Polytopic Decomposition of Linear Parameter-Varying Models by Tensor-Product Model Transformation

Ph.D. Dissertation

Zoltán Petres

Supervisors:
Péter Baranyi Ph.D.
Péter Korondi Ph.D.

Budapest, 2006.

Alulírott Petres Zoltán kijelentem, hogy ezt a doktori értekezést magam készítettem és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

A disszertáció bírálatai és a védésről készült jegyzőkönyv megtekinthető a Budapest Műszaki és Gazdasságtudományi Egyetem Villamosmérnöki és Informatikai Karának Dékáni Hivatalában.

Budapest, 2006. november 6.

Petres Zoltán

Acknowledgments

I would like to express my deepest appreciation and gratitude to my supervisors *Péter Baranyi* and *Péter Korondi*. Without their guidance, unconditional support, scientific and ethical encouragement throughout my studies and research, this dissertation would have been impossible.

I wish to thank *Prof. Hideki Hashimoto* his invitation to his laboratory at The University of Tokyo, and his support for doing my research for 18 months. Grateful acknowledgments are due to my friends, *Anna Szemereki* and *Barna Reskó* for their generosity to help and manage all my Hungarian responsibilities while I was doing my research in Japan, and to *Ádam Csapó* for uncountably reviewing my manuscripts.

Moreover, I cannot be thankful enough to *my parents*, *family* and *friends* for self-devotion, encouragement, and support in the past years. I appreciate the support of Ministry of Education, Culture, Sport, Science and Technology of Japan that covered my financial needs during my stay, and also the financial support of Computer and Automation Research Institute of the Hungarian Academy of Sciences.

Contents

List of Figures

Chapter 1

Introduction

1.1 Preliminaries and scientific background of the research work

The research work leading to the results presented in this dissertation is based on the significant paradigm changes in control theory, mathematics and system theory, appearing almost simultaneously in the last decade.

In the last decade, the representation of identification models in system theory has changed significantly. The origins of the paradigm shift can be linked with the famous speech given by D. HILBERT in Paris in 1900. HILBERT listed 23 conjectures, hypotheses concerning unsolved problems which he believed would provide the biggest challenge in the 20th century. According to the 13th conjecture there exist continuous multi-variable functions which cannot be decomposed as the finite superposition of continuous functions of less variables [40, 41, 43, 50]. In 1957 ARNOLD disproved this hypothesis [3], moreover, in the same year, KOLMOGOROV [52] formulated a general representation theorem, along with a constructive proof, where the functions in the decomposition were one dimensional. This proof justified the existence of *universal approximators*. KOLMOGOROV's representation theorem was further improved by several authors (SPRECHER [73] and LORENTZ [61]). Based on these results, starting from the 1980s, it has been proved that universal approximators exist among the approximation tools of biologically inspired neural networks and genetic algorithms, as well as fuzzy logic [17, 25, 27, 45, 55, 66, 82, 88]. In this manner, these approximators have appeared in the identification models of system theory, and turned out to be effective tools even for systems that can hardly be described in an analytical way.

One of the most fruitful developments in the world of linear algebra and linear algebra-based signal processing is the concept of the Singular Value Decomposition (SVD) of matrices. The history of matrix decomposition goes back to the 1850s. During the last 150 years several mathematicians—Eugenio Beltrami (1835–1899), Camille Jordan (1838–1921), James Joseph Sylvester (1814–1897), Erhard Schmidt (1876–1959), and Hermann Weyl (1885–1955), to name a few of the more important ones—were responsible for establishing the existence of the singular value decomposition and developing its theory [75]. Thanks to the pioneering efforts of Gene Golub, there exist efficient, stable algorithms to compute the singular value decomposition [39]. More recently, SVD started to play an important role in several scientific

fields [28, 63, 83]. Its popularity also grew in parallel with the more and more efficient numerical methods. Due to the development of personal computers it became possible to handle larger-scale, multi-dimensional problems, and there is a greater demand for the higher-order generalization of SVD for tensors. Higher Order SVD (HOSVD) is used efficiently in independent component analysis (ICA) [58], as well as in the dimensionality reduction for higher-order factor analysis-type problems—thus reducing the computational complexity [57]—to name a few examples. The HOSVD concept was first published as a whole multi-dimensional SVD concept in 2000 [60], and the Workshop on Tensor Decompositions and Applications held in Luminy, Marseille, France, August 29–September 2, 2005 was the first event where the key topic was HOSVD. Its very unique power in linear algebra comes from the fact that it can decompose a given N-dimensional tensor into a full orthonormal system in a special ordering of singular values, expressing the rank properties of the tensor in order of L_2-norm. In effect, the HOSVD is capable of extracting the very clear and unique structure underlying the given tensor. The Tensor Product (TP) model transformation is a further extension to continuous N-variable functions. It is capable of extracting the fully orthonormal and singular value ordered structure of the given function. Note that this structure cannot be analytically achieved, since there is no general analytic solution for the HOSVD. The TP model transformation was also extended to linear parameter-varying (LPV) models in 2003. It generates the HOSVD of LPV models. To be specific: it generates the parameter-varying combination of Linear Time-Invariant (LTI) models that represents the given LPV model in such a way that: i) the number of the LTI components are minimized; ii) the weighting functions are univariate functions of the parameter vector; iii) the weighting functions are in an orthonormal system for each parameter; iv) the LTI systems are also in orthogonal position; v) the LTI systems and the weighting functions are ordered by the singular values.

