KERNEL PCA FOR LPV MODELING

Kernel-based PCA, more simply known as kernel PCA, is an extension of the traditional PCA approach, in which linear dot-product operation is performed, albeit in a higher dimensional feature space [58]. Due to the high dimension of the feature space, separation of features or components in the data is much easily realizable. The beauty 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 like feature extraction in facial recognition [63] and denoising [64], among several others. In LPV systems, this can lead to reduced over-parametrization by reducing scheduling variables more effectively.

Let us assume that the scheduling variables of the LPV system (2.1) are mapped into the feature space as $\Phi(\theta_1), \Phi(\theta_2), \cdots, \Phi(\theta_n)$. We also assume that the mapped parameters are centered, *i.e.*, $\sum_{j=1}^{n} \Phi(\theta_j) = 0$. Since it is not possible to obtain the mean of data we do not explicitly calculate, we will assume at this point that the data is centered in the feature space and explore an implicit alternative to centering the data, later in this section. To perform traditional PCA on this data, we derive the covariance matrix as follows:

$$C = \frac{1}{n} \sum_{j=1}^{n} \Phi(\theta_j) \Phi^{\top}(\theta_j). \tag{2.11}$$

In order to select the principal components, the relation $\lambda v = Cv$ should hold. We can therefore consider the following:

$$\lambda(\Phi(\theta_j) \cdot v) = (\Phi(\theta_j) \cdot Cv), \quad \forall\ j = 1, \cdots n, \tag{2.12}$$

where $(a \cdot b)$ signifies dot product given by $a^{\mathrm{T}}b$, and that there exists coefficients α_j for $j = 1, \cdots, n$ such that

$$v = \sum_{j=1}^{n} \alpha_j \Phi(\theta_j). \tag{2.13}$$

This means that the eigenvectors of the matrix C belong to the span of $\Phi(\theta_j)$ for $j = 1, \cdots, n$. Combining (2.12) and (2.13), we obtain

$$\lambda \sum_{j=1}^{n} \alpha_j (\Phi(\theta_i) \cdot \Phi(\theta_j)) = \frac{1}{n} \sum_{j=1}^{n} \alpha_j \left((\Phi(\theta_i) \cdot \sum_{u=1}^{n} \Phi(\theta_u) \right) (\Phi(\theta_u) \cdot \Phi(\theta_j)) \tag{2.14}$$

for all $i = 1, \cdots, n$. The eigenvalue problem in (2.14) only involves dot products of mapped shaped vectors in the feature space and does not explicitly require computing $\Phi(\cdot)$. Now, we define an $n \times n$ matrix K as follows:

$$K_{ij} = (\Phi(\theta_i) \cdot \Phi(\theta_j)) = k(\theta_i, \theta_j), \tag{2.15}$$

where matrix $K \in \mathbb{R}^{n \times n}$ is known as the Gram matrix, or kernel matrix, and k is the nonlinear kernel function. Manipulating the right and left hand sides of (2.14) as given in [65] leaves us with the final problem of finding the solutions to

$$n\lambda\alpha = K\alpha \tag{2.16}$$

for non-zero eigenvalues. The solutions α_r belonging to non-zero eigenvalues need to be normalized by requiring that the corresponding vectors in feature space are normalized. This translates to the condition $(v_r \cdot v_r) = 1$. Using (2.13), (2.15) and (2.16), we obtain

$$1 = \sum_{i,j=1}^{n} \alpha_r^i \alpha_r^j (\Phi(\theta_i) \cdot \Phi(\theta_j)) = (\alpha_r \cdot K\alpha_r) = \lambda_r (\alpha_r \cdot \alpha_r), \tag{2.17}$$

where α_r^i denotes the i^{th} component of α_r. This translates to dividing each eigenvector α_r of K by $\sqrt{\lambda_r}$ in order to normalize it. Lastly, for principal component extraction, we compute the projections of the image of a test-point $\theta(t)$ at a given time t onto the eigenvectors v_r in the feature

space as

$$\rho^r(t) = (v_r \cdot \Phi(\theta(t))) = \sum_{j=1}^{n} \alpha_r^j (\Phi(\theta_j) \cdot \Phi(\theta(t))) \\ = \sum_{j=1}^{n} \alpha_r^j \; k(\theta_j, \theta(t)), \tag{2.18}$$

where $\rho^r(t)$ is the r^{th} component of the said projection of $\Phi(\theta(t))$ on v_r. It is noteworthy to mention that the above equation does not explicitly require the computation of $\Phi(\theta_j)$. What is needed is only the dot product in feature space. This can be computed by using nonlinear kernel functions k. This way, the *kernel trick* allows us to project the image of a point onto the eigenvectors in the feature space without explicitly mapping the data in the feature space.

Finally, it can be recalled that the data was initially, and rather simplistically, assumed to be centered in the feature space. This, however, is not always the case, and care must be taken to center the data. However, we cannot in general center the scheduling variables data in the feature space, since we cannot compute the mean of the data that we explicitly do not calculate. We, therefore, work around this issue and obtain a centered Gram matrix as [58]

$$\tilde{K}_{ij} = K - 1_n K - K 1_n + 1_n K 1_n, \tag{2.19}$$

where $1_n \in \mathbb{R}^{n \times n}$ with each entry being $1/n$. Kernel functions can be chosen from a variety of different functions that exist in the literature, and have been used for classification, feature extraction, and other applications [66]. These include the polynomial kernel given as

$$k(\theta_i, \theta_j) = (\theta_i \cdot \theta_j + 1)^d, \tag{2.20}$$

the radial basis function kernel given as

$$k(\theta_i, \theta_j) = e^{-\frac{|| \theta_i - \theta_j ||^2}{\sigma^2}}, \tag{2.21}$$

and the sigmoid kernel given as

$$k(\theta_i, \theta_j) = \tanh(a \; \theta_i \cdot \theta_j + b), \tag{2.22}$$

where d, σ, a, and b in (2.20), (2.21), and (2.22) refer to the degree of polynomial, the spread of radial basis function, and the slope and bias in sigmoid functions, respectively. Beside these kernels, there are a handful of other kernel functions, and the choice of a function would depend highly on the nonlinearities associated with a given LPV system. According to Mercer's theorem, if a kernel function satisfies the conditions of being a continuous function of positive integral operator, then all mathematical and statistical properties of linear PCA are retained by kernel PCA as well [58]. This essentially means that kernel PCA is in fact performing linear PCA in the feature space. The main advantage of kernel PCA lies in its ability to extract nonlinear components or features in a data based on the choice of a particular kernel function, without requiring any nonlinear optimization. Rather, it does so by simply solving an eigenvalue problem.

2.4.1 Accuracy of the estimated LPV model

Using linear or kernel PCA, one can reduce the LPV model to have affine dependence on lesser number of scheduling variables. The accuracy of the estimated model can be gauged from the fraction of total data variation, calculated as

$$a(m) = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{v} \lambda_i}, \tag{2.23}$$

where m is the number of reduced parameters, and λ_i denotes the i^{th} eigenvalue of the kernel matrix $\tilde{K}$. In case of linear PCA, the eigenvalue is that of the covariance matrix [67]. Since the kernel and covariance matrices are square matrices of dimensions n and l, respectively, v is consequently equal to n and l for kernel and linear PCA, respectively. Therefore, by choosing the number of scheduling parameters, a trade-off can be achieved between accuracy and complexity.

2.4.2 The preimaging problem

In the LPV model reduction problem, it is important that we are able to get the original LPV variables back by using the reduced parameters. This is because of the fact that for LPV control design, the controller matrices are scheduled based on the reduced parameters. The control matrices however depend on the open-loop LPV system matrices which are a function of the original scheduling

variables. Therefore, one can also say that the original scheduling variables are required for calculating the new representation of the system. It is therefore of paramount importance that the reduced parameters are able to estimate the original LPV scheduling variables back. This gives us the motivation to solve the preimaging problem.

The kernel PCA suffers from an inability to reconstruct the original patterns in a straightforward way. So to say, given reduced parameter vector ρ_i extracted from θ_i using kernel PCA, there is no systematic way to reconstruct the original parameters θ_i, since feature space algorithms express their solutions as expansions in terms of mapped data points. Therefore, one cannot in general say that a preimage exists under the map Φ, for which $\Phi(\theta_i) = \Psi$. Moreover, this problem of finding the preimage might be unsolvable in the general case, since the preimage might not always exist [64]. Schölkopf *et al.* therefore argued in [64] that rather than finding the exact preimage, it is possible to find an estimate of the preimage. This is done by considering the fact that we may seek to approximate the preimage of a feature space expansion $\Psi = \sum_{i=1}^{n} \gamma_i \Phi(\theta_i)$ by its estimate $\Psi' = \beta\Phi(\tilde{\theta}_i)$, where $\tilde{\theta}$ denotes the approximated preimage. One can, therefore, attempt to minimize the distance between Ψ and Ψ'. Somewhat equivalently, one can minimize the distance between Ψ and the orthogonal projection of Ψ onto the span of $\Phi(\tilde{\theta})$, *i.e.*,

$$||\frac{(\Psi \cdot \Phi(\tilde{\theta}))}{(\Phi(\tilde{\theta}) \cdot \Phi(\tilde{\theta}))}\Phi(\tilde{\theta}) - \Psi||^2 = ||\Psi||^2 - \frac{(\Psi \cdot \Phi(\tilde{\theta}))^2}{(\Phi(\tilde{\theta}) \cdot \Phi(\tilde{\theta}))} \tag{2.24}$$

This can be achieved by maximizing

$$H(\tilde{\theta}) = \frac{(\Psi \cdot \Phi(\tilde{\theta}))^2}{(\Phi(\tilde{\theta}) \cdot \Phi(\tilde{\theta}))}, \tag{2.25}$$

and once the maximum is achieved, we set the variable β as $\beta = \Psi \cdot \Phi(\tilde{\theta}) \ /(\ \Phi(\tilde{\theta}) \cdot \Phi(\tilde{\theta}))$. To find the extremum, we solve

$$\begin{aligned} \nabla_{\tilde{\theta}} H(\tilde{\theta}) &= \nabla_{\tilde{\theta}} \frac{(\Psi \cdot \Phi(\tilde{\theta}))^2}{(\Phi(\tilde{\theta}) \cdot \Phi(\tilde{\theta}))} = 0 \\ &= \nabla_{\tilde{\theta}} \frac{(\sum_{i=1}^{n} \gamma_i k(\theta_i, \tilde{\theta}))^2}{k(\tilde{\theta}, \tilde{\theta})} = 0 \end{aligned} \tag{2.26}$$

Algorithm 1 Applying kernel PCA for LPV model reduction

Step 1: Generate a set of scheduling signals using measurement or simulation, covering the expected range of operation
Step 2: Compute the trajectories of the corresponding scheduling variables θ_i for $i = 1, \cdots, n$
Step 3: Compute matrix K using (2.15)
Step 4: Center the data in feature space to find $\tilde{K}$ using (2.19)
Step 5: Solve (2.16) by diagonalizing $\tilde{K}$
Step 6: Normalize the eigenvectors using (2.17)
Step 7: For online LPV control: For each set of observed scheduling variable $\theta(t)$ at time t, get the r^{th} component of reduced- dimension parameter $\rho^r(t)$ by using (2.18)
Step 8:
while $g(\tilde{\theta})$ is not minimized, **do**
Compute preimage $\tilde{\theta}(t)$ from $\rho(t)$ using (2.35)
end while

PREIMAGE FOR GAUSSIAN KERNELS

For the Gaussian kernel, since $k(\tilde{\theta}, \tilde{\theta}) = 1$, (2.26) becomes

$$\nabla_{\tilde{\theta}} \left(\sum_{i=1}^{n} \gamma_{\,i} e^{-\frac{||\theta_i - \tilde{\theta}||^2}{2\sigma^2}} \right)^2 = 0, \tag{2.27}$$

giving us

$$\tilde{\theta} = \frac{\sum_{i=1}^{n} \gamma_{\,i}\; k(\theta_i, \tilde{\theta}) \theta_i}{\sum_{i=1}^{n} \gamma_{\,i}\; k(\theta_i, \tilde{\theta})}. \tag{2.28}$$

PREIMAGE FOR POLYNOMIAL KERNELS

For polynomial kernel, (2.26) becomes

$$\nabla_{\tilde{\theta}} \frac{\left(\sum_{i=1}^{n} \gamma_{\,i}\; (\theta_i^{\mathrm{T}} \tilde{\theta} + 1)^d \right)^2}{(\tilde{\theta}^{\mathrm{T}} \tilde{\theta} + 1)^d} = 0, \tag{2.29}$$

giving us

$$\tilde{\theta} = \frac{\left(\sum_{i=1}^{n} \gamma_{\,i}\; k^{d-1}(\theta_i, \tilde{\theta}) \theta_i \right) k^d(\tilde{\theta}, \tilde{\theta})}{\left(\sum_{i=1}^{n} \gamma_{\,i}\; k^{d}(\theta_i, \tilde{\theta}) \theta_i \right) k^{d-1}(\tilde{\theta}, \tilde{\theta})}, \tag{2.30}$$

where $k^d(\cdot, \cdot)$ indicates a polynomial kernel function of degree d.

PREIMAGE FOR SIGMOID KERNELS

For the sigmoid kernel, (2.26) becomes

$$\nabla_{\tilde{\theta}} \frac{\left(\sum_{i=1}^{n} \gamma_i \ \tanh(a\ \theta_i^{\mathrm{T}} \tilde{\theta} + b)\right)^2}{\tanh(a\ \tilde{\theta}^{\mathrm{T}} \tilde{\theta} + b)} = 0, \tag{2.31}$$

giving us

$$\tilde{\theta} = \frac{\left(\sum_{i=1}^{n} \gamma_i \mathrm{sech}^2(a\ \theta_{\mathrm{i}}^{\mathrm{T}} \tilde{\theta} + \mathrm{b})\right) \theta_i \left(\tanh(a\ \tilde{\theta}^{\mathrm{T}} \tilde{\theta} + b)\right)}{\left(\sum_{i=1}^{n} \gamma_i \tanh(a\ \theta_i^{\mathrm{T}} \tilde{\theta} + b)\right) \left(\mathrm{sech}^2(a\ \tilde{\theta}^{\mathrm{T}} \tilde{\theta} + \mathrm{b})\right)}, \tag{2.32}$$

where $\gamma_i = \sum_{j=1}^{m} \rho^j \alpha_j^i$, with ρ^j being the j^{th} component of the reduced parameter vector ρ used to estimate the scheduling parameter [68]. Similar relations can be worked out for other kernels as well.

A GENERALIZED ITERATIVE UPDATE RULE

To formulate an iterative update equation for the estimate $\tilde{\theta}$, we can generalize (2.28), (2.30), and (2.32) as

$$\tilde{\theta} = f(\tilde{\theta}). \tag{2.33}$$

Defining a new function $g(\tilde{\theta})$,

$$g(\tilde{\theta}) = \tilde{\theta} - f(\tilde{\theta}) = 0, \tag{2.34}$$

we can reduce the problem at hand to finding the root $\tilde{\theta}$ of $g(\tilde{\theta})$ such that $g(\tilde{\theta}) = 0$. Note that $g(\tilde{\theta})$ is a vector-valued function of vector $\tilde{\theta}$. To find the root of $g(\tilde{\theta})$, one can use iterative root-finding algorithms like Newton's method. An alternative to root finding is to use a steepest descent method with variable or fixed step size, in order to minimize $g(\tilde{\theta})$. Steepest descent methods are similar to Newton's method, but they incorporate a small step-size in their convergence towards the solution in order to ensure stability by iteratively estimating the preimage $\tilde{\theta}$ according to the following equation:

$$\begin{aligned} h(k) &= -[J_g(\tilde{\theta}(k))]^{\top} g(\tilde{\theta}(k)), \\ \tilde{\theta}(k+1) &= \tilde{\theta}(k) + \eta h(k), \end{aligned} \tag{2.35}$$

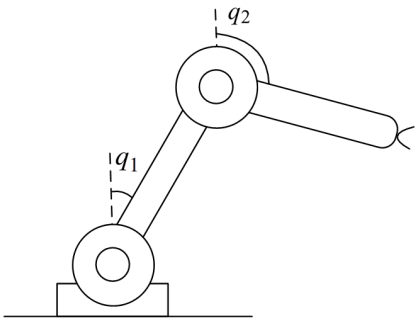

Figure 2.2: Schematic of 2-DoF robotic manipulator

where k is the iteration index, $g[\tilde{\theta}(k)]$ is as defined in (2.34), and $f(\tilde{\theta})$ would be formulated according to the kernel function in use. For Gaussian, polynomial, or sigmoid kernels, $f(\tilde{\theta})$ is as defined in (2.28), (2.30), or (2.32), respectively. Variable η is the step size and has to be chosen. A small η ensures stability but might take more iterations to converge, whereas a larger η might lead to instability. Step size can also be chosen adaptively at each iteration, but that requires an optimization routine for finding the optimal η at each iteration as follows:

$$\eta(k) = \underset{\eta}{\operatorname{argmin}} \frac{1}{2} g^{\top} g[\tilde{\theta}(k) + \eta(k) h(\tilde{\theta}(k)]. \tag{2.36}$$

Since Newton's method requires finding the derivative of $g(\tilde{\theta})$, which in this case would be the Jacobian matrix J_g, an alternative to avoiding the calculation of the Jacobian would be to use other approximate Quasi-Newton methods like Secant or Broyden's method. The idea behind Broyden's method is to avoid calculating the Jacobian, or its inverse, at each iteration. It should be noted that the success or failure of these methods are highly dependent on the nonlinearities in the data, the number of minima in the function, and the choice of the kernel function. For our case study given in the next section, we employed steepest descent method and obtained very good convergence of estimates with a manageable computational time. For the algorithm to converge, one requires a "decent" initial condition for $\tilde{\theta}$ and an initial estimate of the Jacobian. In case of numerical instabilities, which is rare, the algorithm simply requires a restart with different starting values. To

summarize, the proposed kernel PCA algorithm for dimensionality reduction in LPV systems is finally formulated in Algorithm 1.

2.5 NUMERICAL EXAMPLE

A dynamic model of a two-link planar robot is considered here with a schematic diagram of the robotic manipulator shown in Figure 2.2. Detailed nonlinear model and the associated parameters have been taken from [69]. The lower arm of the robot is known as the shoulder, while the upper part is simply known as the arm, which is attached to the actuator. The two joints are connected via a gear servo mechanism. Following the ideas in [30] and [69], the derived LPV model is presented as follows:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ cd\theta_3 & -be\theta_4 & \theta_5 & b\theta_6 \\ -bd\theta_7 & ae\theta_8 & \theta_9 & \theta_{10} \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ cnk_m\theta_1 & -bnk_m\theta_2 \\ -bnk_m\theta_2 & ank_m\theta_1 \end{bmatrix}, C = \begin{bmatrix} I & \emptyset \end{bmatrix}, D = \emptyset, \quad (2.37)$$

where

$$\begin{aligned} h &= ac - b^2\cos^2\Delta \\ \theta_1 &= \frac{g}{h}, \\ \theta_2 &= \frac{g\cos\Delta}{h}, \\ \theta_3 &= \frac{\mathrm{sinc}(q_1)}{h}, \\ \theta_4 &= \cos\Delta\frac{\mathrm{sinc}(q_2)}{h}, \\ \theta_5 &= \frac{(-b^2\sin\Delta\ \cos\Delta\dot{q_1} - (c + b\cos\Delta)f)}{h}, \\ \theta_6 &= \frac{-c\sin\Delta\dot{q_2} + \cos\Delta f}{h}, \\ \theta_7 &= \frac{\cos\Delta\mathrm{sinc}(q_1)}{h}, \\ \theta_8 &= \frac{\mathrm{sinc}(q_2)}{h}, \\ \theta_9 &= \frac{(ab\ \sin\Delta\dot{q_1} + f(a + b\cos\Delta))}{h}, \end{aligned}$$

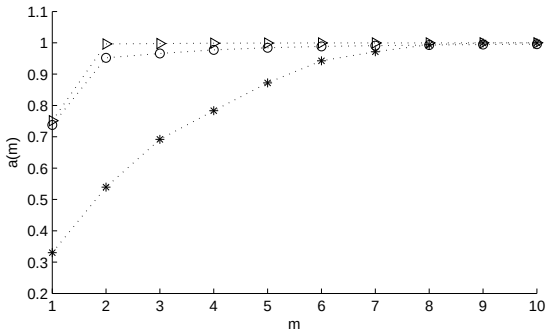

Figure 2.3: Accuracy of estimates (2.23) as a function of the number of scheduling parameters m for (*) linear PCA, (o) kernel PCA with polynomial kernel, and ($\triangleright$) kernel PCA with sigmoid kernel

$$\theta_{10} = \frac{(b^2 \sin\Delta \cos\Delta \dot{q_2} - af)}{h},$$

where $\cos(\Delta) = \cos(q_1 - q_2)$, $\sin(\Delta) = \sin(q_1 - q_2)$, and I and $\emptyset$ are identity and zero matrices of appropriate dimensions, respectively. The state vector is given by $x = [q_1\ q_2\ \dot{q_1}\ \dot{q_2}]^{\mathrm{T}}$. Angles of the two links with respect to the vertical reference make up the first two states, while angular velocities make up the other two states of the LPV model. This LPV model has a total of 10 scheduling parameters. The objective here is to reduce the number of scheduling parameters in order to reduce the over-parametrization in the given LPV model. The scheduling signals in this case are the states, hence, $s = 4$. Number of scheduling parameters is $l = 10$. We seek to find, using PCA, the smallest possible number of reduced parameters m, with $m \leq l$ that gives an LPV model that approximates the original model (2.37) well.

A set of trajectories is generated with open-loop simulation of the LPV system using sinusoidal inputs u and scheduling parameters are computed based on the states. We apply linear and kernel PCA on the scheduling parameters data. For the kenrel PCA, we choose a polynomial kernel of degree 12 and sigmoid kernel with $a = 0.5,\ b = -13$, based on trial and error. Gaussian kernel-based PCA provides an accuracy measure close to that of linear PCA with this problem, and hence is not discussed here. We then compare the total "variation" (2.23) of the approximated

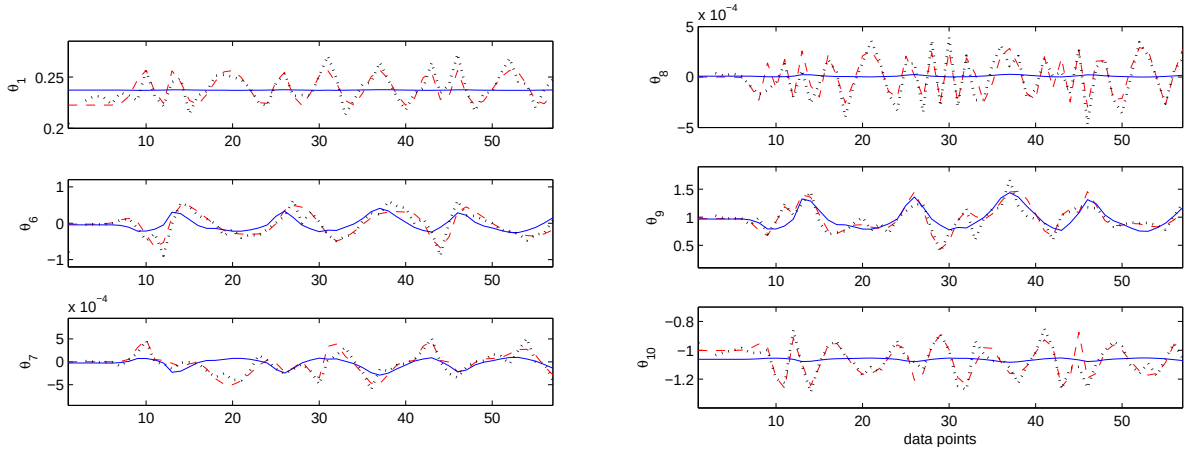

Figure 2.4: Scheduling parameters θ_i (red dashed line), PCA-based approximation $\hat{\theta}_i$ (blue solid line), and kernel PCA-based approximation $\tilde{\theta}$ (black dotted line) with $m = 1$.

LPV models as a function of the number of LPV variables, m. This is interpreted as the accuracy of the approximation, and is shown in Figure 2.3. As can be seen, kernel PCA for both kernels gives around 78% accuracy with one scheduling parameter, much higher than 33% accuracy of the linear PCA. From the initial measure of accuracy, we foresee that for the kernel PCA case, LPV model with one parameter might provide a good estimate of the actual system. Using two parameters should give an almost perfect estimate, without much loss of information. Since both polynomial and sigmoid kernels give an almost equal measure of accuracy, we present the results for polynomial kernel-based PCA only, since it performed comparatively better during preimage estimation. Having reduced the scheduling parameters from $l = 10$ to $m = 1$, we re-estimate the actual scheduling parameters using preimaging. We employ the steepest descent estimation of (2.35) with a small and fixed step size $\eta = 0.001$. Initial estimate is chosen randomly along each dimension, and preimaging algorithm is run. Results for a few parameters are shown in Figure 2.4. Other parameters show a very similar trend to those plotted here. These results show a good fit between the actual scheduling parameters (dashed line), and those re-estimated by kernel PCA-

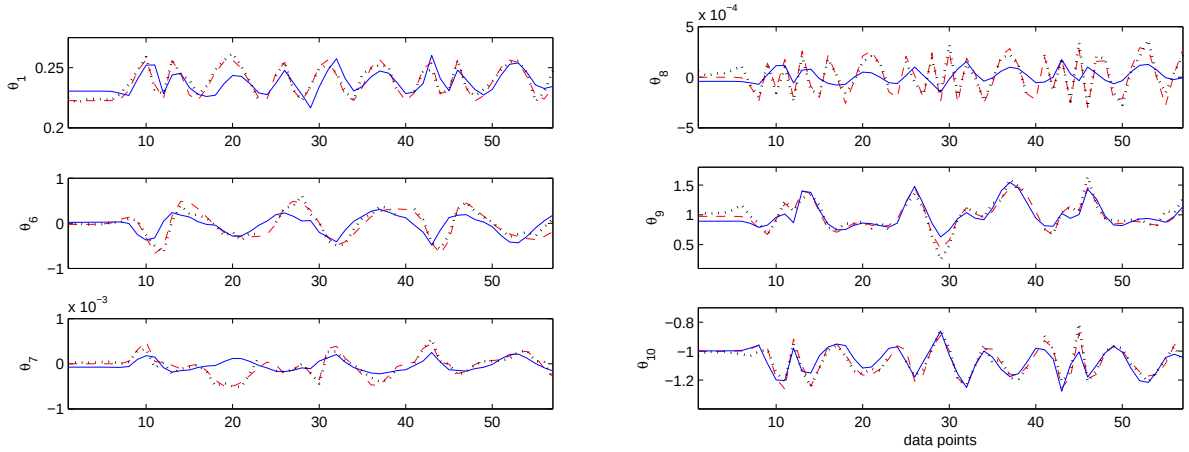

Figure 2.5: Scheduling parameters θ_i (red dashed line), PCA-based approximation $\hat{\theta}_i$ (blue solid line), and kernel PCA-based approximation $\tilde{\theta}$ (black dotted line) with $m = 2$.

based reduced data (dotted line). As observed, linear PCA (solid line) simply fails to estimate $\theta_1, \theta_7, \theta_8$, and θ_{10}, while showing some trend in the other parameters.

Using a different scheduling data set, we performed linear PCA and kernel PCA for $m = 2$. Linear PCA does perform better when we choose $m = 2$, as observed in the plots shown in Figure 2.5. Estimates for θ_1 and θ_{10} show marked improvements, while θ_7, θ_8 still do not re-estimate the parameters completely. Kernel PCA performs well with both, one and two, parameters.

2.6 Concluding Remarks

We have explored the use of kernel-based PCA for the purpose of dimensionality reduction of the scheduling variables in LPV representations. Reducing the number of scheduling parameters is a problem of paramount importance, since it directly translates to the reduction in computational complexity for LPV controller design. In polytopic LPV systems, the number of *linear matrix inequality* (LMI) constraints needed to be solved in order to perform LPV $\mathcal{H}_\infty$ controller synthesis is exponential in the dimension of the scheduling variables, since these conditions need to be satisfied at each vertex of the polytope. Though parameters mapped into feature space cannot be

systematically mapped back to the scheduling region as in the case of linear PCA, they can still be estimated efficiently using an iterative update rule derived in this paper. While an iterative approach to estimation is not as quick and straightforward as projecting the parameters on an eigenvector space, the improvement in dimensionality reduction makes it a worthy trade-off. As part of a future extension, an optimization problem can be designed that retains the affine structure of the reduced LPV model, despite the kernelized (nonlinear) mapping of the original scheduling variables into the feature space. Results in this paper provide encouraging insights into the use of kernel PCA for LPV dependency reduction. These results can be exploited to formulate a controller synthesis problem that is computationally less complex when compared to a similar synthesis based on the original full LPV model.

CHAPTER 3

A KERNEL-BASED PCA APPROACH TO MODEL REDUCTION OF LINEAR PARAMETER-VARYING SYSTEMS [1]

[1] Syed Z. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin: A Kernel-based PCA Approach to Model Reduction of Linear Parameter-varying Systems. 2016. *IEEE Trans. Control Systems Technology*. 24(5):1883-1891.

ABSTRACT

This paper presents a model reduction method for *linear parameter-varying* (LPV) systems using kernel-based *principal component analysis* (PCA). For LPV state-space 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.

3.1 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 [30]. 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 [32, 33]. 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 [33]. 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 [33]. 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 cor-

responding 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 [61]. 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 [12, 30]).

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, 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 [61]. 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 3.2 gives an introduction to LPV systems and formulates the problem of interest. In Section 3.3, 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 3.4 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.

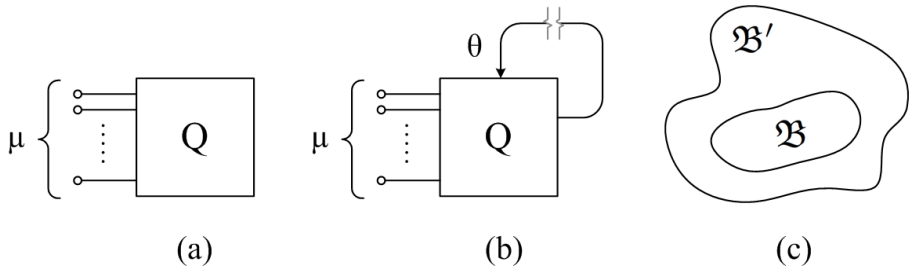

Figure 3.1: (a) Original nonlinear system representation. (b) LPV model representation. (c) Resulting behaviors (set of signal trajectories as the solution of the corresponding dynamical model).

3.2 Preliminaries and Problem Statement

Consider a continuous-time nonlinear system Q shown in Figure 3.1. 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 3.1(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 [1]. 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 3.1.

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

$$\begin{aligned} \dot{x}_t &= A(\theta_t)x_t + B(\theta_t)u_t, \\ y_t &= C(\theta_t)x_t + D(\theta_t)u_t, \end{aligned} \tag{3.1}$$

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

$$\theta_t = p(\mu_t), \quad p:\ \mathbb{R}^s \to \mathbb{R}^l. \tag{3.2}$$

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

$$Q(\theta_t) = Q_0 + \sum_{i=1}^{l} Q_i \theta_{t,i}, \tag{3.3}$$

where $\theta_{t,i}$ is the i^{th} entry of θ_t, and $Q(\theta_t)$ is a compact representation of the system $Q(\theta_t) = \begin{bmatrix} A(\theta_t) & B(\theta_t) \\ C(\theta_t) & D(\theta_t) \end{bmatrix}$. 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 $R_\theta := \mathrm{Co}\{\theta_{v_1} \cdots \theta_{v_N}\}$, 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

$$\theta_t = \sum_{i=1}^{N} \beta_i \theta_{v_i},\ \text{with } \beta_i \geq 0,\ \sum_{i=1}^{N} \beta_i = 1. \tag{3.4}$$

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 (3.3)-(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* LPV state-space representation [70]. The same applies to a desired reduced LPV model of (3.1). 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 [70]. Similar tractability can be achieved when the LPV representation is rationally dependent on the scheduling variables [71]. Therefore, given the system representation (3.1), with measurable signals μ_t, and scheduling variables defined in (3.2), the problem of LPV model reduction explored in this paper can be stated as follows: Find a mapping defined by

$$\rho_t = q(\mu_t), \quad q: \mathbb{R}^s \to \mathbb{R}^m, \tag{3.5}$$

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

$$\begin{aligned} \dot{\hat{x}}_t &= \tilde{A}(\rho_t)\hat{x}_t + \tilde{B}(\rho_t)u_t, \\ y_t &= \tilde{C}(\rho_t)\hat{x}_t + \tilde{D}(\rho_t)u_t, \end{aligned} \tag{3.6}$$

can accurately follow the trajectories of (3.1) 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.

3.3 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 (3.1), the scheduling variables $\theta_t \in \mathbb{R}^l$

are computed and collected as

$$\begin{aligned}\Theta &= \begin{bmatrix}\theta_1 & \cdots & \theta_n\end{bmatrix} \\ &= \begin{bmatrix}p(\mu_1) & \cdots & p(\mu_n)\end{bmatrix} \in \mathbb{R}^{l\times n}\end{aligned}$$

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 [72]. 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 [30] 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 [61]. 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 [64]. 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 [63] and denoising [64], 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 (3.1) 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

$$\bar{C} = \frac{1}{n}\sum_{j=1}^{n} \Phi(\theta_j)\Phi^\top(\theta_j). \tag{3.7}$$

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

$$\lambda(v \cdot \Phi(\theta_i)) = (\bar{C}v \cdot \Phi(\theta_i)), \quad \forall\, i = 1, \cdots, n, \tag{3.8}$$

where $(a \cdot b) = a^\top b$ denotes dot product. Note that (3.7) 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 (3.7) in (3.8), we get [61]

$$\lambda \sum_{w=1}^{n} \alpha_w (\Phi(\theta_w) \cdot \Phi(\theta_i)) = \frac{1}{n} \sum_{j=1}^{n} \sum_{w=1}^{n} \alpha_w \left(\Phi(\theta_j) \cdot \Phi(\theta_w)\right) \left(\Phi(\theta_j) \cdot \Phi(\theta_i)\right), \tag{3.9}$$

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

$$K_{ij} = (\Phi(\theta_i) \cdot \Phi(\theta_j)) = k(\theta_i, \theta_j), \tag{3.10}$$

where $k(\cdot, \cdot)$ is a nonlinear kernel function; later, we shall elaborate on the choice of these kernels. Substituting (3.10) in (3.9) 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 [61]. 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

$$\begin{aligned} \rho_{t,r} = (v_r \cdot \Phi(\theta_t)) &= \sum_{j=1}^{n} \alpha_{r,j} (\Phi(\theta_j) \cdot \Phi(\theta_t)) \\ &= \sum_{j=1}^{n} \alpha_{r,j}\; k(\theta_j, p(\mu_t)) = q(\mu_t), \end{aligned} \tag{3.11}$$

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. These include the *polynomial kernel* given as

$$k(\theta_i, \theta_j) = ((\theta_i \cdot \theta_j) + 1)^d, \tag{3.12}$$

the *radial basis function kernel* given as

$$k(\theta_i, \theta_j) = \exp\left(-\frac{|| \theta_i - \theta_j ||^2}{\sigma^2}\right), \tag{3.13}$$

and the *sigmoid kernel* given as

$$k(\theta_i, \theta_j) = \tanh(a(\theta_i \cdot \theta_j) + b), \tag{3.14}$$

where d, σ, a, and b in (3.12), (3.13), and (3.14) refer to the degree of polynomial, the spread of radial basis function, and the slope and bias in sigmoid functions, respectively; these are essentially tuning parameters chosen by the user [66]. 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 [73]; here, we reproduce the centered Gram matrix as

$$\tilde{K} = K - 1_n K - K 1_n + 1_n K 1_n, \tag{3.15}$$

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

3.3.1 Accuracy of the estimated LPV model

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

$$a(m) = \frac{\sum_{i=1}^{m}\lambda_i}{\sum_{i=1}^{\bar{m}}\lambda_i}, \tag{3.16}$$

where m is the number of reduced variables, and λ_i denotes the i^{th} eigenvalue of the kernel matrix $\tilde{K}$ in (3.15); 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 [73], 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.

3.3.2 THE PRE-IMAGING PROBLEM

For linear PCA, the reduced LPV model is affine in the new scheduling variables ρ_t, as shown in [30]. 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 [64] 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 [64] whose convergence depends highly on factors such as initial conditions and the choice of kernel function [74]. 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

$$\tilde{Q}(\rho_t) = \begin{bmatrix} \tilde{A}(\rho_t) & \tilde{B}(\rho_t) \\ \tilde{C}(\rho_t) & \tilde{D}(\rho_t) \end{bmatrix} \approx \begin{bmatrix} A(\tilde{\theta}_t) & B(\tilde{\theta}_t) \\ C(\tilde{\theta}_t) & D(\tilde{\theta}_t) \end{bmatrix} = Q(\tilde{\theta}_t). \tag{3.17}$$

Using (3.3) and (3.17), we can relate the reduced model to the full-order LPV model as follows:

$$\tilde{Q}(\rho_t) \approx Q(\tilde{\theta}_t) = Q_0 + \sum_{i=1}^{l} Q_i \tilde{\theta}_{t,i} = Q_0 + \sum_{i=1}^{l} Q_i f_i \left(\rho_t, \tilde{\theta}_t\right), \tag{3.18}$$

where $f(\rho_t, \tilde{\theta}_t)$ is a nonlinear pre-image function as derived in [74].

3.3.3 OPTIMIZING THE REDUCED LPV MODEL FOR CONTROLLER SYNTHESIS

The reduced model in (3.18) depends on ρ_t through a nonlinear function. In our problem statement in Section 3.2, 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. [70], the vertex property implies that for such an LPV representation, the state-space matrices for any ρ_t belong to a matrix polytope as

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

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 [71] 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

Algorithm 2 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 θ_t for $t = 1, \cdots, n$.
Step 3: Compute the kernel matrix K using (3.10).
Step 4: Center the data in the feature space to find $\tilde{K}$ using (3.15).
Step 5: Diagonalizing $\tilde{K}$, normalize eigenvectors α, and save α for projecting scheduling variables θ_t and extracting ρ_t.
Step 7: Compute ρ_t for $t = 1, \cdots, n$ using (3.11).
Step 8: Solve the optimization problem (3.19) and obtain a reduced LPV state-space model.

$$\min_{R_i, P_i, S_i \; i=0,1,\cdots,m} \quad \frac{1}{n} \sum_{t=1}^{n} \left\| Q(\theta_t) - R(\rho_t) \right\|_{\mathrm{F}}^2, \tag{3.19}$$

where $Q(\theta_t)$ is as defined in (3.3), $\|\cdot\|_{\mathrm{F}}$ represents the Frobenius norm, m is the number of reduced scheduling variables, n is the total samples, and

$$R(\rho_t) = R_0 + \sum_{i=1}^{m} R_i \rho_{t,i} = \begin{bmatrix} \breve{A}(\rho_t) & \breve{B}(\rho_t) \\ \breve{C}(\rho_t) & \breve{D}(\rho_t) \end{bmatrix}$$

for an affine approximation, or

$$R(\rho_t) = \left\{ P_0 + \sum_{i=1}^{m} P_i \left(\rho_{t,i}\right)^j \right\}^{-1} \left\{ S_0 + \sum_{i=1}^{m} S_i \left(\rho_{t,i}\right)^j \right\} = \begin{bmatrix} \breve{A}(\rho_t) & \breve{B}(\rho_t) \\ \breve{C}(\rho_t) & \breve{D}(\rho_t) \end{bmatrix}$$

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 model is expected to provide a good estimate of the actual model with $m < l$. The proposed model reduction method is summarized in Algorithm 2.

3.4 Robotic Manipulator Example

The dynamic model of a two-link planar robotic manipulator is considered here with its configuration shown in Figure 3.2. Detailed nonlinear model and the associated parameters have been taken

from [69]. 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 [30, 69], a LPV state-space 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

$$A(\theta) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ cd\theta_3 & -be\theta_4 & \theta_5 & b\theta_6 \\ -bd\theta_7 & ae\theta_8 & \theta_9 & \theta_{10} \end{bmatrix},\ B(\theta) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ cnk_m\theta_1 & -bnk_m\theta_2 \\ -bnk_m\theta_2 & ank_m\theta_1 \end{bmatrix}, C = I_{4\times4},\ D = O_{4\times2}, \tag{3.20}$$

where

$$\begin{aligned} h &= ac - b^2\cos^2\Delta, \\ \theta_1 &= \frac{g}{h}, \theta_2 = \frac{g\cos\Delta}{h}, \theta_3 = \frac{\text{sinc}(q_1)}{h}, \\ \theta_4 &= \cos\Delta\frac{\text{sinc}(q_2)}{h}, \\ \theta_5 &= \frac{(-b^2\sin\Delta\ \cos\Delta\dot{q_1} - (c + b\cos\Delta)f)}{h}, \\ \theta_6 &= \frac{-c\sin\Delta\dot{q_2} + \cos\Delta f}{h}, \\ \theta_7 &= \frac{\cos\Delta\text{sinc}(q_1)}{h}, \theta_8 = \frac{\text{sinc}(q_2)}{h}, \\ \theta_9 &= \frac{(ab\ \sin\Delta\dot{q_1} + f(a + b\cos\Delta))}{h} \\ \theta_{10} &= \frac{(b^2\sin\Delta\cos\Delta\dot{q_2} - af)}{h}, \end{aligned}$$

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, see [30, 69]. 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

Table 3.1: Accuracy measure for different kernels as a function of number of scheduling variables m.

kernel fcn	*parameters*	accuracy (%)									
		$m=1$	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$	$m=7$	$m=8$	$m=9$	$m=10$
polynomial	$d=2$	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
	$d=6$	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
RBF	$\sigma=5$	49.96	74.49	83.77	89.20	93.08	95.24	96.70	97.70	98.49	98.94
	$\sigma=10$	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
sigmoid	$\kappa=0.05, b=0.01$	70.09	89.01	96.40	99.63	99.98	100	100	100	100	100
	$\kappa=0.05, b=0.05$	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

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.

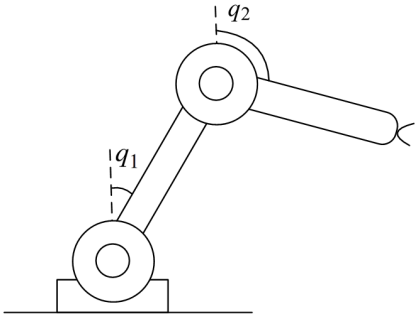

Figure 3.2: Configuration of the 2-DoF robotic manipulator

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 (3.16) of the approximated LPV models as a function of the dimension of reduced variables.

3.4.1 CHOOSING THE KERNEL FUNCTION

The general question of how to choose a kernel function is still an open problem [75]. Jolliffe *et al.* argued in [72] 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 [61] 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 (3.16). 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 3.1. Accuracy results for polynomial, RBF, and sigmoid kernels as functions of extracted component m are shown in Figure 3.3, 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 7^{th} 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 [75]. The results obtained for kernel PCA with the chosen degree 7 polynomial kernel are shown in comparison with linear PCA in Figure 3.4. Higher accuracy is clearly observed in case of kernel PCA for $m \leq 5$.

Once the reduced scheduling variables are obtained, cost function (3.19) 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

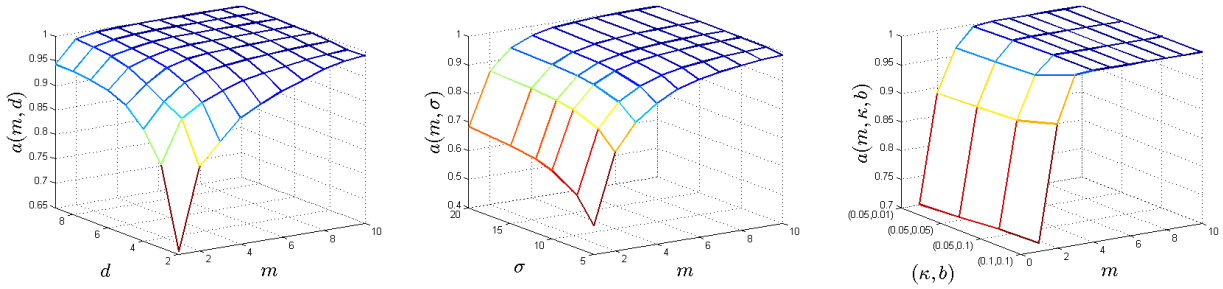

Figure 3.3: From left to right: Accuracy plots as functions of number of scheduling variables m for (a) polynomial kernel (b) radial basis function kernel and (c) sigmoid kernel. In each case, the y-axis shows the kernel parameters.

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 3.5 shows the states of the open-loop robotic manipulator over a period of 10 seconds. It 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

$$\text{BFR} := 100\% \cdot \max\left(\frac{\|x - \hat{x}\|_2}{\|x - \bar{x}\|_2}, 0\right), \tag{3.21}$$

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

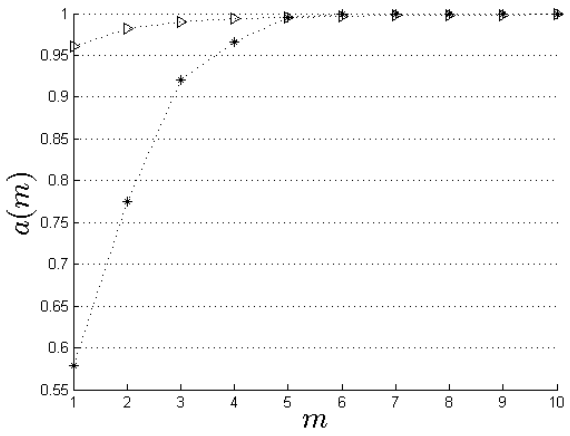

Figure 3.4: Accuracy of the approximation (3.16) as a function of the number of scheduling variables m for linear PCA (*) and polynomial-based kernel PCA ($\triangleright$).

averaged over the four states and tabulated in Table 3.2, 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.

3.4.2 LPV CONTROLLER DESIGN

The LPV controller design configuration is illustrated in Figure 3.6 (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

$$K_c(\rho): \begin{cases} \dot{\zeta}_t = A_K(\rho)\zeta_t + B_K(\rho)y_t, \\ u_t = C_K(\rho)\zeta_t + D_K(\rho)y_t. \end{cases} \tag{3.22}$$

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 [70]. 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) [70]: For the closed-loop LPV system shown in Figure 3.6 (a), the energy-to-energy gain, or induced $\mathcal{L}_2$ gain, from external disturbance w to controlled outputs z is defined as

Table 3.2: Open-loop simulation Monte-Carlo statistics for linear and kernel PCA

	$m=1$		$m=2$		$m=3$		$m=4$		$m=5$	
	ker.	*lin.*	*ker.*	*lin.*	*ker.*	*lin.*	*ker.*	*lin.*	*ker.*	*lin.*
BFR (%)	84.13	62.31	89.08	71.53	93.71	78.41	96.06	85.15	97.67	91.22

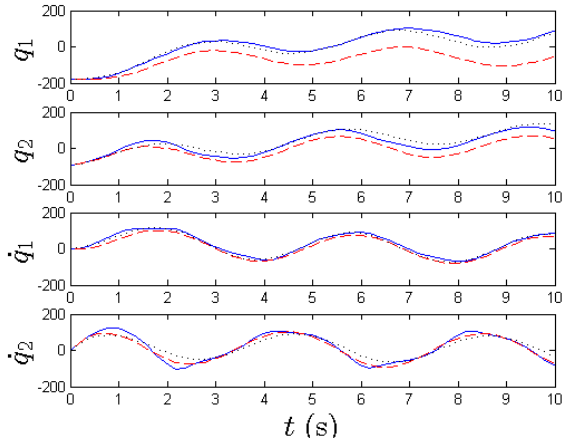

Figure 3.5: States of the robotic manipulator full order LPV model (solid blue line), linear PCA-based reduced LPV model (red dashed line), and kernel PCA-based reduced LPV model (black dotted line) for $m = 1$.

$$\|T_{wz}\|_{i,2} = \sup_{\rho} \sup_{w \neq 0} \frac{\|z\|_{\mathcal{L}_2}}{\|w\|_{\mathcal{L}_2}}, \tag{3.23}$$

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 ρ.

Definition 2 (Induced $\mathcal{L}_2$ gain performance) [70]: The closed-loop LPV system of Figure 3.6 (a) has an induced $\mathcal{L}_2$ gain performance less than γ if there exists a symmetric positive-definite matrix X such that

$$\begin{bmatrix} A_{cl}^{\top}(\rho)X + XA_{cl}(\rho) & XB_{cl}(\rho) & C_{cl}^{\top}(\rho) \\ \star & -\gamma I & D_{cl}^{\top}(\rho) \\ \star & \star & -\gamma I \end{bmatrix} \prec 0, \tag{3.24}$$

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,

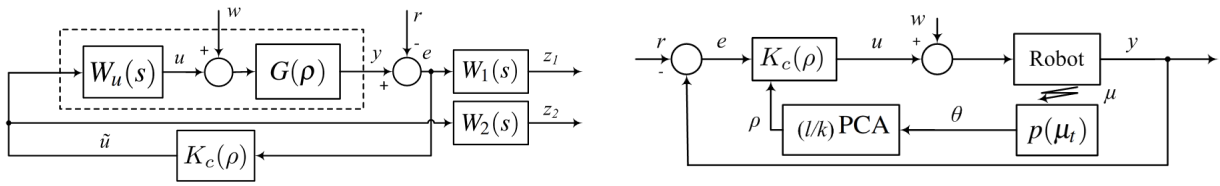

Figure 3.6: (Left) Generalized configuration of closed-loop system composed of reduced LPV model, LPV controller, and design weights; (right) Control of the robotic manipulator using an LPV controller $K_c(\rho)$ based on the reduced LPV model.

the matrix X in the inequality (3.24) 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 (3.24) holds for all trajectories of ρ within the polytope, if it holds true at the vertices (for proof, see [70]). Therefore, we can achieve a closed-loop $\mathcal{L}_2$ gain performance γ if (3.24) 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.