In conclusion, the TP model transformation finds the clear well defined and unique structure of the given LPV model. This cannot be achieved via analytical derivations. Thus the result of the TP model transformation was termed as the HOSVD-based canonical form of polytopic or LPV models in 2006 [10, 11].

The appearance of *Lyapunov-based stability criteria* made a significant improvement in the control theory of nonlinear systems. This change of the viewpoint was invoked by the reformulation of these criteria in the form of *linear matrix inequalities*, in the early 1990s. Herewith, the stability questions of control theory were given in a new representation, and the feasibility of Lyapunov-based criteria was reinterpreted as a convex optimization problem, as well as, extended to an extensive model class. The pioneers GAHINET, BOKOR, CHILAI, BOYD, and APKARIAN were responsible for establishing this new concept [1, 2, 19, 29, 32, 33, 36, 49, 64, 71]. The geometrical meaning and the methodology of this new representation were developed in the research group of Prof. József BOKOR. Soon, it was also proved that this new representation could be used for the formulation of different control performances—in the form of linear matrix inequalities—beyond the stability issues together with the optimization problem. Ever since, the number of papers about linear matrix inequalities guaranteeing different stability and control properties are increasing drastically. BOYD's paper [20] states that it is true of a wide class of control problems that if the problem is formulated in the form of linear matrix inequalities, then the problem is practically solved.

In parallel with the above research and thanks to the significant increase in the computational performance of computers, efficient numerical mathematical methods and algorithms were developed for solving *convex optimization* problems—thus linear matrix inequalities. The breakthrough in the use of convex optimization in practical applications dates back to the introduction of *interior point methods*. These methods were developed in series of papers [48], and have real importance in connection with linear matrix inequality problems in the work of Yurii NESTEROV and Arkadii NEMIROVSKI [65]. Today, these methods are used in "everyday" engineering work, and it turns out to be equally efficient in cases when the closed formulation is unknown. In consequence, the formulation of analytical problems has gained a new meaning.

It is well-known that a considerable part of the problems in modern control theory necessitate the solution of *Riccati-equations*. However, the general analytical (closed formulation) solution of multiple Riccati-equations is unknown. In turn—with the usage of numerical methods of convex optimization—we consider solved those problems today that require the resolution of a large number of convex algebraic Ricatti-equations, in spite of the fact that a result of the obtained solution is not a closed (in classical sense) analytical equation.

> In conclusion, the most advantageous property of the new, convex optimization based representation in control theory is that it is possible to easily combine different controller design conditions and goals in the form of numerically manageable linear matrix inequalities [20]. This makes it possible to solve numerous (complex) control theory problems with remarkable efficiency.

This is especially true of Lyapunov-based analysis and synthesis, but also of optimal Linear Quadratic (LQ) control, H_∞ control [30, 38, 76], as well as minimal variance control. The linear matrix inequality-based design also appeared in other areas such as estimation, identification, optimal design, structural design, and matrix-sizing problems. The following enumeration lists further problems that can be managed and solved in a representation using linear matrix inequalities: robust stability of linear time-invariant systems with uncertainty (μ-analysis) [67, 74, 92], quadratic stability [21, 44], Lyapunov-based stability of parameter-dependent systems [35], the guarantee of constraints on linear time-invariant system inputs, state variables, and outputs, or other goals [20], multi-model and multi-objective state-feedback control [4, 16, 20, 26, 51], robust pole-placement, optimal LQ control [20], robust H_∞ control [34, 46], multi-goal H_∞ synthesis [26, 51, 62], control of stochastic systems [20], weighted interpolation problems [20].

1.2 General description of the scientific problem

According to the above, a model, given either by a closed analytical formulation, or as the output of an identification using soft-computing techniques such as fuzzy logic, neural networks, or genetic algorithms, can be transformed to HOSVD-based canonical form by TP model transformation. The only requirement is that the model must be able to discretize over a grid. From the HOSVD-based canonical form, different convex polytopic models can be generated by simple numerical transformations depending on the further application of the

model. The control goals also can be given as a convex optimization problem in the form of linear matrix inequalities, and can be solved using modern numerical convex optimization algorithms. The convex polytopic models generated by the TP model transformation can be immediately applied to the linear matrix inequalities, and this makes it possible to solve a wide class of problems irrespective of the existence or non-existence of an analytical solution in a closed form. The TP model transformation based controller design methodology offers a uniform, tractable, straightforward numerical controller design framework. In many cases, the analytical derivation, affine decomposition, or the convex optimization can be really troublesome, time consuming or even impossible even with state-of-the-art analytical tools, while in many applications it can be proved that numerical methods can easily overcome on these difficulties. The intention of the presented dissertation is to analyze the applicability and feasibility of TP model transformation in control theory problems.