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 [70]). This is not the case for the manipulator example considered here; both the full order LPV model (3.20) 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 3.6 (a), by augmenting the plant with a low pass filter $W_u(s)$ having sufficiently large bandwidth [70]. 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

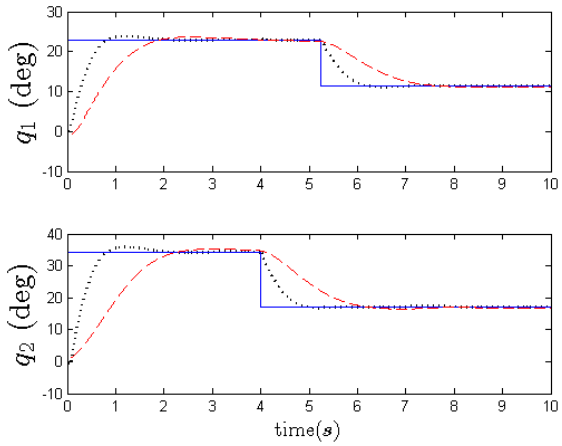

Figure 3.7: Reference trajectory (blue solid line) in comparison with controlled outputs based on linear PCA-based reduced model (red dashed line) and kernel PCA-based reduced model (black dotted line).

external disturbance $w_t = \left[\begin{smallmatrix} r_{t,1} & r_{t,2} & w_{t,1} & w_{t,2} \end{smallmatrix}\right]^\top$ to the controlled outputs $z_t = \left[\begin{smallmatrix} z_{t,1} & z_{t,2} \end{smallmatrix}\right]^\top$ 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 8^{th} 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 3.6 (b)) to control the robotic manipulator model. Reduced scheduling variables are obtained using (3.11) 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. 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 3.7 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 3.8(a) shows the controller outputs for the linear and kernel PCA cases. Figure 3.8(b) shows a magnified view of the linear and kernel-PCA based

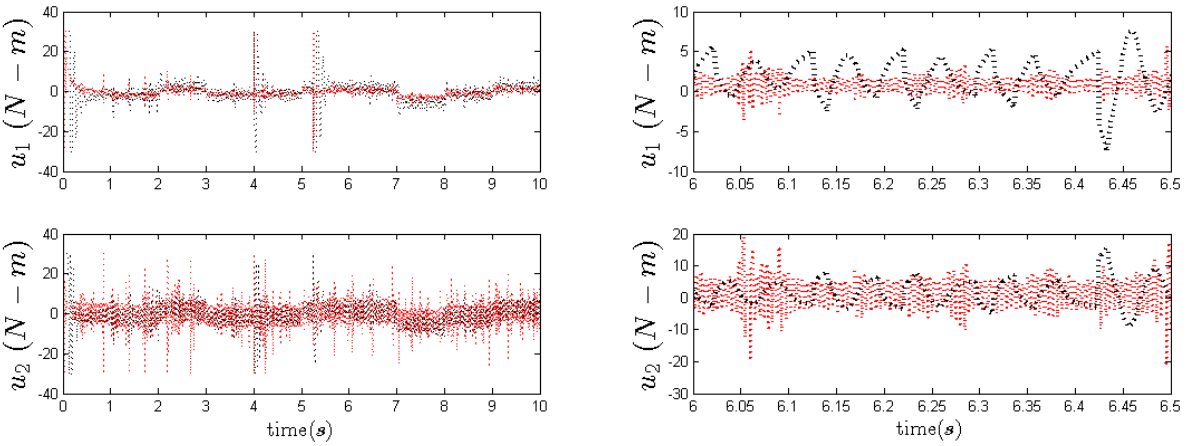

Figure 3.8: From left to right: (a) Outputs of the gain scheduled controllers designed using linear PCA-based reduced model (red dashed line) and kernel PCA-based reduced model (black dotted line); (b) Close-up view of controller outputs.

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. The improvement is observed, not only in terms of reduced rise time and settling time, but also in terms of an overall reduction in chattering, as seen in Figure 3.8.

3.5 Concluding Remarks

In this paper, we have explored the use of kernel-based PCA for dimensionality reduction of the scheduling variables in LPV modeling. 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. While exploitation of the kernel trick provides us with increased degree of freedom, and if properly tuned, better model reduction results and less expensive controller design, the trade-off is observed in terms of the rigorous tuning. Kernel hyper-parameters require efficient tuning. Usually, these hyper-parameters can be searched over a grid. It is noted that this tuning is done offline and does not need to be carried out during online control. Results in this paper provide encouraging insights into the use of kernel PCA for LPV dependency reduction.

CHAPTER 4

A KERNEL-BASED APPROACH TO MIMO LPV STATE-SPACE IDENTIFICATION AND APPLICATION TO A NONLINEAR PROCESS SYSTEM [1]

[1] Syed Z. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin: A Kernel-based Approach to MIMO LPV State-space Identification and Application to a Nonlinear Process System. 2015. *Proc. of the 1st IFAC Workshop on Linear Parameter-Varying Systems*, Grenoble, France: pp. 85-90. Original version can be found using the Digital Object Identifier (DOI): 10.1016/j.ifacol.2015.11.118

ABSTRACT

This paper first describes the development of a nonparametric identification method for *linear parameter-varying* (LPV) state-space models and then applies it to a nonlinear process system. The proposed method uses kernel-based *least-squares support vector machines* (LS-SVM). While parametric identification methods require proper selection of basis functions in order to avoid over-parametrization or structural bias, the problem of variance-bias tradeoff is avoided by estimating the functional dependencies of the state-space representation on the LPV scheduling variables using measured input and output data under the LS-SVM framework. The proposed formulation allows for LS-SVM to reconstruct and uncover static, as well as dynamic dependencies on scheduling variables in *multi-input multi-output* (MIMO) LPV models. This is achieved by assuming that the states are measurable, which is a common scenario during online control of many chemical processes described by lumped parameter models. The proposed method does not require an explicit declaration of the feature maps of the nonlinearities of the assumed model structure; instead, it requires the selection of a nonlinear kernel function and tuning its parameters. The developed identification method is applied to a continuous stirred tank reactor (CSTR) model under realistic noise conditions. Another numerical example along with the CSTR system illustrates the performance of the proposed algorithm under both static and dynamic dependence on the scheduling variables.

4.1 INTRODUCTION

Linear parameter-varying (LPV) models provide a powerful framework for identification of nonlinear systems. Under this framework, nonlinear models can be represented as a linear dynamic relation of the input and output variables; the relation is itself dependent on measurable time-varying signals, commonly known as *scheduling variables*. These scheduling signals express the varying operating conditions of the system. Thus, LPV models represent an intermediate stage between linear time-invariant (LTI) and nonlinear systems, while preserving many attractive attributes of LTI systems. This simplicity of LPV models allows one to apply linear optimal control techniques to nonlinear systems represented by LPV models, opening up the possibility of applying powerful LPV control synthesis tools. In particular, in process systems, it has often been observed by control engineers and modeling practitioners that the dynamics of the process is well captured by linear models at any given operating condition. The concept of LPV models, therefore, comes in very handy in order to extend the LTI models over a wide range of varying operating conditions.

As a natural consequence of this property, LPV identification has attracted a lot of attention in the past decade [5], with different identification schemes developed for both LPV input-output and state-space models [35, 36, 50, 51, 76–78].

For LPV state-space model identification, to the best of authors' knowledge, most techniques in the literature fall under the category of parametric approaches. In parametric approaches, the scheduling dependencies of the model coefficients are described as a linear combination of basis functions that need to be chosen *a priori*. However, the selection of these basis functions remains difficult since overestimating the number of basis functions leads to over-parameterized models and hence a large variance in the estimates despite the low order of the model. On the other hand, an inappropriate selection of these functions is known to cause structural bias (see [36]). For LPV state-space and bilinear models, parametric methods are based on various subspace approaches, which are extensions of the well-accepted subspace identification methods used for LTI systems. These methods usually require a high computational demand due to the enormous dimension of

the data matrices involved (the growth is polynomial in the state dimension and exponential in the scheduling variables). Verhaegen et al. proposed a solution to overcome this curse of dimensionality on the expanse of approximation of the data equations and identify LPV state-space models with affine parameter dependence [49]. Other subspace-based methods were later published in [46, 51, 52, 79].

Nonparametric methods provide an alternative way to avoid the bias-variance tradeoff by obtaining nonparametric reconstruction of the scheduling dependencies in LPV models. In particular, with the emergence of kernel-based techniques, a new avenue of nonparametric identification, classification, and data processing has appeared in the last two decades. Kernels are functions that enable us to perform linear operations in high-dimensional feature spaces, often mapping nonlinear dependencies very efficiently using the so-called *kernel trick* [61]. This has sprouted the use of kernel-based techniques for solving different problems under the umbrella of LPV system identification, ranging from LPV model reduction [74] to estimating coefficient dependencies in LPV I/O models [36]. Verhaegen et al. incorporated kernel methods in their earlier subspace-based technique in order to reduce high dimensional data matrices (see [50]). The identification approaches published in [36, 41, 80] reported efficient kernel-based methods employing LS-SVM for LPV I/O models; the results showed consistent estimates of the coefficient dependencies with a very attractive bias-variance tradeoff. A mixed parametric method for LPV state-space identification was proposed recently in [57]. The authors described the C matrix using a nonparametric LS-SVM-based model, while the A matrix was described by a parametric model. The model structure was assumed to be in companion reachability canonical form (CR-CF), and the coefficients were estimated using an iterative routine.

In this work, we specifically aim to use the properties of LS-SVM for LPV modeling of nonlinear process systems. Many chemical processes, including high purity distillation columns, exhothermic and non-isothermal chemical reactors, and batch systems are inherently nonlinear. Due to their nonlinearities, they cannot be efficiently stabilized and monitored with controllers and estimators designed on the basis of linearized models around an operating point. Sources ranging

from the Arrhenius temperature dependence of reaction rates, radiative heat transfer phenomena, to complex reaction mechanisms cause for the highly nonlinear behaviors in chemical processes [81]. Increasingly, limitations of traditional linear control and modeling methods have become apparent in dealing with nonlinear chemical processes as processes are required to operate over a wide range of conditions. *Multi-input multi-output* (MIMO) LPV state-space models can fill this gap by modeling the processes with a linear dynamic relation between the process inputs and outputs, where the relation itself is a nonlinear function of time-varying parameters. This way, several linear control techniques can be easily applied to systems represented by an LPV model.

In this paper, we present a nonparametric kernel-based identification method for MIMO LPV state-space models with measurable states; this is a common situation in several lumped parameter models of process systems in which the states are directly accessible for measurement. We employ LS-SVM in order to explore the coefficient dependencies on the scheduling variables with a good variance-bias tradeoff. We finally validate the proposed technique on the model of an ideal continuous stirred tank reactor (CSTR). The paper is arranged as follows. The problem formulation is presented in Section 4.2. The LPV state-space model is formulated in an LS-SVM setting and identification algorithm is derived in Section 4.3. Numerical examples are provided in Section 4.4, where a discussion about the results is also given. Concluding remarks are finally made in Section 4.5.

4.2 Problem Formulation

Consider an LPV system represented by the following discrete-time state-space model with innovation noise model

$$\begin{aligned} x_{k+1} &= A(p_k)x_k + B(p_k)u_k + K(p_k)e_k, \\ y_k &= C(p_k)x_k + e_k, \end{aligned} \tag{4.1}$$

where k denotes discrete time, matrices $A(p_k) \in \mathbb{R}^{n\times n}$, $B(p_k) \in \mathbb{R}^{n\times n_u}$, $K(p_k) \in \mathbb{R}^{n\times n_y}$, and $C(p_k) \in \mathbb{R}^{n_y\times n}$ are functions of time-varying scheduling variables $p_k \in \mathbb{R}^{n_p}$, and $e_k \in \mathbb{R}^{n_y}$ is a

stochastic white noise process. In addition, x and y represent the model states and sensor measurements, respectively. Assuming that the states are available for measurement, we aim at employing nonlinear kernel functions under the LS-SVM framework in order to estimate the functional dependencies of the state-space matrices on the scheduling variables. We can rewrite (4.1) as

$$\begin{aligned} x_{k+1} &= \underbrace{\left(A(p_k) - K(p_k)C(p_k)\right)}_{\tilde{A}(p_k)} x_k + B(p_k)u_k + K(p_k)y_k, \\ y_k &= C(p_k)x_k + e_k, \end{aligned} \tag{4.2}$$

which can be reformulated as follows

$$\begin{aligned} x_{k+1} &= W_1\Phi_1(p_k)x_k + W_2\Phi_2(p_k)u_k + W_3\Phi_3(p_k)y_k, \\ y_k &= W_4\Phi_4(p_k)x_k + e_k, \end{aligned} \tag{4.3}$$

where $W_{1,2,3} \in \mathbb{R}^{n\times n_H}$ and $W_4 \in \mathbb{R}^{n_y\times n_H}$ are unknown weighting matrices, while matrices $\Phi_1(p_k) \in \mathbb{R}^{n_H\times n}$, $\Phi_2(p_k) \in \mathbb{R}^{n_H\times n_u}$, $\Phi_3(p_k) \in \mathbb{R}^{n_H\times n_y}$, and $\Phi_4(p_k) \in \mathbb{R}^{n_H\times n}$ represent unknown feature maps. Variable n_H represents dimension of, a possibly infinite dimensional feature space. The problem, therefore, boils down to finding the state-space matrices dependencies $W_i\Phi_i(p_k)$ for $i = 1, \cdots, 4$, given the data $\{u_k, y_k, x_k, p_k\}_{k=1}^N$, where N is the number of data points (samples).

4.3 KERNEL-BASED LPV STATE-SPACE MODEL IDENTIFICATION

The estimates of the LPV state-space matrices in the form of $W_i\Phi_i(p_k)$ can be obtained by minimizing the following cost function

$$J = \quad \frac{1}{2}\sum_{i=1}^{4} ||W_i||_{\mathrm{F}}^2 + \frac{1}{2}\sum_{k=1}^{N} e_k^\top \Gamma e_k, \tag{4.4}$$

over $W_{1,2,3,4}$, where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, and $\Gamma = \mathrm{diag}(\gamma_1, \cdots, \gamma_n)$ is a diagonal weighting matrix on the residual errors e_k, and is known as the regularization matrix. The aforedescribed optimization problem can be solved by introducing *Lagrange multipliers* and substituting the inner product $\Phi_i\Phi_i^\top$ using an *a priori* chosen nonlinear kernel function as shown in [36]. This

is expected to give us a nonparametric estimate of the coefficients function matrices representing the state-space matrices of the original LPV model. We define the Lagrangian function as

$$\begin{aligned}\mathcal{L}(W_1, W_2, W_3, W_4, \boldsymbol{\alpha}, \beta, e) = \\ J - \sum_{j=1}^{N} \alpha_j^\top \left\{W_1\Phi_1(p_j)x_j + W_2\Phi_2(p_j)u_j + W_3\Phi_3(p_j)y_j - x_{j+1}\right\} \\ - \sum_{j=1}^{N} \beta_j^\top \left\{W_4\Phi_4(p_j)x_j + e_j - y_j\right\},\end{aligned} \tag{4.5}$$

where $\alpha_j \in \mathbb{R}^n$, $\beta_j \in \mathbb{R}^{n_y}$ are the Lagrange multipliers at discrete time j. Due to the convexity of the problem, the global optimum is obtained when the derivatives are equal to zero as follows

$$\frac{\partial \mathcal{L}}{\partial \alpha_j} = 0 \Rightarrow x_{j+1} = W_1\Phi_1(p_j)x_j + W_2\Phi_2(p_j)u_j + W_3\Phi_3(p_j)y_j, \tag{4.6a}$$

$$\frac{\partial \mathcal{L}}{\partial W_1} = 0 \Rightarrow W_1 = \sum_{j=1}^{N} \alpha_j x_j^\top \Phi_1^\top(p_j), \tag{4.6b}$$

$$\frac{\partial \mathcal{L}}{\partial W_2} = 0 \Rightarrow W_2 = \sum_{j=1}^{N} \alpha_j u_j^\top \Phi_2^\top(p_j), \tag{4.6c}$$

$$\frac{\partial \mathcal{L}}{\partial W_3} = 0 \Rightarrow W_3 = \sum_{j=1}^{N} \alpha_j y_j^\top \Phi_3^\top(p_j), \tag{4.6d}$$

$$\frac{\partial \mathcal{L}}{\partial W_4} = 0 \Rightarrow W_4 = \sum_{j=1}^{N} \beta_j x_j^\top \Phi_4^\top(p_j), \tag{4.6e}$$

$$\frac{\partial \mathcal{L}}{\partial e_j} = 0 \Rightarrow \beta_j = \Gamma e_j, \tag{4.6f}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = 0 \Rightarrow y_j = W_4\Phi_4(p_j)x_j + e_j. \tag{4.6g}$$

Substituting (4.6a)-(4.6g) in (4.3), we can write the following

$$\begin{aligned}x_{k+1} &= W_1\Phi_1(p_k)x_k + W_2\Phi_2(p_k)u_k + W_3\Phi_3(p_k)y_k \\ &= \underbrace{\left\{\sum_{j=1}^{N} \alpha_j x_j^\top \Phi_1^\top(p_j)\right\}}_{W_1}\Phi_1(p_k)x_k + \underbrace{\left\{\sum_{j=1}^{N} \alpha_j u_j^\top \Phi_2^\top(p_j)\right\}}_{W_2}\Phi_2(p_k)u_k \\ &+ \underbrace{\left\{\sum_{j=1}^{N} \alpha_j y_j^\top \Phi_3^\top(p_j)\right\}}_{W_3}\Phi_3(p_k)y_k,\end{aligned} \tag{4.7}$$

$$y_k = \underbrace{\left\{\sum_{j=1}^{N} \beta_j x_j^\top \Phi_4^\top(p_j)\right\}}_{W_4} \Phi_4(p_k)x_k + \underbrace{\Gamma^{-1}\beta_k}_{e_k}. \tag{4.8}$$

Replacing the inner-product $\Phi_i(p_k)^\top \Phi_i(p_j)$ by a kernel function $\bar{k}^i(p_j, p_k)$, we further define kernel matrices Ω and Ξ as

$$\begin{aligned}
[\Omega]_{j,k} &= \sum_{i=1}^{3} z_i^\top(j)\bar{k}^i(p_j, p_k)z_i(k), \\
[\Xi]_{j,k} &= x_j^\top \bar{k}^4(p_j, p_k)x_k,
\end{aligned} \tag{4.9}$$

$$\text{where} \qquad z_i(k) = \begin{cases} x_k, & i = 1 \\ u_k, & i = 2 \\ y_k, & i = 3. \end{cases}$$

While a wide variety of kernel functions exists in the literature to choose from, commonly used kernels include the *Radial Basis Function* (RBF), *polynomial* or *sigmoid* kernels among many others (see [61]). A typical RBF kernel, also known as the Gaussian kernel, is represented by

$$\bar{k}^i(p_j, p_k) = \exp\left(-\frac{\|p_j - p_k\|_2^2}{2\sigma_i^2}\right), \tag{4.10}$$

where σ_i is a free kernel parameter, and $\|\cdot\|_2$ represents the 2-norm. We can now write (4.7)-(4.8) in a compact form as follows

$$X_{k+1} = \boldsymbol{\alpha}\Omega, \tag{4.11}$$

$$Y = \beta\Xi + \Gamma^{-1}\beta, \tag{4.12}$$

where $\Omega \in \mathbb{R}^{N\times N}$ and $\Xi \in \mathbb{R}^{N\times N}$ are kernel matrices as defined above, $\boldsymbol{\alpha} = [\alpha_1 \cdots \alpha_N] \in \mathbb{R}^{n\times N}$ and $\beta = [\beta_1 \cdots \beta_N] \in \mathbb{R}^{n_y\times N}$ are the matrices containing the Lagrange multipliers, $X_{k+1} = [x_2 \cdots x_{N+1}] \in \mathbb{R}^{n\times N}$ and $Y = [y_1 \cdots y_N] \in \mathbb{R}^{n_y\times N}$ contain the states and outputs for the N samples, respectively. The solution to the above equations can be obtained as follows

$$\boldsymbol{\alpha} = X_{k+1}\Omega^{-1}, \tag{4.13}$$

$$\text{vec}(\beta) = \left(I_N \otimes \Gamma^{-1} + \Xi^\top \otimes I_{n_y}\right)^{-1} \text{vec}(Y), \tag{4.14}$$

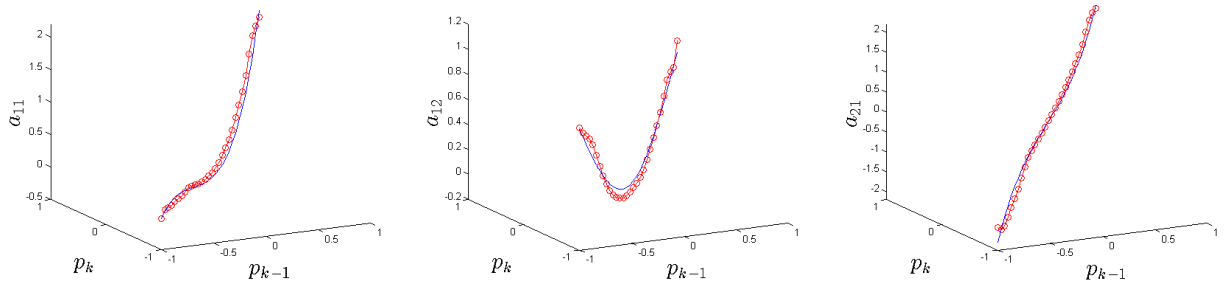

Figure 4.1: Example 1: Elements (functions) a_{11}, a_{12} and a_{21} of the state matrix as a function of scheduling variable p_k and its delayed sample p_{k-1}. Solid blue line represents the original elements while the circled red line indicates their estimates.

where $\otimes$ denotes the Kronecker product and $\mathrm{vec}(\cdot)$ denotes vectorization function, which stacks subsequent columns in a matrix below one another; matrices I_N and I_{n_y} represent identity matrices of dimensions N and n_y, respectively. The solution (4.14) is obtained using the solution to the classical sylvester equation [82]. Once trained, the estimate of the state-space matrices can be calculated by using (4.6b)-(4.6e) as

$$\tilde{A}_e(\cdot) = W_1\Phi_1(\cdot) = \sum_{k=1}^{N} \alpha_k x_k^\top \bar{k}^1(p_k, \cdot), \tag{4.15a}$$

$$B_e(\cdot) = W_2\Phi_2(\cdot) = \sum_{k=1}^{N} \alpha_k u_k^\top \bar{k}^2(p_k, \cdot), \tag{4.15b}$$

$$K_e(\cdot) = W_3\Phi_3(\cdot) = \sum_{k=1}^{N} \alpha_k y_k^\top \bar{k}^3(p_k, \cdot), \tag{4.15c}$$

$$C_e(\cdot) = W_4\Phi_4(\cdot) = \sum_{k=1}^{N} \beta_k x_k^\top \bar{k}^4(p_k, \cdot), \tag{4.15d}$$

where subscript e denotes estimate. Once estimates of $\tilde{A}, C, K$ are obtained, estimate $A_e = \tilde{A}_e + K_e C_e$ can be calculated accordingly. This gives a nonparametric estimate of the state-space matrices. It is noteworthy here that the parameter matrices W_i or the basis functions $\Phi_i(\cdot)$ are not accessible explicitly. What we are able to estimate via nonlinear kernel functions is $W_i\Phi_i(\cdot)$.

Identification of LPV models with dynamic dependence on the scheduling variables

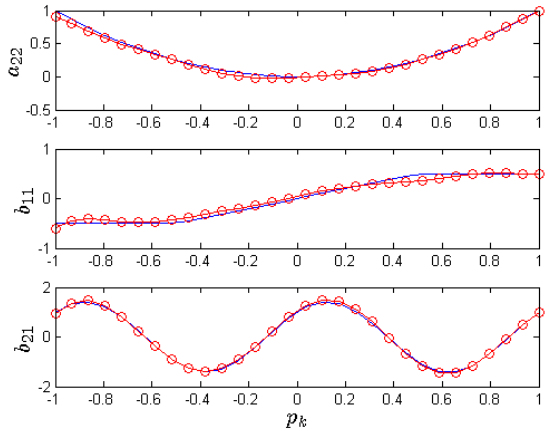

Figure 4.2: Example 1: Elements a_{22}, b_{11} and b_{21} of the LPV state-space matrices as a function of the scheduling variable p_k. The solid blue line represents the original functions while the circled red line indicates their estimates

Next, we consider the case, where the state-space matrices of the LPV model have dynamic dependence on the scheduling variables. Such an LPV state-space model can be described by

$$\begin{aligned} x_{k+1} &= A(p_k, \tilde{n})x_k + B(p_k, \tilde{n})u_k + K(p_k, \tilde{n})e_k, \\ y_k &= C(p_k, \tilde{n})x_k + e_k, \end{aligned} \tag{4.16}$$

where $A(p_k, \tilde{n}) = A(p_k, \cdots, p_{k-\tilde{n}})$ signifies the dependence of A on $\tilde{n}$ past values of the scheduling variables. Substituting $e_k = y_k - C(p_k, \tilde{n})x_k$ in the equation for x_{k+1}, we can rewrite the above set of equations as

$$\begin{aligned} x_{k+1} &= \tilde{A}(p_k, \tilde{n})x_k + B(p_k, \tilde{n})u_k + K(p_k, \tilde{n})y_k, \\ y_k &= C(p_k, \tilde{n})x_k + e_k, \end{aligned} \tag{4.17}$$

where $\tilde{A}(p_k, \tilde{n}) = A(p_k, \tilde{n}) - K(p_k, \tilde{n})C(p_k, \tilde{n})$. Following the same procedure as before, we arrive at the following equation for the states x_k for $k \in \{1, \cdots, N\}$

$$\begin{aligned} x_{k+1} &= \sum_{i=1}^{3} \left\{ \left(\sum_{j=1}^{N} \alpha_j z_i^\top(j) \Phi_i^\top(p_j, \tilde{n}) \right) \Phi_i(p_k, \tilde{n}) z_i(k) \right\}, \\ y_k &= \left\{ \sum_{j=1}^{N} \beta_j x_j^\top \Phi_4^\top(p_j, \tilde{n}) \right\} \Phi_4(p_k, \tilde{n}) x_k + \Gamma^{-1} \beta_k, \end{aligned} \tag{4.18}$$

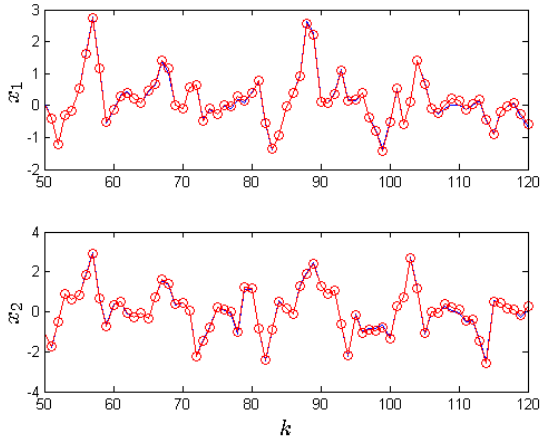

Figure 4.3: Example 1: States x_1, x_2 of the LPV system model. The solid blue and the circled red lines represent the original and simulated state response of the estimated model, respectively.

where $z_i(k)$ is defined as before. The kernel matrices Ω and Ξ can then be written in a modified form as

$$[\Omega]_{j,k} = \sum_{i=1}^{3} z_i^\top(j)\bar{k}^i(p(j,\tilde{n}),p(k,\tilde{n}))z_i(k).$$
$$[\Xi]_{j,k} = x_j^\top \bar{k}^4(p(j,\tilde{n}),p(k,\tilde{n}))x_k. \tag{4.19}$$

The RBF kernel is calculated as follows

$$\bar{k}^i(p(j,\tilde{n}),p(k,\tilde{n})) = \exp\left(-\frac{\|\mathbf{p}(j,\tilde{n})-\mathbf{p}(k,\tilde{n})\|_2^2}{2\sigma_i^2}\right), \tag{4.20}$$

where $\mathbf{p}(j,\tilde{n}) = \begin{bmatrix} p_j^\top & p_{j-1}^\top & \cdots & p_{j-\tilde{n}}^\top \end{bmatrix}^\top$. Other kernels can be defined in a similar way. By applying the kernel function over a dynamic range of present and past values of the scheduling variables, dynamic coefficient dependencies of the state-space matrices on the scheduling variables can be mapped. In the next section, we examine the performance of the developed algorithm by means of different nonlinear examples with static and dynamic dependence on the scheduling variables.

4.4 Numerical Examples

The following examples are considered

Table 4.1: Example 1: Monte-Carlo simulation results for example 1: BFR values for the underlying coefficient functions.

Fcn	Mean (BFR)	STD (BFR)
a_{11}	89.24%	0.532
a_{12}	88.01%	0.718
a_{21}	93.01%	0.098
a_{22}	92.05%	0.188
b_{11}	87.22%	1.102
b_{21}	92.54%	0.021

4.4.1 EXAMPLE 1

The following numerical example of a second order discrete-time LPV state-space model is considered.

$$x_{k+1} = A(p_k, p_{k-1})x_k + B(p_k)u_k + e_k,$$

with

$$A(p_k, p_{k-1}) = \begin{bmatrix} p_k^3 + p_{k-1}^2 & p_k \tanh(p_{k-1}) \\ p_k^3 + p_{k-1} & p_k^2 \end{bmatrix}, \; B(p_k) = \begin{bmatrix} \mathrm{sat}(p_k) \\ \sin(2\pi p_k) + \cos(2\pi p_k) \end{bmatrix},$$

where

$$\mathrm{sat}(p_k) = \begin{cases} -0.5, & p_k < -0.5 \\ 0.5, & p_k > 0.5 \\ 0 & \text{otherwise.} \end{cases}$$

As can be noticed, elements a_{11}, a_{12} and a_{21} have a dynamic dependence on p_k, while a_{22}, b_{11} and b_{21} depend only on p_k. A total of 1200 samples of scheduling variables $p_k \in [-1, 1]$ are generated as $p_k = \sin(0.5k)$. Input signals u_k are generated randomly. The generated data is divided into 900 and 300 samples for training and validation, respectively. Gaussian white noise e_k is added such that an output signal-to-noise ratio (SNR) of 25 dB is maintained. RBF kernel is chosen with its parameters tuned as $\sigma_i = 0.45 \; \forall i$ and $\Gamma = \mathrm{diag}\{300, 300\}$. The proposed LS-SVM algorithm is run and the Lagrange multipliers are estimated. Validation is performed on the validation data;

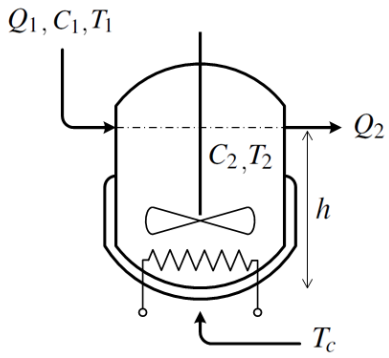

Figure 4.4: An ideal continuous stirred tank reactor

we define *Best Fit Ratio* (BFR) as

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

where $\hat{x}$ and $\bar{x}$ represent predicted states and mean value of the states, respectively. An average output BFR of 92.05% with a standard deviation of 1.54 is achieved for 100 runs of the Monte-Carlo simulation. BFR statistical information for the underlying functional dependencies of the coefficients over $p \in [-1, 1]$ is tabulated in Table 4.1. Estimated elements a_{11}, a_{12}, and a_{21} of the state-space matrix A that have dynamic dependence on the scheduling variable are shown in Figure 4.1. Other functions, namely a_{22}, b_{11} and b_{21} are shown in Figure 4.2; the estimated functions do not show dependency on time-shifted scheduling variables, and are hence, plotted in a two-dimensional plot. Figure 4.3 shows predicted states as compared to the actual ones. The BFR values, as well as the approximated functional dependencies shown in the figures, demonstrate the remarkable ability of the proposed kernel-based method to estimate nonlinear dependencies, with both static and dynamic dependence, with a great accuracy.

4.4.2 Example 2: A continuous stirred tank reactor

In the second example, the model of an ideal *continuous stirred tank reactor* (CSTR) is considered. Schematic diagram of the CSTR process is shown in Figure 4.4. It shows the chemical reaction, under ideal conditions, that converts an inflowing liquid to a product; this reaction is non-isothermal as described in [83]. A heat coolant-based exchanger is used in order to control the

temperature inside the reactor. The first principles-based model is described as

$$\begin{aligned} \dot{C}_2 &= \frac{Q_1}{V}(C_1 - C_2) - k_0 e^{-\frac{E_A}{RT_2}} C_2, \\ \dot{T}_2 &= \frac{Q_1}{V}(T_1 - T_2) - \frac{U_{HE}}{A_{HE}}(T_2 - T_c) + \frac{\Delta H k_0}{\rho c_\rho} e^{-\frac{E_A}{RT_2}} C_2. \end{aligned} \tag{4.21}$$

A typical control objective is to regulate the concentration and the temperature in the reactor, denoted by C_2 and T_2, respectively. To this end, variables Q_1 and T_c, which represent the flow of inflowing liquid and the temperature of the coolant, respectively, are used as manipulatable control signals. Steady-state operating conditions as described in [84] and [83], are tabulated in Table 4.2. The authors in [83] show that introducing a step-disturbance in the inflowing rate of the liquid shows different behavior in the dynamics of the controlled variables T_2 and C_2 in terms of both the time constant and the relative gain when operating at different values of the inflowing liquid concentration C_1. Since the raw material can be obtained from different sources, the concentration C_1 can have differing values ranging from 50% to 150% of the nominal value. The dynamics differs not only in the values of the time constant and the relative gains, the response also shows a change in the sign of the gain exhibiting non-minimum phase behavior. It is evident that a PID controller designed for the nominal value of C_1 might easily fail to stabilize the plant. Hence, LPV modeling naturally appears to be a logical representational choice, using the concentration C_1 as the scheduling variable. It is assumed that the reactor is mixed ideally, that the density and the physical properties of the process remain constant, that the reaction is first order with a temperature relation according to Aarrhenius law, and that the temperature increase in the coolant over the coil can be neglected. It is also assumed that the inflow and outflow rates, Q_1 and Q_2, are kept equal to each other.

For the aforedescribed system, a *pseudo random binary sequence* (PRBS) of the two inputs are used to excite the CSTR model. A trajectory of slowly varying scheduling variable, C_1, is generated ranging from 50% to 150% of the nominal value given in Table 4.2. Gaussian white noise is added such that SNRs of 20dB and 30dB are maintained for T_2 and C_2, respectively. An RBF kernel is selected for training and regularization, and kernel parameters are tuned to be $\Gamma = \mathrm{diag}\{10^4,\ 10^4\}$ and $\sigma = 620$. Both σ and Γ are tuned after a fine grid search over possible combinations of

Table 4.2: Example 2: Steady-state values of variables and constants for the CSTR model

V	Reactor volume	5 m^3
C_1	Concentration of the inflowing liquid	800 kg/m^3
C_2	Concentration in the reactor	213.69 kg/m^3
Q_1	Inflowing rate	0.01 m^3/s
Q_2	Outflowing rate	0.01 m^3/s
k_0	Pre-exponential term	$25s^{-1}$
E_A	Activation energy of reaction	30,000 (J/kg)
T_1	Temp. of inflowing liquid	353 K
T_2	Temp. in the reactor	428.5 K
T_c	Coolant temperature	300 K
ρ	Density	800 kg/m^3
c_ρ	Specific heat	1000(J/kg.s)
ΔH	Heat of reaction	125,000 J/kg
U_{HE}	Heat transfer coefficient	1,000 (J/kg.s)
A_{HE}	Surface area of heat exchanger	1 m^2
h	Liquid level	5 m
R	Gas constant	8.31 (J/mol.K)

parameter values. An adequate sampling time of 60s is chosen, and the model is simulated to generate input and output samples. Collected data is divided into training and validation sets and the proposed LS-SVM routine is run. Trained model is then validated on noise-free validation data and results are shown in Figure 4.5; an output BFR of 85.04% is achieved. The scheduling variable trajectory is also shown in Figure 4.5. While LPV modeling based on *orthogonal basis functions* (OBF) have shown comparatively higher BFR values for validation data in [83], the OBF method uses a polynomial interpolation of order 8. As opposed to that, in the present method, the state order is not increased, and a good approximation is achieved with a second order LPV state-space approximation.

4.5 Concluding Remarks

This paper has introduced a non-iterative and nonparametric identification scheme for LPV state-space models whose states are available for measurement, a situation often occurring in process systems, where the states are directly measurable. The proposed technique is an extension to the LS-SVM identification method for LPV models in an input-output form introduced in [36]. We utilize a dual optimization scheme that lowers the variance of the estimates and is able to uncover

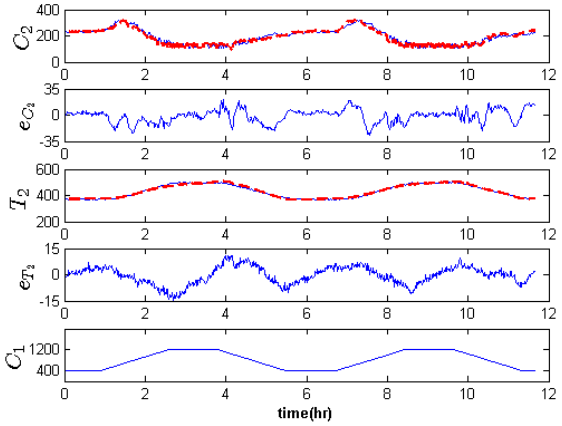

Figure 4.5: (Top to bottom): Concentration in reactor C_2; error in C_2 estimation; reactor temperature T_2; error in T_2 estimation; concentration of inflowing liquid C_1. Variable C_1 represents the scheduling variable. Solid blue lines represent the actual values while dotted red lines represent the predicted ones.

the structural dependency of a MIMO LPV model without over-parametrization. Both static and dynamic dependence of the state-space matrices on the scheduling variables have been explored under noisy conditions. As a case study, nonlinear model of an ideal CSTR has been considered and the proposed model is used to fit identification data, giving encouraging prediction results when subjected to a fresh set of validation data. The proposed method, which requires only a proper choice of a kernel and tuning of the respective kernel parameters is able to map nonlinear dependencies, thereby providing a model that can be used to design LPV controllers.

CHAPTER 5

AN IV-SVM-BASED APPROACH FOR IDENTIFICATION OF STATE-SPACE LPV MODELS UNDER GENERIC NOISE CONDITIONS [1]

[1] Syed Z. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin: An IV-SVM-based Approach for Identification of State-Space LPV Models under Generic Noise Conditions. 2015. *Proc. of the 54th IEEE Conference on Decision and Control*, Osaka, Japan: pp. 7380-7385.

ABSTRACT

This paper presents a nonparametric identification method for state-space linear parameter-varying (LPV) models using a modified support vector machine (SVM) approach. While most LPV identification schemes in the state-space form fall under the general category of parametric methods, regularization-based SVMs provide a viable alternative to model scheduling dependencies, without the need of specifying the dependency structure and with an attractive bias-variance trade-off. In this paper, a solution is proposed for nonparametric identification of LPV state-space models in terms of least-squares SVMs (LS-SVM) and is then extended in a way that the proposed estimation is robust to errors in the noise model estimation. The so-called instrumental variables (IV) method has been used in linear system identification for quite some time, and has recently seen its application in the identification of both nonlinear and LPV systems in the input-output (IO) form. The IV method reduces the bias in estimated LPV state-space models in case the noise model is not estimated properly or is unknown. In the proposed method of this paper, the attractive bias-variance trade-off properties of LS-SVMs are combined with statistical properties of IV-based methods to give robust estimates of the functional dependencies. Numerical examples are provided to compare the performances of the proposed IV-based technique with the LS-SVM-based LPV model identification methods.

5.1 INTRODUCTION

Linear parameter-varying (LPV) models offer an efficient framework for modeling nonlinear systems by representing the nonlinear model as a linear dynamic relation of the input and output variables where the relations themselves are dependent on the measurable time-varying signals, commonly known as the *scheduling variables*. This way, the scheduling signals take into account the varying operating conditions of the system. This simplicity of LPV models allows for the application of several linear control techniques to nonlinear systems represented by LPV models and opens the door for the application of powerful LPV control synthesis tools. Naturally then, LPV identification has attracted a lot of attention in the past decade [5], with different identification schemes developed for both input-output (IO) and LPV state-space models [36, 51, 77, 85].

Most identification methods in the literature for LPV state-space models fall under the category of parametric approach, where the scheduling dependencies of the state-space model matrices are assumed to be parameterized with respect to *known basis functions* [35]. This leads to over-parametrization of the model coefficients, causing a large variance in the estimates. On the other hand, an inappropriate selection of these functions is known to cause a structural bias [36]. For LPV state-space and bilinear models, parametric methods are mostly based on various subspace approaches. These methods usually require a high computational demand due to the enormous dimensions of the data matrices involved. The authors in [49] proposed a tractable way to reduce this curse of dimensionality for LPV state-space models with affine parameter dependence. A few more subspace-based methods were later published in [51, 52] among others.

Nonparametric methods provide an alternative that can avoid the bias-variance trade-off by obtaining nonparametric reconstruction of the scheduling dependencies in LPV models. With the emergence of kernel-based techniques, a new avenue of nonparametric identification, classification and data processing has appeared in the last two decades. Kernels are functions that enable us to perform linear operations in high-dimensional feature spaces, often mapping nonlinear dependencies efficiently using the so-called *kernel trick* [61]. This has sprouted the use of kernel-based techniques for solving various problems under the umbrella of LPV system identification [36, 74].

The identification approaches in [36,41,86,87] reported efficient kernel-based methods employing *least-squares support vector machines* (LS-SVM) for LPV-IO and state-space models; the results showed consistent estimates with an attractive bias-variance trade-off. An iterative mixed parametric method for LPV state-space identification was proposed recently in [57], in which the authors described the C matrix using a nonparametric LS-SVM model, while the A matrix was described by a parametric model.

While regularization-based SVMs provide an attractive bias-variance trade-off for efficient nonparametric identification, these methods suffer from severe restrictions imposed on noise, where the obtained estimates would be truly unbiased *w.r.t.* the noise only when writing the estimation problem into a regression form and the resulting noise process is white [86]. In order to obtain unbiased estimates for more generic noise conditions, *instrumental variables* (IV) have been employed in LTI identification theory in [88], and recently in the context of nonparametric identification for nonlinear ARX models in [86] and LPV IO models in [86].

In this paper, we present an LS-SVM-based nonparametric identification method for multi-input multi-output (MIMO) LPV state-space models. We further derive an IV-based SVM optimization problem to model the coefficient dependencies on the scheduling variables in the presence of colored noise, often correlated with the scheduling variables. The paper is arranged as follows. The problem is formulated in Section 5.2. An LS-SVM identification algorithm is formulated in Section 5.3. Section 5.4 discusses the conditions on the instruments and on the LPV structure of the noise. Section 5.5 formulates the proposed IV-based algorithm. Performance of the proposed method is demonstrated on a numerical example in Section 5.6. Concluding remarks are made in Section 5.7.

5.2 Problem Formulation

We now set to formally formulate the problem at hand, *i.e.*, identifying a discrete-time LPV model in the state-space form that can handle the presence of colored noise, as well as noise that is possibly correlated with the scheduling variables. Consider an LPV system represented by the

following discrete-time state-space model

$$\begin{aligned} x(k+1) &= A(p_k)x(k) + B(p_k)u(k) + v(k), \\ y(k) &= C(p_k)x(k) + z(k), \end{aligned} \tag{5.1}$$

where $k \in \mathbb{Z}$ denotes the discrete time instant, and matrices $A(p_k) \in \mathbb{R}^{n\times n}$, $B(p_k) \in \mathbb{R}^{n\times n_u}$, and $C(p_k) \in \mathbb{R}^{n_y\times n}$ are functions of time-varying scheduling variables $p(k) \in \mathbb{R}^{n_p}$, denoted as p_k for better readability. Variables $v(k) \in \mathbb{R}^n, z(k) \in \mathbb{R}^{n_y}$ are zero-mean, quasi-stationary stochastic noise processes, not necessarily white, but independent from $u(k)$. We aim at employing nonlinear kernel functions under the LS-SVM framework in order to estimate the functional dependencies of the state-space matrices on the scheduling variables. We can formulate the problem by writing the LPV state-space model (5.1) as follows

$$\begin{aligned} x(k+1) &= W_x\varphi_x^\top(k) + \varepsilon_v(k), \\ y(k) &= W_y\varphi_y^\top(k) + \varepsilon_z(k) \end{aligned} \tag{5.2}$$

where $\varepsilon_v(k), \varepsilon_z(k)$ are residual errors on the states and outputs, $W_x = [W_1 \ W_2] \in \mathbb{R}^{n\times 2n_H}$ and $W_y = W_3 \in \mathbb{R}^{n_y\times n_H}$ are weighting matrices, and $\varphi_x^\top(k) \in \mathbb{R}^{2n_H\times 1}$ and $\varphi_y^\top(k) \in \mathbb{R}^{n_H\times 1}$ are unknown regressors given by

$$\begin{aligned} \varphi_x^\top(k) &= [(\Phi_1(p_k)x(k))^\top \ (\Phi_2(p_k)u(k))^\top]^\top, \\ \varphi_y^\top(k) &= \Phi_3(p_k)x(k). \end{aligned} \tag{5.3}$$

In addition, n_H represents the dimension of a possibly infinite-dimensional feature space. From (5.2)-(5.3), we can gauge that $A(p_k) = W_1\Phi_1(p_k), B(p_k) = W_2\Phi_2(p_k)$, and $C(p_k) = W_3\Phi_3(p_k)$. We assume that the states are available for measurement, a possibility often encountered in applications like chemical processes. Estimating dependencies without state measurements is the subject of our ongoing research and is not covered in this paper. The problem, therefore, reduces to finding the dependency of $W_x\varphi_x^\top(k)$ and $W_y\varphi_y^\top(k)$ on p_k given the data $\{u(k), x(k), y(k), p(k)\}_{k=1}^N$, where N is the number of samples.

The estimates of the LPV state-space matrix functions can be obtained by minimizing the following cost function

$$\mathcal{J}(W_x, W_y, \varepsilon_v, \varepsilon_z) = \frac{1}{2}\left(\|W_x\|_{\mathrm{F}}^2 + \|W_y\|_{\mathrm{F}}^2\right) + \frac{1}{2}\left(\sum_{k=1}^{N} \varepsilon_v^\top(k)\Gamma\varepsilon_v(k) + \sum_{k=1}^{N} \varepsilon_z^\top(k)\Psi\varepsilon_z(k)\right), \tag{5.4}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, and $\Gamma = \mathrm{diag}(\gamma_1, \cdots, \gamma_n)$ and $\Psi = \mathrm{diag}(\psi_1, \cdots, \psi_{n_y})$ are diagonal weighting matrices on the residual errors $\varepsilon_v(k)$ and $\varepsilon_z(k)$ in (5.2); these weighting matrices are known as the regularization parameters. The above optimization can be solved by introducing *Lagrangian multipliers* and substituting the inner product $\Phi_i^\top \Phi_i$ using an *a priori* chosen nonlinear kernel function as shown in [36]. We define the Lagrangian as

$$\begin{aligned}\mathcal{L}(W_x, W_y, \boldsymbol{\alpha}, \boldsymbol{\beta}, \varepsilon_v, \varepsilon_z) &= \mathcal{J}(W_x, W_y, \varepsilon_v, \varepsilon_z) \\ &- \sum_{j=1}^{N} \alpha_j^\top \left\{W_x\varphi_x^\top(j) + \varepsilon_v(j) - x(j+1)\right\} - \sum_{j=1}^{N} \beta_j^\top \left\{W_y\varphi_y^\top(j) + \varepsilon_z(j) - y(j)\right\},\end{aligned} \tag{5.5}$$

where $\alpha_j \in \mathbb{R}^n$, $\beta_j \in \mathbb{R}^{n_y}$ are the Lagrange multipliers at the discrete time j. In order to solve for the global optimum, saddle points are obtained by solving the *Karush-Kuhn-Tucker* (KKT) conditions as follows

$$\frac{\partial \mathcal{L}}{\partial W_x} = 0 \Rightarrow W_x = \sum_{j=1}^{N} \alpha_j \varphi_x(j), \tag{5.6a}$$

$$\frac{\partial \mathcal{L}}{\partial W_y} = 0 \Rightarrow W_y = \sum_{j=1}^{N} \beta_j \varphi_y(j), \tag{5.6b}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_j} = 0 \Rightarrow \varepsilon_v(j) = x(j+1) - W_x\varphi_x^\top(j), \tag{5.6c}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = 0 \Rightarrow \varepsilon_z(j) = y(j) - W_y\varphi_y^\top(j), \tag{5.6d}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_v(j)} = 0 \Rightarrow \alpha_j = \Gamma\varepsilon_v(j), \tag{5.6e}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_z(j)} = 0 \Rightarrow \beta_j = \Psi\varepsilon_z(j). \tag{5.6f}$$

Substituting the relations (5.6a)-(5.6f) into (5.2), we can eliminate the primal decision variables and write

$$x(k+1) = W_x \varphi_x^\top(k) + \varepsilon_v(k) = \underbrace{\left\{ \sum_{j=1}^{N} \alpha_j \varphi_x(j) \right\}}_{W_x} \varphi_x^\top(k) + \underbrace{\Gamma^{-1}\alpha_k}_{\varepsilon_v(k)}, \tag{5.7}$$

$$y(k) = W_y \varphi_y^\top(k) + \varepsilon_z(k) = \underbrace{\left\{ \sum_{j=1}^{N} \beta_j \varphi_y(j) \right\}}_{W_y} \varphi_y^\top(k) + \underbrace{\Psi^{-1}\beta_k}_{\varepsilon_z(k)}. \tag{5.8}$$

Replacing the inner product $\Phi_i^\top(p_j)\Phi_i(p_k)$ by a kernel function $\bar{k}^i(p_j, p_k)$, we define kernel matrices Ω and Ξ as

$$[\Omega]_{j,k} = \varphi_x(j)\varphi_x^\top(k) = \sum_{i=1}^{2} z_i^\top(j)\bar{k}^i(p_j, p_k)z_i(k), \tag{5.9}$$

$$[\Xi]_{j,k} = \varphi_y(j)\varphi_y^\top(k) = x^\top(j)\bar{k}^3(p_j, p_k)x(k), \tag{5.10}$$

where $z_i(k) = x(k)$ and $z_i(k) = u(k)$ for $i = 1, 2$, respectively; the function $\bar{k}^i(\cdot, \cdot)$ denotes a nonlinear kernel function. While a wide variety of kernel functions exist in the literature to choose from, commonly used kernels include the *Radial Basis Function* (RBF), polynomial and sigmoid kernels among many others [61]. A typical RBF kernel, also known as the Gaussian kernel, is represented by

$$\bar{k}^i(p_j, p_k) = \exp\left(-\frac{\|p_j - p_k\|_2^2}{2\sigma_i^2} \right), \tag{5.11}$$

where σ_i is a free kernel parameter and $\|\cdot\|_2$ represents the Euclidean norm. By using (5.9) and (5.10), we can write (5.7)-(5.8) in a more compact form as

$$X = \boldsymbol{\alpha}\Omega + \Gamma^{-1}\boldsymbol{\alpha},$$

$$Y = \boldsymbol{\beta}\Xi + \Psi^{-1}\boldsymbol{\beta},$$

where $\Omega \in \mathbb{R}^{N\times N}$ and $\Xi \in \mathbb{R}^{N\times N}$ are the kernel matrices, $\boldsymbol{\alpha} = [\alpha_1 \cdots \alpha_N] \in \mathbb{R}^{n\times N}$ and $\boldsymbol{\beta} = [\beta_1 \cdots \beta_N] \in \mathbb{R}^{n_y\times N}$ are Lagrange multipliers, and $X = [x(1) \cdots x(N)] \in \mathbb{R}^{n\times N}$ and $Y = [y(1) \cdots y(N)] \in \mathbb{R}^{n_y\times N}$ contain the states and outputs for the N samples. The solution to

the above equations can be obtained as follows

$$\mathrm{vec}(\boldsymbol{\alpha}) = \left(I_N \otimes \Gamma^{-1} + \Omega^\top \otimes I_n\right)^{-1} \mathrm{vec}(X), \tag{5.12}$$

$$\mathrm{vec}(\boldsymbol{\beta}) = \left(I_N \otimes \Psi^{-1} + \Xi^\top \otimes I_{n_y}\right)^{-1} \mathrm{vec}(Y), \tag{5.13}$$

where $\otimes$ denotes Kronecker product and $\mathrm{vec}(\cdot)$ denotes vectorization function, which stacks subsequent columns in a matrix below one another; matrix I_N represents identity matrix of dimension N. The solutions (5.12)-(5.13) are obtained using the solution to the classical Sylvester equation. Once estimated, the estimate of the state-space matrices can be calculated by using (5.6a)-(5.6b) as

$$\hat{A}(\cdot) = W_1 \Phi_1(\cdot) = \sum_{k=1}^{N} \alpha_k x^\top(k) \bar{k}^1(p_k, \cdot), \tag{5.14a}$$

$$\hat{B}(\cdot) = W_2 \Phi_2(\cdot) = \sum_{k=1}^{N} \alpha_k u^\top(k) \bar{k}^2(p_k, \cdot), \tag{5.14b}$$

$$\hat{C}(\cdot) = W_3 \Phi_3(\cdot) = \sum_{k=1}^{N} \beta_k x^\top(k) \bar{k}^3(p_k, \cdot), \tag{5.14c}$$

giving us a nonparametric estimate of the state-space matrices. It is noteworthy that the parameter matrices W_i or the basis functions $\Phi_i(\cdot)$ are not accessible explicitly. What we estimate via nonlinear kernel functions is $W_i \Phi_i(\cdot)$.

Remark 5.1 *It is noteworthy here that since the objective function (5.4) and constraints (5.2) constitute a convex primal problem with linear equality constraints, strong duality holds, and there exists no duality gap between the dual solution (5.12)-(5.13) and the solution to the primal problem.*

5.4 Noise Model Estimation and Instrumental Variables

Consider that there exists W_{x0}, W_{y0} such that the following holds true for the true data-generating LPV system (5.1):

$$x(k+1) = W_{x0} \varphi_x^\top(k) + v_0(k), \tag{5.15a}$$

$$y(k) = W_{y0} \varphi_y^\top(k) + z_0(k). \tag{5.15b}$$

It was shown in [86] that LS-SVM-based optimization can provide consistent estimates under the assumption that W_{x0}, W_{y0} are bounded smooth functions and the condition that there is an absence of correlation between $\varphi_x(k)$ and $v_0(k)$, *e.g.*, when v_0 is a white noise process. The same holds true for correlation between $\varphi_y(k)$ and $z_0(k)$. This is not often the case in practice with sensor and actuator noise; noise is mostly colored and often correlated with the scheduling variables. To address the identification problem in the presence of colored noise, one could increase the complexity of the noise model; this, however, would lead to an increase in the complexity of the estimation and might in many cases, lead to a non-convex optimization problem. The so-called instrumental variable (IV) methods have provided a viable alternative by providing estimates that are robust to noise model estimation errors. Suppose that the colored noise $v(k)$, $z(k)$ have the following LPV structure

$$v(k) = \sum_{i=0}^{\infty} f_i(p_k, k) \cdot e(k-i), \tag{5.16a}$$

$$z(k) = \sum_{i=0}^{\infty} g_i(p_k, k) \cdot e(k-i), \tag{5.16b}$$

where $e(k)$ represents zero-mean white noise. The IV-based solutions provide unbiased estimates as long as the scheduling variables p_k are independent from $e(k)$ and the LPV filters (5.16a)-(5.16b) represent monic *infinite impulse response* (IIR) filters that are asymptotically stable. Recently, a nonparametric IV solution has been derived for nonlinear ARX models in [86] that provides unbiased estimates irrespective of the noise model. The idea follows the IV for linear systems in [88], which states that one can circumvent the problem of biased estimates due to noise modeling errors by introducing so-called instruments ζ_x and ζ_y such that

$$\mathbb{E}\{v_0(k)\zeta_x(k)\} = \mathbb{E}\{z_0(k)\zeta_y(k)\} = 0, \ \forall k \in \mathbb{Z}. \tag{5.17}$$

Consequently, we can introduce the instrument in the LS-SVM optimization problem derived in the previous section.

5.5 IV-SVM MODIFICATION

5.5.1 THE IV-SVM SCHEME

We now modify the cost function (5.4) such that the condition (5.17) is met; we do this by introducing instruments $\zeta_x(k),\ \zeta_y(k) \in \mathbb{R}^{1\times 2n_H}$. While the regressors $\varphi_x(k), \varphi_y(k)$ depend on the past samples of states and inputs, the instruments can be chosen by the user. The modified cost function can be written as

$$\mathcal{I}(W_x, W_y, \varepsilon_v, \varepsilon_z) = \frac{1}{2}\left(\|W_x\|_{\mathrm{F}}^2 + \|W_y\|_{\mathrm{F}}^2\right) + \frac{1}{2}\left(\sum_{k=1}^{N} \|\Gamma\varepsilon_v(k)\zeta_x(k)\|_{\mathrm{F}}^2 + \sum_{k=1}^{N} \|\Psi\varepsilon_z(k)\zeta_y(k)\|_{\mathrm{F}}^2\right), \tag{5.18}$$

The Lagrangian will now be defined as

$$\begin{aligned}\mathcal{L}(W_x, W_y, \boldsymbol{\alpha}, \boldsymbol{\beta}, \varepsilon_v, \varepsilon_z) &= \mathcal{I}(W_x, W_y, \varepsilon_v, \varepsilon_z) \\ &- \sum_{j=1}^{N} \alpha_j^\top \left\{W_x\varphi_x^\top(j) + \varepsilon_v(j) - x(j+1)\right\} - \sum_{j=1}^{N} \beta_j^\top \left\{W_y\varphi_y^\top(j) + \varepsilon_z(j) - y(j)\right\}. \end{aligned} \tag{5.19}$$

Like before, the stationary points for the Lagrangian are obtained by solving the KKT conditions as follows

$$\frac{\partial \mathcal{L}}{\partial W_x} = 0 \Rightarrow W_x = \sum_{j=1}^{N} \alpha_j \varphi_x(j), \tag{5.20a}$$

$$\frac{\partial \mathcal{L}}{\partial W_y} = 0 \Rightarrow W_y = \sum_{j=1}^{N} \beta_j \varphi_y(j), \tag{5.20b}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_j} = 0 \Rightarrow x(j+1) = W_x\varphi_x^\top(j) + \varepsilon_v(j), \tag{5.20c}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = 0 \Rightarrow y(j) = W_y\varphi_y^\top(j) + \varepsilon_z(j), \tag{5.20d}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_v(j)} = 0 \Rightarrow \varepsilon_v(j) = \left(2\zeta_x(j)\zeta_x^\top(j)\Gamma^\top\Gamma\right)^{-1}\alpha_j, \tag{5.20e}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_z(j)} = 0 \Rightarrow \varepsilon_z(j) = \left(2\zeta_y(j)\zeta_y^\top(j)\Psi^\top\Psi\right)^{-1}\beta_j. \tag{5.20f}$$

Next, like before, we eliminate the primal decision variables by substituting (5.20a)-(5.20f) into (5.2) and obtain the following:

$$\begin{aligned} x(k+1) &= W_x \varphi_x^\top(p_k) + \varepsilon_v(k) \\ &= \underbrace{\left\{ \sum_{j=1}^{N} \alpha_j \varphi_x(p_j) \right\}}_{W_x} \varphi_x^\top(p_k) + \underbrace{\left(2\zeta_x(k)\zeta_x^\top(k)\Gamma^\top\Gamma\right)^{-1}\alpha_k}_{\varepsilon_v(k)}, \end{aligned} \tag{5.21a}$$

$$\begin{aligned} y(k) &= W_y \varphi_y^\top(k) + \varepsilon_z(k) \\ &= \underbrace{\left\{ \sum_{j=1}^{N} \beta_j \varphi_y(j) \right\}}_{W_y} \varphi_y^\top(k) + \underbrace{\left(2\zeta_y(k)\zeta_y^\top(k)\Psi^\top\Psi\right)^{-1}\beta_k}_{\varepsilon_z(k)}. \end{aligned} \tag{5.21b}$$

The kernel matrices Ω and Ξ are defined as before, and (5.21a)-(5.21b) can be written in compact form as

$$X = \boldsymbol{\alpha}\Omega + (\Gamma^\top\Gamma)^{-1}\boldsymbol{\alpha}Z_x, \tag{5.22}$$

$$Y = \boldsymbol{\beta}\Xi + (\Psi^\top\Psi)^{-1}\boldsymbol{\beta}Z_y, \tag{5.23}$$

where

$$Z_x = \operatorname{diag}\Big(\left(2\zeta_x(1)\zeta_x^\top(1)\right)^{-1}, \cdots, \left(2\zeta_x(N)\zeta_x^\top(N)\right)^{-1} \Big), \tag{5.24}$$

$$Z_y = \operatorname{diag}\Big(\left(2\zeta_y(1)\zeta_y^\top(1)\right)^{-1}, \cdots, \left(2\zeta_y(N)\zeta_y^\top(N)\right)^{-1} \Big). \tag{5.25}$$

The solution to (5.22)-(5.23) can be obtained as follows

$$\begin{aligned} XZ_x^{-1} &= \boldsymbol{\alpha}\Omega Z_x^{-1} + (\Gamma^\top\Gamma)^{-1}\boldsymbol{\alpha}, \operatorname{vec}(\boldsymbol{\alpha}) \\ &= \underbrace{\left(I_N \otimes (\Gamma^\top\Gamma)^{-1} + (\Omega Z_x^{-1})^\top \otimes I_n\right)^{-1}}_{D_x^{IV}} \operatorname{vec}(XZ_x^{-1}), \end{aligned} \tag{5.26}$$

$$\begin{aligned} YZ_y^{-1} &= \boldsymbol{\beta}\Xi Z_y^{-1} + (\Psi^\top\Psi)^{-1}\boldsymbol{\beta}, \operatorname{vec}(\boldsymbol{\beta}) \\ &= \underbrace{\left(I_N \otimes (\Psi^\top\Psi)^{-1} + (\Xi Z_y^{-1})^\top \otimes I_{n_y}\right)^{-1}}_{D_y^{IV}} \operatorname{vec}(YZ_y^{-1}). \end{aligned} \tag{5.27}$$

Remark 5.2 *By choosing the regularization parameters as $\gamma_1 = \cdots = \gamma_n$ and $\psi_1 = \cdots = \psi_{n_y}$, one can, albeit at the cost of introducing conservatism in the estimation, simplify the solution to (5.22)-(5.23) by avoiding the calculation of inverses in D_x^{IV} and D_y^{IV}. Equations (5.22)-(5.23) can then be written in a simplified manner with a simpler solution as*

$$X = \boldsymbol{\alpha}\Omega + \boldsymbol{\alpha} Z_x \gamma_1^{-2} \implies \boldsymbol{\alpha} = X \underbrace{\left(\Omega + Z_x \gamma_i^{-2}\right)^{-1}}_{\hat{D}_x^{IV}}, \tag{5.28}$$

$$Y = \boldsymbol{\beta}\Xi + \boldsymbol{\beta} Z_y \psi_1^{-2} \implies \boldsymbol{\beta} = Y \underbrace{\left(\Xi + Z_y \psi_i^{-2}\right)^{-1}}_{\hat{D}_y^{IV}}. \tag{5.29}$$

Thus, calculation of the inverses $D_x^{IV} \in \mathbb{R}^{Nn \times Nn}$ and $D_y^{IV} \in \mathbb{R}^{Nn_y \times Nn_y}$ is replaced by calculating the inverses of matrices with much smaller dimensions, i.e. $\hat{D}_x^{IV} \in \mathbb{R}^{N \times N}$, $\hat{D}_y^{IV} \in \mathbb{R}^{N \times N}$.

5.5.2 Selection of the Instruments

To ensure unbiased estimates, the instruments $\zeta_x(k)$ and $\zeta_y(k)$ should be uncorrelated to the noise signals $v(k)$ and $z(k)$; in other words, condition (5.17) should hold true. As described in [88], the chosen instrument should be correlated with the regression variables, in this case, $\varphi_x(k)$ and $\varphi_y(k)$, but should be uncorrelated with the noise processes. Most instruments used in practice are generated by passing the past inputs through a filter; this is true in the LTI case where instruments can be generated using a *least squares* (LS) estimated model, while estimated LS-SVM-based nonlinear models have been used to generate instruments in the nonlinear [86] and LPV [86] cases in the input-output form. While the choice of an instrument generally remains an open problem, and depends highly on the structure of the system and the noise model, following the general principles outlined in [88] and later adopted in [86], we choose the instruments as

$$\zeta_x^\top(k) = [(\Phi_1(p_k)\xi_1(k))^\top \ (\Phi_2(p_k)\xi_2(k))^\top]^\top, \tag{5.30}$$

$$\zeta_y^\top(k) = \Phi_3(p_k)\xi_1(k), \tag{5.31}$$

where $\xi_i(k) = \hat{x}(k)$ and $\xi_i(k) = u(k)$ for $i = 1, 2$, respectively. Variable $\hat{x}(k)$ denotes noise-free states. Since we usually do not have access to noise-free measurements, $\hat{x}(k)$ can be considered to

Algorithm 3 IV-SVM algorithm for LPV state-space identification

Initialize: iter = 0

1: Given $\mathcal{D} = \{u(k), x(k), y(k), p(k)\}_{k=1}^{N}$, obtain an initial estimate $\hat{A}(p_k), \hat{B}(p_k)$, and $\hat{C}(p_k)$ by obtaining Lagrange multipliers $\boldsymbol{\alpha}^{(0)}, \boldsymbol{\beta}^{(0)}$ using LS-SVM solution (5.12)-(5.13) as proposed in Section 5.3; this model is labeled as $\mathcal{M}^0$.

2: Simulate the developed model in Step 1 to get estimates $\hat{x}(k)$.

3: iter = 1

while $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ do not converge according to (5.34) **do**

4: Define the instruments $\zeta_x(k), \zeta_y(k)$ as (5.30)-(5.31); compute (5.32)-(5.33) to obtain the matrices Z_x, Z_y.

5: Estimate the Lagrange multipliers $\boldsymbol{\alpha}^{\text{iter}}, \boldsymbol{\beta}^{\text{iter}}$ using IV-SVM solution (5.26)-(5.27) and solve (5.14a)-(5.14c) to obtain the model $\mathcal{M}^{\text{iter}}$.

6: Using $\mathcal{M}^{\text{iter}}$, generate the estimates $\hat{x}(k)$.

7: iter $\leftarrow$ iter + 1.

end while

be estimates of $x(k)$. The inner-product $\zeta_i(k)\zeta_i^\top(k)$ $(i = x, y)$ in (5.24)-(5.25) can now be replaced by kernel functions, giving us Z_x and Z_y as follows:

$$Z_x = \operatorname{diag}\left(\left(2\sum_{i=1}^{2} \xi_i^\top(1)\bar{k}^i(p_1,p_1)\xi_i(1)\right)^{-1}, \cdots, \left(2\sum_{i=1}^{2} \xi_i^\top(N)\bar{k}^i(p_N,p_N)\xi_i(N)\right)^{-1} \right), \tag{5.32}$$

$$Z_y = \operatorname{diag}\left(\left(2\xi_1^\top(1)\bar{k}^3(p_1,p_1)\xi_1(1)\right)^{-1}, \cdots, \left(2\xi_1^\top(N)\bar{k}^3(p_N,p_N)\xi_1(N)\right)^{-1} \right). \tag{5.33}$$

In case the chosen kernel function $\bar{k}^i$ is an RBF kernel, $\bar{k}^i(p_j, p_j) = 1$; however, we refrain from making this simplification in (5.32)-(5.33) since the derived expression is generic and may pertain to the choice of any kernel function, not necessarily RBF. The estimates of the state-space matrices $\hat{A}(\cdot)$, $\hat{B}(\cdot)$ and $\hat{C}(\cdot)$ can then be computed by using (5.14a)-(5.14c) in which $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the Lagrange multipliers obtained from the IV-SVM update (5.26)-(5.27), and $x(k)$ and $u(k)$ denote the recorded noisy states and inputs, respectively. The estimates $\hat{x}(k)$ used for the calculation of Z_x and Z_y can be considered to be those obtained from the developed SVM-based model, and can be improved iteratively until the solutions $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ converge according to the following criterion:

$$\mathcal{O}^{\text{iter}} = \left\|\boldsymbol{\alpha}^{\text{iter}} - \boldsymbol{\alpha}^{\text{iter}-1}\right\|_{\text{F}} + \left\|\boldsymbol{\beta}^{\text{iter}} - \boldsymbol{\beta}^{\text{iter}-1}\right\|_{\text{F}} \leq \epsilon, \tag{5.34}$$

where ϵ is a small number chosen for the stopping criterion of the iterative procedure. Detailed steps needed to implement the IV-SVM routine are described in Algorithm 3.

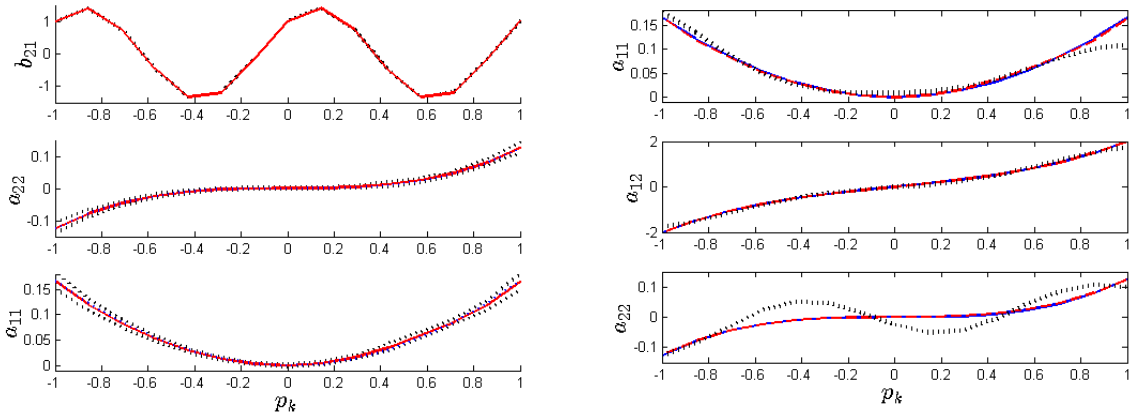

Figure 5.1: Example 1: (Left) Functional dependencies of elements $\{a_{11}\}$ and $\{a_{22}\}$, $\{b_{21}\}$ on the scheduling variables p_k as estimated by IV-SVM. The plot shows mean (red) and standard deviation (dotted black) data over the Monte-Carlo simulations; original functional dependencies are shown by dotted blue line. (Right) Functional dependencies estimation comparison between LS-SVM (dotted black), IV-SVM (solid red), and original functions (blue solid).

5.6 Simulation Results

The following numerical example of a second order discrete-time LPV state-space model is considered.

$$x(k+1) = A(p_k)x(k) + B(p_k)u(k) + v(k),$$

$$A(p_k) = \begin{bmatrix} \frac{1}{6}p_k^2 & p_k^3 + p_k \\ 0 & \frac{1}{8}p_k^3 \end{bmatrix}, B(p_k) = \begin{bmatrix} \mathrm{sat}(p_k) \\ \sin(q) + \cos(q) \end{bmatrix},$$

where $q = 2\pi p_k$, and

$$\mathrm{sat}(p_k) = \begin{cases} -0.5, & p_k < -0.5 \\ 0.5, & p_k > 0.5 \\ 0 & \text{otherwise.} \end{cases}$$

The signal $v(k)$ represents a colored noise correlated with the scheduling variable p_k and given by

$$v(k+1) = \begin{bmatrix} 0.95p_k & 0 \\ 0 & 0.95p_k \end{bmatrix} v(k) + \begin{bmatrix} -0.3p_k^3 \\ -0.15p_k \end{bmatrix} e(k),$$

Table 5.1: Monte-Carlo simulation results for output BFR

	Mean (BFR %)	Std. (BFR %)
LS-SVM	86.22	0.8247
IV-SVM	97.72	0.1596

where $e(k) \sim \mathcal{N}(0, \sigma_e^2)$ is a white noise sequence. Given the measurements of states $x(k)$, we are interested only in the estimation of $A(p_k)$ and $B(p_k)$, and hence, only Lagrange multipliers $\boldsymbol{\alpha}$ are sought; estimation of $C(p_k)$ would follow the same procedure for the estimation of $\boldsymbol{\beta}$ as outlined in the previous section. A total of 1000 samples of scheduling variables $p_k \in [-1, 1]$ are generated such that $p_k = \sin(0.8k)$. Uniformly distributed random inputs $u \in [-1, 1]$ are generated. White noise $e(k)$ is generated with standard deviation $\sigma_e = 0.15$, and $v(k)$ is calculated. This results in an average SNR of 12dB over the two states. Data is divided into 700 and 300 samples for estimation and validation, respectively. An RBF function is chosen for the kernel functions $\bar{k}^i$ with $\sigma_1 = \sigma_2 = 0.7$ for both the LS-SVM and the IV-SVM cases. Regularization parameters $\gamma_{1,2}$ are tuned to 2400 and 3200 for the two cases by searching over a grid space of hyperparameters in order to maximize the following *best fit rate* (BFR) in each case

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

where $\hat{x}$ represents simulated states of the estimated model and $\bar{x}$ denotes sample mean value of the states.

The proposed LS-SVM and IV-SVM algorithms are run and the Lagrange multipliers are estimated in each case. The performance of the algorithms is assessed on noise-less validation data set and average BFR values are calculated for the two states. BFR statistics over 50 runs of Monte-Carlo simulations are tabulated in Table 5.1.

Elements of the identified state-space matrices using IV-SVM show remarkable accuracy. The BFR values for the functional dependencies of these elements evaluated over the interval $[-1, 1]$ are tabulated in Table 5.2. Figure 5.1(a) shows three of these dependencies, namely, elements a_{11}, a_{22}, and b_{21}, with the dashed line showing mean functional value over all Monte-Carlo runs and the dotted black line showing the standard deviation. The improvement in output BFR values for the

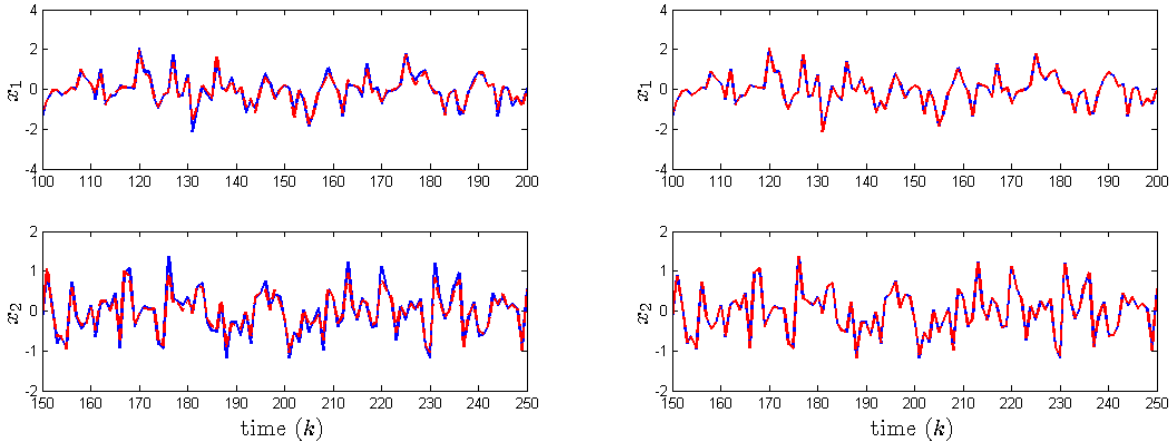

Figure 5.2: Validations results for LS-SVM estimation (left) and IV-SVM estimation (right) showing states of the original (solid blue) and identified (dashed red) LPV models.

Table 5.2: Monte-Carlo simulation results: Functional dependencies as estimated by the proposed IV-SVM-based method

Function	Mean (BFR %)	Std. (BFR %)
$a_{11}(p_k)$	79.775	0.015
$a_{12}(p_k)$	99.099	0.052
$a_{21}(p_k)$	96.012	0.078
$a_{22}(p_k)$	86.501	0.044
$b_{11}(p_k)$	97.315	0.101
$b_{21}(p_k)$	97.508	0.025

IV-SVM is reflected in the estimation of these functions as well. In Figure 5.1(b), we compare the mean functional values as estimated by the LS-SVM and IV-SVM algorithms. Owing to the colored nature of the noise, and the fact that the considered noise is correlated with the scheduling variables, some functional estimates like a_{22} in the LS-SVM case show a visible bias. Validation results for the two states are shown in Figure 5.2 for the two approaches.

5.7 Concluding Remarks

This paper has presented a nonparametric identification scheme for LPV state-space models under generic noise conditions. The proposed technique makes use of the nonparametric LS-SVM method that has shown encouraging estimation results for input-output LPV models, and further

incorporates the so-called "*instruments*" to induce robustness to noise modeling errors. Instrumental variables have proven to be effective in the LTI context and recently seen their extension to nonlinear ARX and polynomial LPV models. To gauge the performance of the proposed method, we have considered simulation studies with a high level of output noise, which was not only colored but also correlated with the scheduling variables. The results have shown encouraging improvements compared to LS-SVM based estimation methods in terms of providing unbiased estimates. Developing IV-SVM-based identification methods for LPV state-space models without the availability of state measurements is the focus of our ongoing research.

CHAPTER 6

STATE-SPACE LPV MODEL IDENTIFICATION USING KERNELIZED MACHINE LEARNING [1]

[1] Syed Z. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin: State-space LPV Model Identification Using Kernelized Machine Learning. 2016. *Submitted to Automatica*.

ABSTRACT

This paper presents a nonparametric method for identification of MIMO *linear parameter-varying* (LPV) models in the state-space form. The states are first estimated up to a similarity transformation via a nonlinear version of *canonical correlation analysis* (CCA) operating in a *reproducing kernel Hilbert space* (RKHS). This enables to reconstruct a minimal dimensional inference between past and future input-output and scheduling variables, making it possible to estimate a state-sequence consistent with the data. Once the states are estimated, an LS-SVM-based identification scheme is formulated, allowing to capture the dependency structure of the dynamic relation on the scheduling variables without requiring an explicit declaration of these often unknown dependencies; instead, it only requires the selection of nonlinear kernel functions and the tuning of the associated hyper-parameters.

6.1 INTRODUCTION

Linear Parameter-Varying (LPV) model identification has attracted a lot of attention within the system identification community in the recent past. Although a significant progress on the identification of LPV systems with *input-output* (IO) models has been achieved [35, 36, 56], identification in an LPV state-space form remains challenging with several open problems.

The main streams of LPV control synthesis approaches in the literature are derived from LPV *state-space* (SS) representations. However, the bulk of discrete-time LPV identification and modeling is often carried out under an IO structure. Therefore, a possible approach would be to transform available LPV-IO models to LPV-SS form. However, such a transformation is complicated as the conversion to equivalent SS models often results in dynamic dependence of the state-space matrices on the scheduling variables while approximative "realizations" deform the dynamical relations between the inputs and outputs, often leading to high output errors [42]. Allowing for such a dynamic dependence increases the complexity of the transformed LPV-SS model thereby making controller synthesis more difficult or even computationally infeasible. It is for this reason that LPV state-space models directly identified from input and output data are of prime importance.

Broadly speaking, LPV identification methods can be categorized into parametric and nonparametric methods. In parametric identification of LPV models, the assumption is made that the scheduling dependencies of the model coefficients are known *a priori* [35]. However, in practice, selecting adequate functions to parameterize these dependencies is a non-trivial task where often one tries to include a wide array of basis functions to ensure that the process dynamics are captured. This often leads to over-parametrization of the model coefficients [47], causing a large variance in the estimates. On the other hand, an inappropriate selection of these functions causes structural bias [36]. Examples of parametric LPV-SS identification include subspace identification methods published in [49, 51, 53]. These methods pertain to systems that can be modeled with affine parameter-dependence, and are usually suitable for only low-dimensional cases. For an overview of other LPV-SS identification schemes, see [56]. An alternative approach with an attractive bias-variance trade-off is to obtain a nonparametric reconstruction of the scheduling depen-

dencies in LPV models. Kernel-based nonparametric identification techniques have demonstrated encouraging results for LPV-IO models in [36, 41, 47], among others; however, very few nonparametric methods for state-space model structures have been reported. A mixed parametric method based on *least-squares support vector machine* (LS-SVM) was proposed recently in [57]. In this work, the state matrix A is described by a parametric model, while the state-readout matrix C is described by a nonparametric one. The problem of selecting basis functions therefore is solved only partially. Additionally, the work [57] focuses only on *single-input single-output* (SISO) LPV-SS models. In our recent work [87], we proposed an LS-SVM-based LPV-SS identification method for *multi-input multi-output* (MIMO) systems. Further improvement was presented in [89] by incorporating instrumental variables, making the technique robust to the presence of colored noise. A limiting factor in both of these works was the assumption of the availability of state measurements, which, most often, is not the case in practical situations.

In this paper, we present an LS-SVM-based nonparametric method for MIMO LPV-SS model identification. The proposed technique works in two steps; first, LS-SVM-based nonlinear *canonical correlation analysis* (CCA) is used to estimate the states of the data-generating system from inputs, outputs, and scheduling variables data. The estimated states are then used with the measured data to identify the state-space matrices of an LPV-SS model of the data-generating system with no assumption made *a priori* on the scheduling dependency structure. The main contribution of this paper lies in the unique formulation of kernelized CCA and LS-SVM for identification purposes such that the linearity structure of LPV state-space models is retained. The paper is arranged as follows. The problem is formulated in Section 6.2. The use of correlation analysis for the estimation of the states is derived and explained in Section 6.3. Section 6.4 details the LS-SVM-based identification algorithm. To demonstrate the capabilities of the developed approach, a simulation study is provided in Section 6.6. Finally, concluding remarks are made in Section 6.7.

6.2 Problem Formulation and Preliminaries

Consider an LPV system represented by the following discrete-time state-space innovation noise model

$$x_{k+1} = A(p_k)x_k + B(p_k)u_k + K(p_k)e_k, \tag{6.1a}$$

$$y_k = C(p_k)x_k + D(p_k)u_k + e_k, \tag{6.1b}$$

where $k \in \mathbb{Z}$ denotes discrete-time, and $A(p_k) \in \mathbb{R}^{n\times n}$, $B(p_k) \in \mathbb{R}^{n\times n_{\mathrm{u}}}$, $K(p_k) \in \mathbb{R}^{n\times n_{\mathrm{y}}}$, $C(p_k) \in \mathbb{R}^{n_{\mathrm{y}}\times n}$, and $D(p_k) \in \mathbb{R}^{n_{\mathrm{y}}\times n_{\mathrm{u}}}$ are smooth functions of time-varying scheduling variables $p_k \in \mathbb{P} \subset \mathbb{R}^{n_{\mathrm{p}}}$ with $\mathbb{P}$ being compact. Variables $u_k \in \mathbb{R}^{n_{\mathrm{u}}}$, $y_k \in \mathbb{R}^{n_{\mathrm{y}}}$, and $x_k \in \mathbb{R}^n$ represent the inputs, outputs, and states of the system at time k, while $e_k \in \mathbb{R}^{n_{\mathrm{y}}}$ is a stochastic white noise process. By substituting $e_k = y_k - C(p_k)x_k + D(p_k)u_k$ in (6.1a), we can re-write the above set of equations as

$$x_{k+1} = \tilde{A}(p_k)x_k + \tilde{B}(p_k)u_k + K(p_k)y_k, \tag{6.2a}$$

$$y_k = C(p_k)x_k + D(p_k)u_k + e_k, \tag{6.2b}$$

where $\tilde{A}(p_k) = A(p_k) - K(p_k)C(p_k)$, and $\tilde{B}(p_k) = B(p_k) - K(p_k)D(p_k)$. Similar to the LTI case, (6.2) must be asymptotically stable in the deterministic sense (even if asymptotic stability of (6.1) is not required), otherwise identification of (6.1) is ill-posed due to the divergence of the variance of the resulting stochastic process. Our aim is to develop a kernelized LS-SVM approach to estimate the functional dependencies of the state-space matrices on the scheduling variables, given only the measurements of $\mathcal{D} = \{u_k, y_k, p_k\}_{k=1}^N$, where N is the number of samples.

6.3 A KCCA-based Approach for State Estimation

To achieve the aforedescribed objective, first we aim at reconstructing an estimate of the state sequence $\{x_k\}_{k=1}^N$ *compatible* with $\mathcal{D}$.

6.3.1 Canonical Correlation Analysis

Canonical correlation analysis (CCA) is a statistical method mainly used to determine linear relations among several variables. Given two sets of variables, $u \in \mathbb{R}^{n_{\mathrm{u}}}$ and $y \in \mathbb{R}^{n_{\mathrm{y}}}$, with N samples of each collected in $U \in \mathbb{R}^{N \times n_{\mathrm{u}}}$ and $Y \in \mathbb{R}^{N \times n_{\mathrm{y}}}$, CCA aims at finding vectors $v_j \in \mathbb{R}^{n_{\mathrm{u}}}$ and $w_j \in \mathbb{R}^{n_{\mathrm{y}}}$ in order to maximize correlation between projected variables Uv_j and Yw_j, also known as variates [90]. This leads to the following constrained optimization problem

$$\max_{v_j, w_j} v_j^\top U^\top Y w_j, \quad \text{s.t.} \quad v_j^\top U^\top U v_j = w_j^\top Y^\top Y w_j = 1.$$

The optimization solved in the dual form leads to a generalized eigenvalue problem. For more details, see [91]. CCA only uses second order information to identify the relation between data sets, an important consequence of which is that CCA and its regularized versions are easily *kernelizable* and can handle nonlinear relationships [92].

6.3.2 Regularized kernel CCA for LPV state estimation

Kernelized CCA can find nonlinear relations between the inputs and outputs of the data-generating LPV model (6.1). This is done by modifying the linear CCA presented above and incorporating kernel functions to map the nonlinearly varying scheduling variables into an *reproducing kernel Hilbert space* (RKHS), where classical CCA is applied [93]. The goal here is to estimate, from a finite set of inputs, outputs, and scheduling variables data, the states associated with (6.1). The main idea behind this is that the states are the minimal interface between the past and future input output and scheduling variable data; therefore, the states are expected to be the representative of the past behavior needed to determine the future behavior [91].

Define the past scheduling variables $\bar{p}_k^d \in \mathbb{R}^{dn_{\mathrm{p}}}$ and future scheduling variables $\bar{p}_{k+d}^d \in \mathbb{R}^{dn_{\mathrm{p}}}$ w.r.t. time instant k as

$$\bar{p}_k^d := \begin{bmatrix} p_{k-d} & \cdots & p_{k-1} \end{bmatrix}^\top, \tag{6.3a}$$

$$\bar{p}_{k+d}^d := \begin{bmatrix} p_k & \cdots & p_{k+d-1} \end{bmatrix}^\top, \tag{6.3b}$$

$$
\begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+d-1} \end{bmatrix} = \underbrace{\begin{bmatrix} C(p_k) \\ C(p_{k+1})\tilde{A}(p_k) \\ \vdots \\ C(p_{k+d-1})\prod_{l=2}^{d}\tilde{A}(p_{k+d-l}) \end{bmatrix}}_{(\mathcal{O}_{\mathrm{f}}^{d}\diamond p)(k)} x_k
$$

$$
+ \underbrace{\begin{bmatrix} D(p_k) & 0 & \cdots & 0 \\ C(p_{k+1})\tilde{B}(p_k) & D(p_{k+1}) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ C(p_{k+d-1})\prod_{l=2}^{d-1}\tilde{A}(p_{k+d-l})\tilde{B}(p_k) & C(p_{k+d-1})\prod_{l=2}^{d-2}\tilde{A}(p_{k+d-l})\tilde{B}(p_{k+1}) & \cdots & D(p_{k+d-1}) \end{bmatrix}}_{(\mathcal{H}_{\mathrm{f}}^{d}\diamond p)(k)} \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+d-1} \end{bmatrix}
$$

$$
+ \underbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 \\ C(p_{k+1})K(p_k) & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ C(p_{k+d-1})\prod_{l=2}^{d-1}\tilde{A}(p_{k+d-l})K(p_k) & C(p_{k+d-1})\prod_{l=2}^{d-2}\tilde{A}(p_{k+d-l})K(p_{k+1}) & \cdots & 0 \end{bmatrix}}_{(\mathcal{L}_{\mathrm{f}}^{d}\diamond p)(k)} \begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+d-1} \end{bmatrix} + \begin{bmatrix} e_k \\ e_{k+1} \\ \vdots \\ e_{k+d-1} \end{bmatrix},
\tag{6.4}
$$

where d denotes the number of past and future samples. Past and future inputs and outputs $\bar{u}_k^d \in \mathbb{R}^{dn_{\mathrm{u}}}$, $\bar{y}_k^d \in \mathbb{R}^{dn_{\mathrm{y}}}$, $\bar{u}_{k+d}^d \in \mathbb{R}^{dn_{\mathrm{u}}}$, and $\bar{y}_{k+d}^d \in \mathbb{R}^{dn_{\mathrm{y}}}$ are defined in a similar way. Further, we also define $\bar{z}_k^d = \begin{bmatrix} \bar{u}_k^d \\ \bar{y}_k^d \end{bmatrix}$, $\bar{z}_{k+d}^d = \begin{bmatrix} \bar{u}_{k+d}^d \\ \bar{y}_{k+d}^d \end{bmatrix} \in \mathbb{R}^{d(n_{\mathrm{u}}+n_{\mathrm{y}})}$. The future outputs of the LPV innovation form (6.2) can be written in the observability form[2] (6.4), described compactly as

$$
\bar{y}_{k+d}^d = (\mathcal{O}_{\mathrm{f}}^d \diamond p)(k) \cdot x_k + (\mathcal{H}_{\mathrm{f}}^d \diamond p)(k) \cdot \bar{u}_{k+d}^d + (\mathcal{L}_{\mathrm{f}}^d \diamond p)(k) \cdot \bar{y}_{k+d}^d + \bar{e}_{k+d}^d, \tag{6.5}
$$

where $(\mathcal{O}_{\mathrm{f}}^d \diamond p)(k) \in \mathbb{R}^{dn_{\mathrm{y}} \times n}$ is the time-varying d-step forward observability matrix[3] at time k along the scheduling trajectory p, $(\mathcal{H}_{\mathrm{f}}^d \diamond p)(k) \in \mathbb{R}^{dn_{\mathrm{y}} \times dn_{\mathrm{u}}}$ is a forward Toeplitz matrix based on the *Infinite Impulse Response* (IIR) coefficients of (6.2), and $(\mathcal{L}_{\mathrm{f}}^d \diamond p)(k) \in \mathbb{R}^{dn_{\mathrm{y}} \times dn_{\mathrm{y}}}$ is a lower

[2]It should be noted here that, in (6.4), we use a left precedence for multiplication, *e.g.*, for $k = d = 4$, $\prod_{l=2}^{d} = \tilde{A}(p_{k+d-l}) = \tilde{A}(p_6)\tilde{A}(p_5)\tilde{A}(p_4)$.

[3]The notation $(\mathcal{O}_{\mathrm{f}}^d \diamond p)(k)$ corresponds to the evaluation of $\mathcal{O}_{\mathrm{f}}^d$ along p at time instant k and is used as a shorthand to express the dynamic dependence of the corresponding matrix functions, e.g., $\mathcal{O}_{\mathrm{f}}^d$ at time k depends on the instantaneous and future sample values of the scheduling variables, *i.e.*, $p_k, ..., p_{k+d-1}$.

triangle matrix. Variable $\bar{e}^d_{k+d}$ denotes a segment of the sample path of e_k. Without the loss of generality, it is assumed that (6.2) is *structurally observable* in the deterministic sense[4]

Definition 6.3.1 *The LPV-SS representation* (6.2) *with state-dimension n is called structurally observable, if there exists a scheduling trajectory $p \in \mathbb{P}^{\mathbb{Z}}$, such that the n-step observability matrix $(\mathcal{O}^n_{\mathrm{f}} \diamond p)(k)$ is full (column) rank for all $k \in \mathbb{Z}$.*

Let $\mathcal{P} \subseteq \mathbb{P}^{\mathbb{Z}}$ be the set of all scheduling trajectories p such that $\operatorname{rank}\big((\mathcal{O}^n_{\mathrm{f}} \diamond p)(k)\big) = n$ for all $k \in \mathbb{Z}$. Then, in order to guarantee that $\mathcal{O}^d_{\mathrm{f}}$ is injective in (6.5), it is assumed that d is chosen such that $d \geq n$ and $p \in \mathcal{P}$ in the given data set $\mathcal{D}$. In other words, we assume that the to-be-estimated model representation is observable along the given trajectory of p on all intervals of length d, which corresponds to a *persistency of excitation* (PE) condition on p. Dropping the dynamic dependence argument for notational ease, the state sequence statistics can be given by

$$x_k = \big(\mathcal{O}^d_{\mathrm{f}}(k)\big)^{\dagger} \Big(\big(I - \mathcal{L}^d_{\mathrm{f}}(k)\big)\bar{y}^d_{k+d} - \mathcal{H}^d_{\mathrm{f}}(k)\bar{u}^d_{k+d}\Big) - \big(\mathcal{O}^d_{\mathrm{f}}(k)\big)^{\dagger}\bar{e}^d_{k+d}, \tag{6.6}$$

where $(\cdot)^{\dagger}$ indicates the Moore-Penrose pseudo-inverse. Since e is an *independent and identically distributed* (i.i.d) zero-mean process, the expected value of the last term on the right-hand side will be zero, giving us, in the conditional mean sense, the following state estimates

$$\hat{x}_k = \underbrace{\big(\mathcal{O}^d_{\mathrm{f}}(k)\big)^{\dagger} \begin{bmatrix} -\mathcal{H}^d_{\mathrm{f}}(k) & I - \mathcal{L}^d_{\mathrm{f}}(k) \end{bmatrix}}_{\varphi_{\mathrm{f}}(\bar{p}^d_{k+d})} \bar{z}^d_{k+d}. \tag{6.7}$$

Note that due to injectivity of the linear map $\mathcal{O}^d_{\mathrm{f}}(k)$, (6.7) can be used for an unbiased data-based estimate of any state trajectory compatible with (6.1) for any arbitrary u, sample path e and any $p \in \mathcal{P}$. Note that (6.7) is a non-minimal variance estimator; however, its asymptotic properties are determined by an equivalent *linear time-varying* Kalman filtering problem [94]. A similar expression can be derived for the past outputs giving us

$$\bar{y}^d_k = (\mathcal{O}^d_{\mathrm{p}} \diamond p)(k) \cdot x_{k-d} + (\mathcal{H}^d_{\mathrm{p}} \diamond p)(k) \cdot \bar{u}^d_k + (\mathcal{L}^d_{\mathrm{p}} \diamond p)(k) \cdot \bar{y}^d_k + \bar{e}^d_k, \tag{6.8}$$

[4]If the assumption of structural observability is not satisfied, then (6.2) is not state-minimal and under some mild assumptions on the class of functional dependencies of the associated matrix functions, there exists a structurally observable LPV-SS realization of the underlying system with equivalent IO map.

where $\mathcal{O}_{\mathrm{p}}^d$ is a d-step *constructibility* matrix, similar to $\mathcal{O}_{\mathrm{f}}^d$, but formulated backwards in time, such that $(\mathcal{O}_{\mathrm{p}}^d \diamond p)(k)$ depends on $p_{k-d}, ..., p_{k-1}$. Likewise, $\mathcal{H}_{\mathrm{p}}^d$ is a backward d-step Toeplitz matrix. We follow a similar path to estimate the states from the above expression and obtain

$$\hat{x}_{k-d} = \underbrace{\left(\mathcal{O}_{\mathrm{p}}^d(k)\right)^\dagger \begin{bmatrix} -\mathcal{H}_{\mathrm{p}}^d(k) & I^* \end{bmatrix}}_{\varphi_{\mathrm{p}}(\bar{p}_k^d)} \bar{z}_k^d. \tag{6.9}$$

Even if both $\varphi_{\mathrm{f}}(\bar{p}_{k+d}^d)$ and $\varphi_{\mathrm{p}}(\bar{p}_k^d)$ are unknown mappings (defined by the to-be-estimated matrix functions), the relations (6.7) and (6.9) allow the use of CCA for the estimation of $\hat{x}_k$. As these maps at least have polynomial dependence on $\bar{p}_{k+d}^d$ and $\bar{p}_k^d$ in case of affine dependence of the original matrix functions of (6.1), a tailor-made kernelized formulation of CCA is required for the underlying estimation problem. To develop this formulation, we define the past and future data sets $\Phi_{\mathrm{p}}, \Phi_{\mathrm{f}} \in \mathbb{R}^{N \times n_{\mathrm{G}}}$ as

$$\Phi_{\mathrm{p}} := \begin{bmatrix} \varphi_{\mathrm{p}}(\bar{p}_1^d)\bar{z}_1^d & \cdots & \varphi_{\mathrm{p}}(\bar{p}_N^d)\bar{z}_N^d \end{bmatrix}^\top, \tag{6.10a}$$

$$\Phi_{\mathrm{f}} := \begin{bmatrix} \varphi_{\mathrm{f}}(\bar{p}_{1+d}^d)\bar{z}_{1+d}^d & \cdots & \varphi_{\mathrm{f}}(\bar{p}_{N+d}^d)\bar{z}_{N+d}^d \end{bmatrix}^\top, \tag{6.10b}$$

where $\varphi_{\mathrm{p}} : \mathbb{R}^{dn_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{G}} \times d(n_{\mathrm{u}}+n_{\mathrm{y}})}$ and $\varphi_{\mathrm{f}} : \mathbb{R}^{dn_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{G}} \times d(n_{\mathrm{u}}+n_{\mathrm{y}})}$ represent unknown feature maps that respectively map the past and future scheduling variables into an RKHS $\mathcal{G}_{\breve{k}}$ defined uniquely by a symmetric and positive definite kernel function $\breve{k} : \mathbb{R}^{dn_{\mathrm{p}}} \times \mathbb{R}^{dn_{\mathrm{p}}} \to \mathbb{R}$ (for details, see [95]); variable n_{G} represents the dimension of this possibly infinite-dimensional feature space. The kernel function, with arguments in the input space $\mathbb{R}^{dn_{\mathrm{p}}}$, corresponds to an inner product in the RKHS as

$$\left[\varphi_{\mathrm{p}}(\bar{p}_k^d)\right]_i^\top \left[\varphi_{\mathrm{p}}(\bar{p}_j^d)\right]_i = \breve{k}\left(\bar{p}_k^d, \bar{p}_j^d\right).$$

Now the CCA problem corresponding to the equivalent representation (6.2) becomes

$$\max_{v_j, w_j} v_j^\top \Phi_{\mathrm{f}}^\top \Phi_{\mathrm{p}} w_j \quad \text{s.t.} \quad v_j^\top \Phi_{\mathrm{f}}^\top \Phi_{\mathrm{f}} v_j = w_j^\top \Phi_{\mathrm{p}}^\top \Phi_{\mathrm{p}} w_j = 1, \tag{6.11}$$

where $v_j \in \mathbb{R}^{n_{\mathrm{G}}}$, $w_j \in \mathbb{R}^{n_{\mathrm{G}}}$ with $j \in \{1, \dots N\}$ are directions in the feature space $\mathcal{G}_{\breve{k}}$, along which the projections of the future and past data have maximum correlation. Problem (6.11) above represents the kernel CCA problem of Bach and Jordan [93], a particular disadvantage of which

is the lack of regularization. An improved and regularized version based on LS-SVM was later introduced in [90], the primal version of which, adapted to the case of the LPV-SS model (6.2), is given as

$$\max_{v_j,w_j} \mathcal{J}(v_j,w_j,s,r) = \gamma \sum_{k=1}^{N} \left(s_k r_k - \nu_{\mathrm{f}} \frac{1}{2} s_k^2 - \nu_{\mathrm{p}} \frac{1}{2} r_k^2 \right) - \frac{1}{2} v_j^\top v_j - \frac{1}{2} w_j^\top w_j,$$
$$\text{s.t. } \; s_k = v_j^\top \varphi_{\mathrm{f}}(\bar{p}_{k+d}^d) \bar{z}_{k+d}^d, \; r_k = w_j^\top \varphi_{\mathrm{p}}(\bar{p}_k^d) \bar{z}_k^d, \tag{6.12}$$

for $k = 1, ..., N$, where $\gamma, \nu_{\mathrm{f}}, \nu_{\mathrm{p}} \in \mathbb{R}^+$ are positive hyper-parameters needed to be chosen. The main advantage of this improved CCA lies in the introduction of the last two terms in the cost function, which help to regularize v_j, w_j, making sure they do not become arbitrarily large. Writing the above problem in a dual form, we define the *Lagrangian* as given in (6.13), where $\eta_j = [\eta_j^1 \cdots \eta_j^N]^\top \in \mathbb{R}^N$ and $\kappa_j = [\kappa_j^1 \cdots \kappa_j^N]^\top \in \mathbb{R}^N$ are Lagrange multipliers. The optimality conditions are obtained via solving the *Karush-Kuhn-Tucker* (KKT) conditions, *i.e.*, finding the derivatives $\frac{\partial \mathcal{L}}{\partial v_j}, \frac{\partial \mathcal{L}}{\partial w_j}, \frac{\partial \mathcal{L}}{\partial s_k}, \frac{\partial \mathcal{L}}{\partial r_k}, \frac{\partial \mathcal{L}}{\partial \eta_j^k}, \frac{\partial \mathcal{L}}{\partial \kappa_j^k}$ and equating them to zero, which are not shown here due to the space limitations.

$$\mathcal{L}(v_j,w_j,s,r,\eta_j,\kappa_j) = \mathcal{J}(v_j,w_j,s,r)$$
$$- \sum_{k=1}^{N} \eta_j^k \left(s_k - v_j^\top \varphi_{\mathrm{f}}(\bar{p}_{k+d}^d) \bar{z}_{k+d}^d \right) - \sum_{k=1}^{N} \kappa_j^k \left(r_k - w_j^\top \varphi_{\mathrm{p}}(\bar{p}_k^d) \bar{z}_k^d \right), \tag{6.13}$$

Using these conditions to eliminate the primal decision variables v_j, w_j, s_k, r_k, and substituting $\lambda_j = 1/\gamma$, the above problem can be simplified to a *regularized* generalized eigenvalue problem as

$$K_{\mathrm{pp}} \kappa_j = \lambda_j (\nu_{\mathrm{f}} K_{\mathrm{ff}} + I) \eta_j, \tag{6.14a}$$
$$K_{\mathrm{ff}} \eta_j = \lambda_j (\nu_{\mathrm{p}} K_{\mathrm{pp}} + I) \kappa_j, \tag{6.14b}$$

where $K_{\mathrm{ff}} = \Phi_{\mathrm{f}} \Phi_{\mathrm{f}}^\top$ and $K_{\mathrm{pp}} = \Phi_{\mathrm{p}} \Phi_{\mathrm{p}}^\top$ are kernel matrices associated with the chosen kernel function $\breve{k}$ generating $\mathcal{G}_{\breve{k}}$. Replacing the inner product of the feature maps of $\mathcal{G}_{\breve{k}}$ with $\breve{k}$ implies that the elements of the kernel matrices are

$$[K_{\mathrm{ff}}]_{l,m} = \bar{z}_{l+d}^{d\,\top} \breve{k}(\bar{p}_{l+d}^d, \bar{p}_{m+d}^d) \bar{z}_{m+d}^d, \tag{6.15a}$$
$$[K_{\mathrm{pp}}]_{l,m} = \bar{z}_l^{d\,\top} \breve{k}(\bar{p}_l^d, \bar{p}_m^d) \bar{z}_m^d. \tag{6.15b}$$

Kernel functions can be chosen from a variety of different functions, including, but not limited to *polynomial kernels* $\breve{k}(p_i, p_j) = ((p_i \cdot p_j) + 1)^q$, and *radial basis function* (RBF) $\breve{k}(p_i, p_j) = \exp\left(-\frac{\| p_i - p_j \|^2}{\sigma^2}\right)$. Parameters $q \in \mathbb{N}$ and $\sigma \in \mathbb{R}^+$ denote the degree of the polynomial and the spread of the RBF function; these are essentially tuning parameters chosen by the user [61]. Thus, by solving (6.14b), one can find the primal decision variables $v_j = \Phi_{\mathrm{f}}^\top \eta_j$ and $w_j = \Phi_{\mathrm{p}}^\top \kappa_j$, which were obtained by solving the KKT conditions $\frac{\partial \mathcal{L}}{\partial v_j} = 0, \frac{\partial \mathcal{L}}{\partial w_j} = 0$. This solution is then used to parameterize the observability-form-based estimate of the state evolution of (6.1) w.r.t. $\mathcal{D}$.

As $K_{\mathrm{pp}}, K_{\mathrm{ff}} \in \mathbb{R}^{N \times N}$, the *regularized* generalized eigenvalue problem (6.14b) can have up to $2N$ different solutions $\eta_j, \kappa_j, j = 1, \ldots, 2N$. Using these solutions, we can find the primal decision variables $v_j = \Phi_{\mathrm{f}}^\top \eta_j$ and $w_j = \Phi_{\mathrm{p}}^\top \kappa_j$. Since each of these solutions gives a direction in the feature space correlating past data with the future data, each solution can give us one component, *i.e.*, the j^{th} component, of the state variable that ties the past behavior to the future. Therefore, the estimate of a compatible state vector at time instant k follows using the j^{th} solution to the KCCA problem as

$$\breve{x}_k^j = v_j^\top \varphi_{\mathrm{f}}(\bar{p}_{k+d}^d) \bar{z}_{k+d}^d. \tag{6.16}$$

Substituting above the earlier solved KKT condition gives $v_j = \Phi_{\mathrm{f}}^\top \eta_j$, and replacing the feature space dot-product $\varphi_{\mathrm{f}}^\top(\cdot)\varphi_{\mathrm{f}}(\cdot)$ with a kernel function $\breve{k}(\cdot, \cdot)$, we obtain

$$\breve{x}_k^j = \eta_j^\top \begin{bmatrix} \bar{z}_{1+d}^{d\,\top} \breve{k}(\bar{p}_{1+d}^d, \bar{p}_{k+d}^d) \\ \vdots \\ \bar{z}_{N+d}^{d\,\top} \breve{k}(p_{N+d}^d, \bar{p}_{k+d}^d) \end{bmatrix} \bar{z}_{k+d}^d. \tag{6.17}$$

Similarly, $w_j = \Phi_{\mathrm{p}}^\top \kappa_j$ gives the estimated j^{th} component of the state vector at time $k - d$ as

$$\breve{x}_{k-d}^j = w_j^\top \varphi_{\mathrm{p}}(\bar{p}_k^d) \bar{z}_k^d = \kappa_j^\top \begin{bmatrix} \bar{z}_1^{d\,\top} \bar{k}(\bar{p}_1^d, \bar{p}_k^d) \\ \vdots \\ \bar{z}_N^{d\,\top} \bar{k}(p_N^d, \bar{p}_k^d) \end{bmatrix} \bar{z}_k^d. \tag{6.18}$$

Remark 6.1 *All $2N$ solutions of the regularized generalized eigenvalue problem (6.14b) in terms of normalized eigenvectors can be analytically calculated via the following economical* singular

value decomposition (SVD)

$$\left[\begin{array}{c:c} \nu_{\mathrm{f}} K_{\mathrm{ff}} + I & 0 \\ \hdashline 0 & \nu_{\mathrm{p}} K_{\mathrm{pp}} + I \end{array}\right]^{-1} \left[\begin{array}{c:c} 0 & K_{\mathrm{pp}} \\ \hdashline K_{\mathrm{ff}} & 0 \end{array}\right] = W \Sigma \left[\begin{array}{c} V_1 \\ \hdashline V_2 \end{array}\right]^{\top} \tag{6.19}$$

where Σ is a diagonal matrix containing all non-zero singular values, and the corresponding solutions are $\eta_j = [V_1]_j$ and $\kappa_j = [V_2]_j$ with $[\cdot]_j$ denoting the j^{th} column of the resulting matrices. Relying on the rank revealing property of the SVD, an economical realization, i.e., automatic state-order selection, can be achieved by only taking into account those $\breve{x}_k^j$ which are associated with the $\hat{n}$ most significant singular values. Note that from the stochastic point of view, without regularization, the reconstructed state sequences are independent and the magnitude of their autocorrelation reveals their significance in the canonical relationship.

Therefore, given d measurements of inputs, outputs, and scheduling variables, a state sequence $\breve{x}_k$, compatible with the system, can be estimated by determining the maximum correlation between $\varphi_{\mathrm{f}}(\bar{p}_{k+d}^d)\bar{z}_{k+d}^d$ and $\varphi_{\mathrm{p}}(\bar{p}_k^d)\bar{z}_k^d$ in the CCA sense. The notion of compatibility corresponds to the fact that the state is estimated up to a linear map or state transformation $T : \mathbb{R}^{n_{\mathrm{p}}} \to \mathbb{R}^{\hat{n} \times n}$, such that $\breve{x}_k = (T \diamond p)(k) \cdot \hat{x}_k$, where $\hat{x}_k$ is the conditional mean in terms of (6.7) and $\hat{n}$ is the order of the estimated model. The state transformation T has dynamic dependence on p [42] and with $\hat{n} \geq n$ it is injective. Hence, $\breve{x}_k$ corresponds to the estimate of the state sequence of an equivalent realization of (6.2) as

$$\begin{aligned} \breve{x}_{k+1} &= (\tilde{A}_{\mathrm{e}} \diamond p)(k)\breve{x}_k + (\tilde{B}_{\mathrm{e}} \diamond p)(k)u_k + (K_{\mathrm{e}} \diamond p)(k)y_k, \\ y_k &= (C_{\mathrm{e}} \diamond p)(k)\breve{x}_k + (D_{\mathrm{e}} \diamond p)(k)u_k + e_k, \end{aligned}$$

where subscript e denotes the estimate and $T\tilde{A} = \tilde{A}_e T$, $T\tilde{B} = \tilde{B}_e$, $TK = K_e$, $C = C_e T$, and $D = D_e$.

It can be argued here that the states of the LPV system summarize all information from the past behavior needed to predict the future behavior. In that sense, the past input and output data $\bar{z}_k^d = \left[\bar{u}_k^{d\,\top}\ \bar{y}_k^{d\,\top}\right]^{\top}$ forms a non-minimal state representation of the system. This well-known fact

of the LTI system theory also holds in the LPV case, i.e., for any finite dimensional LPV-SS representation (6.1) with up to meromorphic scheduling dependencies, it is possible to give an LPV-SS realization with a state vector $\bar{z}_k^d$ which has an equivalent IO map in almost everywhere sense (i.e., all compatible trajectories are equal, except a subset of trajectories with measure zero due to possible singularity of the matrix functions). To illustrate this fact, we next show that the future output behavior is a function of past data $\bar{z}_k^d, \bar{p}_k^d$, and u_k, p_k.

Lemma 6.3.1 *Let* (6.2) *be structurally observable and $d \geq n$. Then, there exists a function $f : \mathbb{R}^{dn_{\mathrm{u}} \times dn_{\mathrm{y}} \times dn_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{y}}}$ such that for any trajectories $p \in \mathcal{P}$, $u \in (\mathbb{R}^{n_{\mathrm{u}}})^{\mathbb{Z}}$ and $e \in (\mathbb{R}^{n_{\mathrm{y}}})^{\mathbb{Z}}$*

$$\bar{y}_{k+1}^d = f(u_k, p_k, \bar{z}_k^d, \bar{p}_k^d) \tag{6.20}$$

Proof 1 *As* (6.2) *is observable along any $p \in \mathcal{P}$, using (6.8), we have*

$$\bar{y}_{k+1}^d = (\mathcal{O}_{\mathrm{p}}^d \diamond p)(k+1) \cdot x_{k-d+1} + (\mathcal{H}_{\mathrm{p}}^d \diamond p)(k+1) \cdot \bar{u}_{k+1}^d + (\mathcal{L}_{\mathrm{p}}^d \diamond p)(k+1) \cdot \bar{y}_{k+1}^d + \bar{e}_{k+1}^d, \tag{6.21}$$

Substituting (6.2a) *into (6.21) above gives*

$$\begin{aligned}\bar{y}_{k+1}^d = (\mathcal{O}_{\mathrm{p}}^d \diamond p)(k+1) \cdot \big\{ &\tilde{A}(p_{k-d}) x_{k-d} + \tilde{B}(p_{k-d}) u_{k-d} \\ &+ K(p_{k-d}) y_{k-d} \big\} + (\mathcal{H}_{\mathrm{p}}^d \diamond p)(k+1) \bar{u}_{k+1}^d + (\mathcal{L}_{\mathrm{p}}^d \diamond p)(k+1) \bar{y}_{k+1}^d + \bar{e}_{k+1}^d. \end{aligned} \tag{6.22}$$

Using (6.8) in (6.23), we obtain

$$\begin{aligned}\bar{y}_{k+1}^d = (\mathcal{O}_{\mathrm{p}}^d \diamond p)(k+1) \cdot \big\{ &\tilde{A}(p_{k-d}) \varphi_{\mathrm{p}}(\bar{p}_k^d) \bar{z}_k^d + \tilde{B}(p_{k-d}) u_{k-d} \\ &+ K(p_{k-d}) y_{k-d} + \bar{e}_k^d \big\} + (\mathcal{H}_{\mathrm{p}}^d \diamond p)(k+1) \bar{u}_{k+1}^d + (\mathcal{L}_{\mathrm{p}}^d \diamond p)(k+1) \bar{y}_{k+1}^d + \bar{e}_{k+1}^d. \end{aligned} \tag{6.23}$$

Since u_{k-d}, y_{k-d}, and p_{k-d} are entries of the vectors $\bar{z}_k^d$ and $\bar{p}_k^d$, and as the diagonal of $\mathcal{L}_{\mathrm{p}}^d$ is zero, we can replace $\bar{z}_{k+1}^d$, $\bar{p}_{k+1}^d$ with $\bar{z}_k^d$, $\bar{p}_k^d$, and add u_k, p_k to the arguments of the right hand side, giving us

$$\bar{y}_{k+1}^d = f\left(u_k, p_k, \bar{z}_k^d, \bar{p}_k^d\right). \tag{6.24}$$

Stated simply, the future inputs and outputs of the system modeled by the LPV model can be mapped to the past inputs and outputs via the states acting as the intermediate variables.

6.4 MATRIX FUNCTION ESTIMATION VIA LS-SVM

Once we have obtained an estimate of the state sequence $\{x_k\}_{k=1}^N$, we can form an extended data set $\breve{\mathcal{D}} = \{u_k, y_k, \breve{x}_k, p_k\}_{k=1}^N$ to estimate the matrix functions in (6.2). We parameterize our LPV-SS model as

$$\breve{x}_{k+1} = W_{\mathrm{x}}\varphi_{\mathrm{x}}^{\top}(p_k) + \varepsilon_k, \tag{6.25a}$$

$$y_k = W_{\mathrm{y}}\varphi_{\mathrm{y}}^{\top}(p_k) + \varsigma_k, \tag{6.25b}$$

where $\breve{x}_k$ is the estimate of x_k, ε_k and ς_k are residual errors on the states and outputs, respectively, $W_{\mathrm{x}} = [W_1 \ W_2 \ W_3] \in \mathbb{R}^{n\times 3n_{\mathrm{H}}}$ and $W_{\mathrm{y}} = [W_4 \ W_5] \in \mathbb{R}^{n_{\mathrm{y}}\times 2n_{\mathrm{H}}}$ are weighting matrices. The functions $\varphi_{\mathrm{x}}^{\top}(p_k) \in \mathbb{R}^{3n_{\mathrm{H}}\times 1}$ and $\varphi_{\mathrm{y}}^{\top}(p_k) \in \mathbb{R}^{2n_{\mathrm{H}}\times 1}$ are defined by

$$\varphi_{\mathrm{x}}^{\top}(p_k) = \left[\left(\Phi_1(p_k)\breve{x}_k\right)^{\top} \left(\Phi_2(p_k)u_k\right)^{\top} \left(\Phi_3(p_k)y_k\right)^{\top}\right]^{\top},$$
$$\varphi_{\mathrm{y}}^{\top}(p_k) = \left[\left(\Phi_4(p_k)\breve{x}_k\right)^{\top} \left(\Phi_5(p_k)u_k\right)^{\top}\right]^{\top},$$

where $\Phi_1, \Phi_4 : \mathbb{R}^{n_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{H}}\times n}$, $\Phi_2, \Phi_5 : \mathbb{R}^{n_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{H}}\times n_{\mathrm{u}}}$, and $\Phi_3 : \mathbb{R}^{n_{\mathrm{p}}} \to \mathbb{R}^{n_{\mathrm{H}}\times n_{\mathrm{y}}}$ are unknown feature maps that map the scheduling variables to a high dimensional (RKHS) $\mathcal{H}_{\bar{k}}$ defined uniquely by the kernel functions $\bar{k}_i : \mathbb{R}^{n_{\mathrm{p}}} \times \mathbb{R}^{n_{\mathrm{p}}} \to \mathbb{R}$ with $i \in 1, \ldots, 5$. From (6.25a)-(6.25b), we can gauge that $\tilde{A}(p_k) \sim W_1\Phi_1(p_k), \tilde{B}(p_k) \sim W_2\Phi_2(p_k)$, $K(p_k) \sim W_3\Phi_3(p_k)$, $C(p_k) \sim W_4\Phi_4(p_k)$, and $D(p_k) = W_5\Phi_5(p_k)$, where $\sim$ stands for an equivalent function under a state transformation. The problem in this paper, therefore, reduces to finding the dependency of $W_i\Phi_i(p_k)$ on p_k given the data $\breve{\mathcal{D}}$. To achieve this, we aim to minimize the following cost function

$$\min_{W_{\mathrm{x}}, W_{\mathrm{y}}, \varepsilon, \varsigma} \mathcal{I}(W_{\mathrm{x}}, W_{\mathrm{y}}, \varepsilon, \varsigma) = \quad \frac{1}{2}\left(\|W_{\mathrm{x}}\|_{\mathrm{F}}^2 + \|W_{\mathrm{y}}\|_{\mathrm{F}}^2 + \sum_{k=1}^{N} \varepsilon_k^{\top}\Gamma\varepsilon_k + \varsigma_k^{\top}\Psi\varsigma_k,\right) \tag{6.26}$$

over W_x, W_y, where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. Variables ε_k and ς_k are residual errors on the state sequence and the outputs, respectively, and are defined in (6.25a)-(6.25b). Matrices $\Gamma = \mathrm{diag}(\gamma_1, \cdots, \gamma_n)$ and $\Psi = \mathrm{diag}(\psi_1, \cdots, \psi_{n_{\mathrm{y}}})$ are diagonal positive weighting matrices on these residuals, and are known as regularization matrices. This problem can be solved in the dual

form by introducing the *Lagrangian* as

$$\mathcal{L}(W_{\mathrm{x}}, W_{\mathrm{y}}, \alpha, \beta, \varepsilon, \varsigma) = \mathcal{I} - \sum_{j=1}^{N} \beta_j^{\top} \left(W_{\mathrm{y}} \varphi_{\mathrm{y}}^{\top}(p_j) + \varsigma_j - y_j \right) - \sum_{j=1}^{N} \alpha_j^{\top} \left(W_{\mathrm{x}} \varphi_{\mathrm{x}}^{\top}(p_j) + \varepsilon_j - \breve{x}_{j+1} \right), \tag{6.27}$$

where $\alpha_j \in \mathbb{R}^n$, $\beta_j \in \mathbb{R}^{n_{\mathrm{y}}}$ are the Lagrange multipliers at time j. The optimal solution is obtained by solving the KKT conditions, *i.e.*, equating the partial derivatives $\frac{\partial \mathcal{L}}{\partial W_{\mathrm{x}}}, \frac{\partial \mathcal{L}}{\partial W_{\mathrm{y}}}, \frac{\partial \mathcal{L}}{\partial \alpha_j}, \frac{\partial \mathcal{L}}{\partial \beta_j}, \frac{\partial \mathcal{L}}{\partial \varepsilon_j}, \frac{\partial \mathcal{L}}{\partial \varsigma_j}$ to zero, given as

$$\frac{\partial \mathcal{L}}{\partial W_x} = 0 \Rightarrow W_x = \sum_{j=1}^{N} \alpha_j \varphi_x(p_j), \tag{6.28a}$$

$$\frac{\partial \mathcal{L}}{\partial W_y} = 0 \Rightarrow W_y = \sum_{j=1}^{N} \beta_j \varphi_y(p_j), \tag{6.28b}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_j} = 0 \Rightarrow \varepsilon_j = \breve{x}_{j+1} - W_x \varphi_x^{\top}(p_j), \tag{6.28c}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = 0 \Rightarrow \varsigma_j = y_j - W_y \varphi_y^{\top}(p_j), \tag{6.28d}$$

$$\frac{\partial \mathcal{L}}{\partial \varepsilon_j} = 0 \Rightarrow \alpha_j = \Gamma \varepsilon_j, \tag{6.28e}$$

$$\frac{\partial \mathcal{L}}{\partial \varsigma_j} = 0 \Rightarrow \beta_j = \Psi \varsigma_j. \tag{6.28f}$$

Substituting these conditions in (6.25a)-(6.25b), we can eliminate the primal decision variables and obtain the following

$$\breve{x}_{k+1} = \underbrace{\left\{ \sum_{j=1}^{N} \alpha_j \varphi_{\mathrm{x}}(p_j) \right\}}_{W_{\mathrm{x}}} \varphi_{\mathrm{x}}^{\top}(p_k) + \underbrace{\Gamma^{-1} \alpha_k}_{\varepsilon_k}, \tag{6.29a}$$

$$y_k = \underbrace{\left\{ \sum_{j=1}^{N} \beta_j \varphi_{\mathrm{y}}(p_j) \right\}}_{W_{\mathrm{y}}} \varphi_{\mathrm{y}}^{\top}(p_k) + \underbrace{\Psi^{-1} \beta_k}_{\varsigma_k}. \tag{6.29b}$$

By replacing the inner-product $\Phi_i^{\top}(p_k)\Phi_i(p_j)$ by a kernel function $\bar{k}_i(p_j, p_k)$ and defining

$$[\Omega]_{j,k} = \sum_{i=1}^{3} z_i^{\top}(j) \bar{k}_i(p_j, p_k) z_i(k), \tag{6.30a}$$

$$[\Xi]_{j,k} = \sum_{i=4}^{5} z_i^{\top}(j) \bar{k}_i(p_j, p_k) z_i(k), \tag{6.30b}$$

where

$$z_i(k) = \begin{cases} \breve{x}_k, & i = 1, 4 \\ u_k, & i = 2, 5 \\ y_k, & i = 3. \end{cases}$$

We can now write (6.29) in a compact form as follows

$$\breve{X}_{k+1} = \alpha\Omega + \Gamma^{-1}\alpha, \tag{6.31a}$$

$$Y = \beta\Xi + \Psi^{-1}\beta, \tag{6.31b}$$

where $\Omega \in \mathbb{R}^{N\times N}$ and $\Xi \in \mathbb{R}^{N\times N}$ are kernel matrices as defined above, $\alpha = [\alpha_1 \cdots \alpha_N] \in \mathbb{R}^{n\times N}$ and $\beta = [\beta_1 \cdots \beta_N] \in \mathbb{R}^{n_{\mathrm{y}}\times N}$ are the matrices containing the Lagrange multipliers, $\breve{X}_{k+1} = [\breve{x}_2 \cdots \breve{x}_{N+1}] \in \mathbb{R}^{n\times N}$ and $Y = [y_1 \cdots y_N] \in \mathbb{R}^{n_{\mathrm{y}}\times N}$ contain the estimated states and outputs for the N samples, respectively. The solution to the above equations can be obtained as:

$$\mathrm{vec}(\alpha) = \left(I_N \otimes \Gamma^{-1} + \Omega^\top \otimes I_n\right)^{-1} \mathrm{vec}(\breve{X}_{k+1}), \tag{6.32a}$$

$$\mathrm{vec}(\beta) = \left(I_N \otimes \Psi^{-1} + \Xi^\top \otimes I_{n_{\mathrm{y}}}\right)^{-1} \mathrm{vec}(Y), \tag{6.32b}$$

where matrices I_N and $I_{n_{\mathrm{y}}}$ represent identity matrices of dimensions N and n_{y}, respectively. For a given solution to (6.32a)-(6.32b), the estimate of the state-space matrices can be calculated by using the KKT conditions in (6.25a)-(6.25b) as

$$\tilde{A}_{\mathrm{e}}(\cdot) = W_1\Phi_1(\cdot) = \sum_{j=1}^{N} \alpha_j \breve{x}_j^\top \bar{k}_1(p_j, \cdot), \tag{6.33a}$$

$$\tilde{B}_{\mathrm{e}}(\cdot) = W_2\Phi_2(\cdot) = \sum_{j=1}^{N} \alpha_j u_j^\top \bar{k}_2(p_j, \cdot), \tag{6.33b}$$

$$C_{\mathrm{e}}(\cdot) = W_4\Phi_4(\cdot) = \sum_{j=1}^{N} \beta_j \breve{x}_j^\top \bar{k}_5(p_j, \cdot), \tag{6.33c}$$

$$K_{\mathrm{e}}(\cdot) = W_3\Phi_3(\cdot) = \sum_{j=1}^{N} \alpha_j y_j^\top \bar{k}_3(p_j, \cdot), \tag{6.33d}$$

$$D_{\mathrm{e}}(\cdot) = W_5\Phi_5(\cdot) = \sum_{j=1}^{N} \beta_j u_j^\top \bar{k}_5(p_j, \cdot). \tag{6.33e}$$

This gives a nonparametric estimate of the SS matrices.

Table 6.1: Monte-Carlo simulation results.

	$\hat{n}$	Mean (BFR %)	Std. (BFR %)
SNR 25dB	4	87.21	1.021
	8	89.93	0.351
SNR 20dB	4	86.41	1.002
	8	87.19	0.072
	9	86.92	1.203

6.5 TUNING OF THE HYPER PARAMETERS

Both the state trajectory estimation in terms of the KCCA problem and the estimation of the matrix functions of the SS representation require the choice of hyper-parameters for the definition of the associated kernel functions and other regularization parameters. To formulate a data-driven choice for these parameters, let τ be a vector containing the elements of $\nu_{\mathrm{f}}, \nu_{\mathrm{p}}, \gamma_1, \ldots, \gamma_n, \psi_1, \ldots, \psi_{n_{\mathrm{y}}}$ and the parameters of the kernels associated with these estimation steps. Let us denote the resulting model $\mathcal{M}(\tau)$ as the solution of the specified state and matrix function estimation problems in Sections 6.3 and 6.4 using the estimation data set $\mathcal{D}$ and a fixed choice of τ. Additionally, let $\mathcal{D}_{\mathrm{val}}$ be an independent data set generated by (6.1) and define

$$\mathrm{BFR}(\tau) = 100\% \cdot \max\left(1 - \frac{\|y_k - \hat{y}_k(\tau)\|_2}{\|y_k - \bar{y}\|_2}, 0\right), \tag{6.34}$$

as fitness score or *best fit rate* (BFR) between the actual output trajectory y of $\mathcal{D}_{\mathrm{val}}$, its mean $\bar{y}$, and the simulated[5] output $\hat{y}$ of the identified model $\mathcal{M}(\tau)$ *w.r.t.* the inputs and scheduling trajectory of $\mathcal{D}_{\mathrm{val}}$. We seek to maximize the objective function BFR over τ. This results in a nonlinear optimization problem with a quadratic cost function, which can be seen as an inference problem between the data sets $\mathcal{D}$ and $\mathcal{D}_{\mathrm{val}}$ under the given parametrization of the estimation problems in terms of τ.

[5] Alternatively, one can formulate a similar objective function *w.r.t.* the predicted output of $\mathcal{M}(\tau)$ using the predictor form (6.2).

6.6 NUMERICAL EXAMPLE

The following numerical example of a *single-input multi-output* discrete-time data-generating system is considered

$$x_{k+1} = A(p_k)x_k + B(p_k)u_k + K(p_k)e_k,$$
$$y_k = C(p_k)x_k + e_k.$$

where

$$A(p_k) = \begin{bmatrix} \mathrm{sat}(p_k) & 1 & 0 & 0 \\ \frac{1}{2} & \frac{p_k^3}{8} & \frac{4}{10} & \frac{1}{5} \\ \frac{3}{10} & 0 & \frac{p_k^2}{5} & \frac{1}{8} \\ 0 & 0 & \frac{1}{2} & \frac{1}{5} \end{bmatrix}, B(p_k) = \begin{bmatrix} \frac{p_k^4}{5} \\ 0 \\ \frac{1}{5} \\ 0 \end{bmatrix},$$

$$C(p_k) = \begin{bmatrix} \frac{p_k^2}{5} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

$$K(p_k) = \begin{bmatrix} \frac{\tanh(p_k)}{3p_k} & 0 \\ 0 & 0 \\ 0 & \sin(2\pi p_k) + \cos(2\pi p_k) \\ 0 & 1 \end{bmatrix},$$

and sat(p_k) is a saturation function with limits at ± 0.5 and unity slope. Given the measurements u_k, y_k, and p_k in $\mathcal{D}$, we want to estimate the matrix functions $A, \ldots, K$. The data-generating system with initial condition $x_0 = [0\ 0\ 0\ 0]^\top$ has been simulated with uniformly distributed input signal $u_k \sim \mathcal{U}(-1, 1)$ and normally distributed measurement noise $e_k \sim \mathcal{N}(0, \sigma_\mathrm{e}^2)$ to generate a data set $\mathcal{D} = \{u_k, y_k, p_k\}_{k=1}^N$ with $N = 1100$ and $p_k = \sin(0.3k)$, where σ_e^2 has been chosen to guarantee a 20dB *signal-to-noise ratio* (SNR). The data is divided into 800 and 300 samples for estimation $\mathcal{D}$ and validation $\mathcal{D}_\mathrm{val}$, respectively. Polynomial kernel is chosen for the state estimation step and the past and future window size of $d = 4$ is selected while for the matrix function estimation problem RBF kernels are employed. Other kernels have also been tested and the selection is made based on

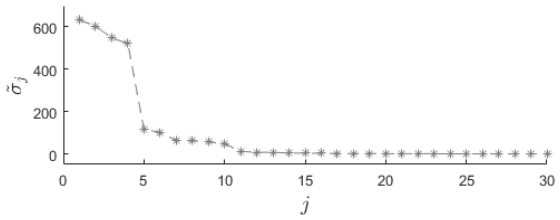

Figure 6.1: Singular values obtained by solving the SVD problem (6.19).

the maximization of the cost function (6.34). The proposed kernel CCA-based algorithm is run and the state sequence is estimated. The order of the system is selected by solving the SVD problem (6.19). A plot of the first 30 singular values $\tilde{\sigma}_j, j = 1, ..., 30$, is shown in Figure 6.1. We observe a significant gap between the first four singular values and the next four that follow them. Using the extended data set $\breve{\mathcal{D}} = \{u_k, y_k, \breve{x}_k, p_k\}_{k=1}^{N}$, we then run the LS-SVM identification algorithm of Section 6.4 based on different choices of system order to estimate the state-space matrices. An RBF kernel is chosen to find an estimate of the matrix functions. The optimal choice of the hyper-parameters τ is obtained by solving the auto-tuning problem (6.34) on a validation data set $\mathcal{D}_{\text{val}}$. This nonlinear optimization problem is solved using the `fmincon` solver in MATLAB. For independently generated data sets, the estimation is repeated 100 times in a Monte-Carlo study and the fitness score statistics are tabulated in Table 6.1. We observe that by selecting the order to be $\hat{n} = 8$, a slightly higher fitness rate is obtained compared to $\hat{n} = 4$. For $\hat{n} = 9$ and onwards, no significant improvement is observed in terms of accuracy of the simulated outputs. This is also corroborated by the singular values plot shown in Figure 6.1. Unfortunately, for $\hat{n} > 4$, the hyper-parameter tuning problem becomes numerically challenging and hence only a gridding based solution is feasible.

For $\hat{n} = 8$, the tuned values of the hyper-parameters are as follows: $\nu_{\text{f}} = 1000$, $\nu_{\text{p}} = 8500$, $\deg = 2$, $\sigma_{1,2,3,4} = \{0.7, 10.15, 0.02, 7\}$, $\gamma_{1,2,...,8} = \{2700, 600, 700, 800, 500, 1400, 1500, 800\}$ and $\psi_{1,2} = \{5500, 50\}$. Parameter $\deg$ denotes the degree of polynomial kernel selected for state estimation part and σ_i denotes the RBF kernel parameter for the second phase of the identification. Figure 6.2 shows, for a single run of simulation, the simulated outputs of the identified model (dashed red line) as compared to the original noise-free outputs of the data-generating system (solid blue line). Once tuned, the Monte-Carlo simulation statistics show a consistent performance of the

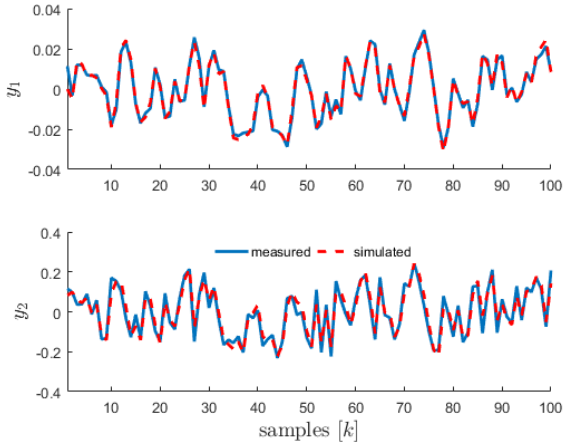

Figure 6.2: Simulated output response of the estimated LPV model in the given example computed on $\mathcal{D}_{\text{val}}$ (blue) and the noise free response of the original system (red); for the sake of clarity, only 100 of the validation data points are shown here.

proposed KCCA-LS-SVM identification algorithm. Lagrange multipliers $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are calculated based on (6.32a)-(6.32b) and the state-space matrices are estimated. Obtained average BFR values given in Table 6.1 demonstrate consistent performance of the proposed LPV model identification algorithm.

6.7 Concluding Remarks

This paper has presented a nonparametric method for identification of LPV-SS models. The proposed technique relies only on the inputs, outputs, and scheduling variables data measurement. Assuming structural observability, the states of the data-generating system are estimated up to a similarity transform by using correlation analysis between the measurements of the past and future inputs, outputs, and scheduling variables. Once estimated, an LS-SVM-based nonparametric scheme is used to identify the underlying LPV model. The proposed scheme solves a convex optimization problem and provides encouraging results on a MIMO state-space numerical example with challenging nonlinearities in the presence of noise. The main contribution of this paper lies in formulating the kernel CCA and LS-SVM solution for this identification problem by preserving the linearity structure in parameter-dependent state-space models. Since LPV-SS models are important

for LPV control synthesis purposes, we believe that this work has the potential to pave the way for efficient low-order LPV modeling and control synthesis.

CHAPTER 7

CONCLUDING REMARKS

In this dissertation, the use of machine learning and multivariate analysis to solve some key problems in the area of LPV model reduction and identification in the state-space form has been explored. The advent of kernel-based methods has introduced a plethora of tools that can solve problems in nonlinear regression, classification, and data analysis by performing linear operations in a high dimensional Hilbert space, or an RKHS. These properties of kernelized ML have been exploited in this dissertation in order to find tighter scheduling space and reduce the scheduling dependency in LPV-SS models. Comparison of results with linear PCA-based LPV model reduction has revealed superior dimensionality-reduction and data variance retention by making use of the same number of principal components, provided that the kernel hyper-parameters are tuned properly. An offline-solved optimization problem ensures that the reduced model maintains an affine dependence on the reduced scheduling variables, despite the nonlinear mapping of the kernel function. Closed-loop simulations have demonstrated that the attained reduced LPV model is viable for LPV control synthesis.

Further, identification of discrete-time LPV-SS models using kernelized ML has been explored in this dissertation. An LS-SVM-based algorithm is proposed that formulates the identification problem as a primal-dual convex optimization problem. What remains is the tuning of kernel hyper-parameters, for which, an auto-tuner has been proposed. If the order of the system is low and number of hyper-parameters is limited to a few, a grid-based search can also be performed. In case when states of the system are not directly accessible to measurements, the use of kernelized CCA is proposed. An algorithm is formulated using an LS-SVM-based regularized version of CCA that

estimates the states of the system using only the measurements of past and future inputs, outputs, and scheduling variables.

7.1 FUTRUE RESEARCH DIRECTIONS

Data-driven LPV model reduction and identification is a research area with immense potential in the practical area of nonlinear control. Accurate LPV models with low number of state variables and low dimensional scheduling variables is a key to designing low-complexity LPV controllers. While we have proposed algorithms and methods to reduce the scheduling space and to unlock the functional dependencies in LPV-SS models, there is room for further research. Possible future directions include:

(1) Method to automate the selection of kernel functions based on trends in the system-generated data;

(2) A more efficient way of auto-tuning the hyper-parameters;

(3) Improvement of identification method that maximizes not only the BFR criterion but also the frequency response matching over a wide bandwidth;

(4) Incorporation of frequency-response matching in the proposed model reduction algorithm; and

(5) Reduction of redundancy while estimating the states of the data-generating system. Due to regularization in KCCA, while estimating the states of the LPV-SS model, we end up with $2N$ solutions to the generalized eigenvalue problem (6.14) instead of N solutions. Every solution obtained has a very similar pair. This is the reason we see a better BFR by fitting an 8^{th} order model to the data generated by a 4^{th} order system, as can be seen in Table 6.1. A future direction can be to explore how to get rid of this redundancy while retaining regularization.

While solving nonlinear problems by virtue of linear dot product in a high-dimensional RKHS gives us added degrees of freedom in terms of mapping the nonlinearities in a system, the choice of kernel functions attempting to approximate those nonlinearities is non-trivial. The so-called *kernel trick* is instrumental in unlocking these nonlinear scheduling dependencies; however, the measure of its success depends heavily on choosing the right kernel function and the tuning of its associated hyper-parameters. According to Schölkopf [75], choosing a kernel function still remains an open problem. Once chosen, tuning the kernel parameters remains a nonlinear optimization problem. In this dissertation, we have mostly, either searched for the parameters over a grid or tuned it by seeking to maximize the BFR criterion. An attempt to find a closed-form solution, or a near-optimum solution to this problem in a more efficient manner would be a worthy research problem to explore. Other directions of future improvement includes incorporating in the identification or model reduction routine, a penalization on frequency response mismatch. While the BFR provides us with a fitness score based solely on the measured time-response of the original and estimated models, it does not explicitly operate in the frequency domain. This, we believe is worth exploring and should define a possible future track of this research. Lastly, as stated above, regularization in KCCA leads to redundancy in solution when solving the generalized eigenvalue problem (6.14) in order to obtain state estimates. Solving this issue will result in reducing the order of the identified model.

BIBLIOGRAPHY

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

[2] O. Sename, P. Gaspar, and J. Bokor, *Robust control and linear parameter varying approaches: application to vehicle dynamics*. Springer, 2013, vol. 437.

[3] D. J. Leith and W. E. Leithead, "Survey of gain-scheduling analysis and design," *Int. J. of Control*, vol. 73, no. 11, pp. 1001–1025, 2000.

[4] J. Mohammadpour and C. Scherer, *Control of linear parameter varying systems with applications*. Springer Science & Business Media, 2012.

[5] W. Rugh and J. Shamma, "Research on gain scheduling," *Automatica*, vol. 36, no. 10, pp. 1401–1425, 2000.

[6] J. Shamma and M. Athans, "Guaranteed properties of gain scheduled control for linear parameter-varying plants," *Automatica*, vol. 27, no. 3, pp. 559–564, 1991.

[7] K. Zhou, J. Doyle, and K. Glover, *Robust and optimal control*. Prentice hall New Jersey, 1996, vol. 40.

[8] P. Apkarian and R. Adams, "Advanced gain-scheduling techniques for uncertain systems," *IEEE Trans. Control Syst. Tech.*, vol. 6, no. 1, pp. 21–32, 1998.

[9] J. Borges, V. Verdult, M. Verhaegen, and M. Botto, "Separable least squares for projected gradient identification of composite local linear state-space models," in *Proc. of the 16*th *Intl. Symp. Math. Theory of Networks and Systems*, 2004.

[10] A. Marcos and G. Balas, "Development of linear-parameter-varying models for aircraft," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 2, pp. 218–228, 2004.

[11] V. Verdult, M. Lovera, and M. Verhaegen, "Identification of linear parameter-varying state-space models with application to helicopter rotor dynamics," *International Journal of Control*, vol. 77, no. 13, pp. 1149–1159, 2004.

[12] 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.

[13] F. Bianchi, R. Mantz, and C. Christiansen, "Gain scheduling control of variable-speed wind energy conversion systems using quasi-LPV models," *Control Engineering Practice*, vol. 13, no. 2, pp. 247–255, 2005.

[14] M. Meisami-Azad and K. Grigoriadis, "Anti-windup linear parameter-varying control of pitch actuators in wind turbines," *Wind Energy*, vol. 18, no. 2, pp. 187–200, 2015.

[15] J. van Wingerden, *Control of wind turbines with'Smart'rotors: proof of concept & LPV subspace identification*. TU Delft, Delft University of Technology, 2008.

[16] S. Baslamisli, I. Köse, and G. Anlaş, "Gain-scheduled integrated active steering and differential control for vehicle handling improvement," *Vehicle System Dynamics*, vol. 47, no. 1, pp. 99–119, 2009.

[17] F. Wijnheijmer, G. Naus, W. Post, M. Steinbuch, and P. Teerhuis, "Modelling and LPV control of an electro-hydraulic servo system," in *Proc. of the 2006 IEEE Conf. on Computer Aided Control System Design*, 2006, pp. 3116–3121.

[18] L. Fiorentini, A. Serrani, M. Bolender, and D. Doman, "Nonlinear robust adaptive control of flexible air-breathing hypersonic vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 2, pp. 402–417, 2009.

[19] M. Tanelli, D. Ardagna, and M. Lovera, "LPV model identification for power management of web service systems," in *Proc. of the 2008 IEEE Intl. Conf. on Control Applications*, 2008, pp. 1171–1176.

[20] B. Paijmans, W. Symens, H. Brussel, and J. Swevers, "Identification of interpolating affine LPV models for mechatronic systems with one varying parameter," *European Journal of Control*, vol. 14, no. 1, pp. 16–29, 2008.

[21] J. Barker and G. Balas, "Comparing linear parameter-varying gain-scheduled control techniques for active flutter suppression," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 948–955, 2000.

[22] A. Bachnas, R. Tóth, J. Ludlage, and A. Mesbah, "A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study," *Journal of Process Control*, vol. 24, no. 4, pp. 272–285, 2014.

[23] H. Nishimura and K. Funaki, "Motion control of three-link brachiation robot by using final-state control with error learning," *IEEE/ASME Trans. on Mechatronics*, vol. 3, no. 2, pp. 120–128, 1998.

[24] G. Millerioux, L. Rosier, G. Bloch, and J. Daafouz, "Bounded state reconstruction error for LPV systems with estimated parameters," *IEEE Trans. on Automatic Control*, vol. 49, no. 8, pp. 1385–1389, 2004.

[25] M. Wassink, M. van de Wal, C. Scherer, and O. Bosgra, "LPV control for a wafer stage: beyond the theoretical solution," *Control Engineering Practice*, vol. 13, no. 2, pp. 231–245, 2005.

[26] P. C. Pellanda, P. Apkarian, and H. D. Tuan, "Missile autopilot design via a multi-channel LFT/LPV control method," *Int. J. of Robust and Nonlinear Control*, vol. 12, no. 1, pp. 1–20, 2002.

[27] L. Giarré, D. Bauso, P. Falugi, and B. Bamieh, "LPV model identification for gain scheduling control: An application to rotating stall and surge control problem," *Control Engineering Practice*, vol. 14, no. 4, pp. 351–361, 2006.

[28] K. Takahashi and S. Massaquoi, "Neuroengineering model of human limb control gain-scheduled feedback control approach," in *Proc. of the 46$^{\text{th}}$ IEEE Conf. on Decision and Control*, 2007, pp. 5826–5832.

[29] J. Gao and H. Budman, "Design of robust gain-scheduled PI controllers for nonlinear processes," *J. of Process Control*, vol. 15, no. 7, pp. 807–817, 2005.

[30] 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.

[31] P. Gaspar, Z. Szabo, and J. Bokor, "The design of an integrated control system in heavy vehicles based on an LPV method," in *Proc. of the 44$^{\text{th}}$ IEEE Conf. Decision and Control, and 2005 European Control Conf.*, 2005, pp. 6722–6727.

[32] M. Siraj, R. Tóth, and S. Weiland, "Joint order and dependency reduction for LPV state-space models," in *Proc. of the 51$^{\text{st}}$ IEEE Conf. Decision and Control*, 2012, pp. 6291–6296.

[33] 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.

[34] L. Zhang and P. Shi, "Model reduction for switched LPV systems with average dwell time," *IEEE Trans. on Automatic Control*, vol. 53, no. 10, pp. 2443–2448, 2008.

[35] B. Bamieh and L. Giarre, "Identification of linear parameter varying models," *Int. J. Robust Nonlinear Control*, vol. 12, no. 9, pp. 841–853, 2002.

[36] R. Tóth, V. Laurain, W. Zheng, and K. Poolla, "Model structure learning: A support vector machine approach for LPV linear-regression models," in *Proc. of the 50*$^{\text{th}}$ *IEEE Conf. Decision and Control, and European Control Conf.*, Dec 2011, pp. 3192–3197.

[37] D. Piga, P. Cox, R. Tóth, and V. Laurain, "LPV system identification under noise corrupted scheduling and output signal observations," *Automatica*, vol. 53, pp. 329–338, 2015.

[38] R. Tóth, V. Laurain, M. Gilson, and H. Garnier, "Instrumental variable scheme for closed-loop LPV model identification," *Automatica*, vol. 48, no. 9, pp. 2314–2320, 2012.

[39] V. Laurain, R. Tóth, M. Gilson, and H. Garnier, "Direct identification of continuous-time linear parameter-varying input/output models," *IET Control Theory & Applications*, vol. 5, no. 7, pp. 878–888, 2011.

[40] V. Laurain, M. Gilson, R. Tóth, and H. Garnier, "Refined instrumental variable methods for identification of LPV box–jenkins models," *Automatica*, vol. 46, no. 6, pp. 959–967, 2010.

[41] F. Abbasi, J. Mohammadpour, R. Tóth, and N. Meskin, "A support vector machine-based method for LPV-ARX identification with noisy scheduling parameters," in *Proc. of the 13*$^{\text{th}}$ *European Control Conf.*, 2014, pp. 370–375.

[42] 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.

[43] A. van der Voort, "LPV control based on a pick-and-place unit," Ph.D. dissertation, Ph. D. thesis, TU Delft, Delft University of Technology, 2002.

[44] J. H. Yung, "Gain scheduling for geometrically nonlinear flexible space structures," Ph.D. dissertation, Ph.D. thesis, Massachusetts Institute of Technology, 2001.

[45] M. Steinbuch, M. van de Molengraft, and A. V. der Voort, "Experimental modelling and LPV control of a motion system," in *Proc. of the American Control Conf.*, 2003, pp. 1374–1379.

[46] M. Lovera and G. Mercére, "Identification for gain-scheduling: a balanced subspace approach," in *Proc. of the American Control Conf.*, 2007.

[47] V. Laurain, R. Tóth, W. Zheng, and M. Gilson, "Nonparametric identification of LPV models under general noise conditions: an LS-SVM based approach," in *Proc. of the 16$^{\text{th}}$ IFAC Symposium on Syst. Identification*, 2012, pp. 1761–1766.

[48] V. Verdult and M. Verhaegen, "Identification of multivariable LPV state space systems by local gradient search," in *2001 European Control Conf.*, 2001, pp. 3675–3680.

[49] ——, "Subspace identification of multivariable linear parameter-varying systems," *Automatica*, vol. 38, no. 5, pp. 805–814, 2002.

[50] ——, "Kernel methods for subspace identification of multivariable LPV and bilinear systems," *Automatica*, vol. 41, no. 9, pp. 1557–1565, 2005.

[51] F. Felici, J. van Wingerden, and M. Verhaegen, "Subspace identification of MIMO LPV systems using a periodic scheduling sequence," *Automatica*, vol. 43, no. 10, pp. 1684–1697, 2007.

[52] J. van Wingerden and M. Verhaegen, "Subspace identification of MIMO LPV systems: The PBSID approach," in *Proc. of the 47$^{\text{th}}$ IEEE Conf. Decision and Control*, 2008, pp. 4516–4521.

[53] ——, "Subspace identification of bilinear and LPV systems for open-and closed-loop data," *Automatica*, vol. 45, no. 2, pp. 372–381, 2009.

[54] N. Lachhab, H. Abbas, and H. Werner, "A neural-network based technique for modelling and LPV control of an arm-driven inverted pendulum," in *Proc. of the 47$^{\text{th}}$ IEEE Conf. on Decision and Control*, 2008, pp. 3860–3865.

[55] P. Gil, J. Henriques, A. Dourado, and H. Duarte-Ramos, "On state-space neural networks for systems identification: Stability and complexity," *IEEE CIS-RAM*, vol. 94, 2006.

[56] R. Tóth, *Modeling and identification of linear parameter-varying systems*. Heidelberg: Springer, 2010.

[57] P. dos Santos, T. Azevedo-Perdicoulis, J. Ramos, S. Deshpande, D. Rivera, and J. de Carvalho, "LPV system identification using a separable least squares support vector machines approach," in *Proc. of the 53*rd *IEEE Conf. Decision and Control*, 2014, pp. 2548–2554.

[58] B. Schölkopf, A. Smola, and K. Müller, "Kernel principal component analysis," in *Advances in kernel methods-support vector learning*. Boston, MA: MIT Press, 1999, pp. 327–352.

[59] M. Siraj and R. Tóth, "On the problem of model reduction of LPV systems," in *Proc. of the 31*st *Benelux Meeting*, 2012.

[60] M. Meisami-Azad, J. Mohammadpour, K. M. Grigoriadis, M. P. Harold, and M. A. Franchek, "PCA-based linear parameter varying control of scr aftertreatment systems," in *Proc. of the American Control Conference*, 2011, pp. 1543–1548.

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

[62] I. Jolliffe, *Principal Component Analysis*, 2nd ed. NY: Springer, 2002.

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

[64] 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.

[65] J. Lee, C. Yoo, S. Choi, P. Vanrolleghem, and I. Lee, "Nonlinear process monitoring using kernel principal component analysis," *Chemical Engineering Science*, vol. 59, pp. 223–234, 2004.

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

[67] S. Hashemi, H. Abbas, and H. Werner, "LPV modeling and control of a 2-DOF robotic manipulator using PCA-based parameter set mapping," in *Proc. of the 48$^{\text{th}}$ IEEE Conf. Decision and Control, and 28$^{\text{th}}$ Chinese Control Conf.*, Shanghai, China, 2009, pp. 7418–7423.

[68] S. Mika, B. Schölkopf, A. Smola, K. Mller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," in *Proc. of the Neural Information Processing Systems*, 1999, pp. 536–542.

[69] 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.

[70] 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.

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

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

[73] 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.

[74] S. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin, "Parameter set-mapping using kernel-based PCA for linear parameter-varying systems," in *Proc. of the 13$^{\text{th}}$ European Control Conf.*, 2014, pp. 2744–2749.

[75] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proc. of the Int. Conf. Artificial Neural Networks*. Springer, 1997, pp. 583–588.

[76] L. Lee and K. Poolla, "Identification of linear parameter-varying systems using nonlinear programming," *ASME J. Dynamic Syst. Measurement Cont.*, vol. 121, no. 1, pp. 71–78, 1999.

[77] V. Verdult, "Nonlinear system identification: a state-space approach," Ph.D. dissertation, University of Twente, The Netherlands, 2002.

[78] F. Casella and M. Lovera, "LPV/LFT modelling and identification: overview, synergies and a case study," in *Proc. of the IEEE Int. Conf. Computer Aided Control Syst.*, 2008, pp. 852–857.

[79] M. Tanelli, D. Ardagna, and M. Lovera, "Identification of LPV state space models for autonomic web service systems," *IEEE Trans. Control Syst. Tech.*, vol. 19, no. 1, pp. 93–103, 2011.

[80] A. Golabi, N. Meskin, R. Tóth, and J. Mohammadpour, "A bayesian approach for estimation of LPV linear-regression models," in *Proc. of the 53*$^{\text{rd}}$ *IEEE Conf. Decision and Control*, 2014, pp. 2555–2560.

[81] P. Christofides and N. El-Farra, *Control of nonlinear and hybrid process systems: Designs for uncertainty, constraints and time-delays*. Springer, 2005, vol. 324.

[82] R. Bartels and G. Stewart, "Solution of the matrix equation $AX + XB = C$," *Comm. ACM*, vol. 15, pp. 820–826, 1972.

[83] R. Tóth, P. V. den Hof, J. Ludlage, and P. Heuberger, "Identification of nonlinear process models in an LPV framework," in *Proc. of the 9th Int. Symp. Dynamics and Control of Process Syst., Leuven, Belgium*, 2010.

[84] B. Roffel and B. Betlem, *Process Dynamics and Control: Modeling for Control and Prediction*. Wiley, 2007.

[85] R. Tóth, P. Heuberger, and P. V. den Hof, "Prediction-error identification of LPV systems: Present and beyond," in *Control of Linear Parameter Varying Systems with Applications*. Springer, 2012, pp. 27–58.

[86] V. Laurain, R. Tóth, D. Piga, and W. Zheng, "An instrumental least squares support vector machine for nonlinear system identification," *Automatica*, vol. 54, pp. 340–347, 2015.

[87] S. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin, "A kernel-based approach to MIMO LPV state-space identification and application to a nonlinear process system," in *Proc. of the 1st IFAC Workshop on Linear Parameter Varying Systems*, Grenoble, France, 2015, pp. 85–90.

[88] L. Ljung, *System identification: theory for the user*. NJ: Prentice Hall, 1987.

[89] S. Rizvi, J. Mohammadpour, R. Tóth, and N. Meskin, "An IV-SVM-based approach for identification of state-space LPV models under generic noise conditions," in *Proc. of the 54th IEEE Conf. Decision and Control*, Osaka, Japan, 2015, pp. 7380–7385.

[90] J. Suykens, T. Gestel, B. D. Moor, and J. Vandewalle, "Basic methods of least squares support vector machines," in *Least Squares Support Vector Machines*. World Scientific, 2002.

[91] V. Verdult, J. Suykens, J. Boets, I. Goethals, and B. D. Moor, "Least squares support vector machines for kernel CCA in nonlinear state-space identification," in *Proc. of the 16th Intl. Symp. Math. Theory of Networks and Syst*, Leuven, Belgium, 2004.

[92] T. D. Bie and B. D. Moor, "On the regularization of canonical correlation analysis," *Int. Symp. Independent Component Analysis and Blind Signal Separation*, pp. 785–790, 2003.

[93] F. Bach and M. Jordan, "Kernel independent component analysis," *The Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2003.

[94] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Prentice Hall, 1979.

[95] F. Cucker and S. Smale, "On the mathematical foundations of learning," *Bulletin of the American Mathematical Society*, vol. 39, no. 1, pp. 1–49, 2001.