The linear matrix inequality based controller design was well-researched in the last decade, therefore its validity and applicability analysis is not set as a goal in this dissertation.

1.3 Goal of the dissertation

As a result of the paradigm shift outlined in Section 1.1 several efficient system theory tools were developed using different representations. Even with the joint usage of the tools, the adaptation of different representations—in the point of view of design methods and the mathematical tools, especially the analytical transformations—are difficult, and in many cases this problem is not solved. During my research work, my main goal was to investigate and analyze the applicability and feasibility, and to extend the usability of the Tensor Product model transformation based controller design methodology, a tractable and uniform controller design methodology for a wide class of complex control theory problems.

Therefore, my comprehensive goals in details are as follows

- Investigate whether the Tensor Product model transformation for complex, benchmark and real-world type dynamic systems yields models that are interpretable and can be formulated for convex optimization problems composed in the form of linear matrix inequalities. An academic benchmark problem is chosen for the theoretical evaluation, and valid comparison with other published methodologies. Besides, industry related experimental analysis is also an important aspect of the investigation.

- Show that the finite element HOSVD-based canonical form and the convex finite element TP models of the chosen models exist and the TP model transformation generates the minimal number of linear time-invariant systems.

- Since a crucial point of the Tensor Product model transformation is the computational complexity explosion for higher dimensional problems, my goal is to investigate the applicability of the Tensor Product model transformation and to suggest algorithms and methods for decreasing the computational complexity load of the transformation for higher dimensional problems.

1.4 Structure of the dissertation

The dissertation is divided into four parts. Part I discusses the preliminaries of the dissertation. In Part II and Part III the new results are studied. Part IV concludes the dissertation.

In Chapter 2 the Higher Order Singular Value Decomposition (HOSVD) of tensors are introduced. The decomposition is a generalization of the Singular Value Decomposition introduced for matrices. The Higher Order Singular Value Decomposition is a key mathematical tool of Tensor Product model transformation. Chapter 3 defines the HOSVD-based canonical form of linear parameter-varying dynamic models that is a parameter-varying weighted combination of parameter independent (constant) system models (linear time-invariant systems). Section 3.4 defines the Tensor Product model transformation that is capable of transforming the linear parameter-varying dynamic models to HOSVD-based canonical form. In Chapter 4 different convex hulls of linear parameter-varying models are shown. These convex hulls make possible the immediate use of linear matrix inequality-based controller design methods and the derivation of multi-objective controller satisfying the given control performance requirements. For the sake of completeness, a short introduction to Parallel Distributed Compensation based control design framework is given in Chapter 5. The framework joints the convex Tensor Product models and the linear matrix inequality-based controller design.

Part II discusses the new theoretical achievements. In Chapter 6 the computational complexity load problem of Tensor-Product model transformation is considered. For problems with high degree of nonlinearity the computation complexity explodes. I propose two solutions to overcome this problem. One is based on the decrease of the discretization grid density, whilst the other's key idea is to avoid the computation of constant elements in the LPV model.

Part III is devoted for applications. I investigated whether the Tensor Product model transformation for a complex, academic benchmark type dynamic system yields models that are interpretable and can be formulated for convex optimization problems composed in the form of linear matrix inequalities in Chapter 7. The TORA system that is considered as a fourth-order benchmark problem is chosen for the theoretical evaluation, and valid comparison with other published methodologies.

Another application is an industry related experimental analysis that is always a key issue of a controller design methodology. In Chapter 8 I investigate the applicability of the Tensor Product model transformation on a real-world, industrial experimental setup. In industrial crane systems the load swinging is a key problem. The Single Pendulum Gantry (SPG) system is a laboratory model of a simplified crane system. I design and evaluate the position and swing angle controller derived by Tensor Product model transformation controller design, a tractable and uniform numerical controller design methodology for a wide class of complex control theory problems.

Part IV summarizes the new results and achievements of the dissertation.

Part I

Preliminaries

Chapter 2

Higher Order Singular Value Decomposition of Tensors

The theoretical utility of matrix decompositions has long been appreciated. More recently, they have become the mainstay of numerical linear algebra, where they serve as computational platforms from which a variety of problems can be solved.

One of the most fruitful developments in the world of linear algebra and linear algebra-based signal processing is the concept of the Singular Value Decomposition (SVD) of matrices. The history of matrix decomposition goes back to 1850s. During the last 150 years several mathematicians—the most important ones are Eugenio Beltrami (1835–1899), Camille Jordan (1838–1921), James Joseph Sylvester (1814–1897), Erhard Schmidt (1876–1959), and Hermann Weyl (1885–1955)—were responsible for establishing the existence of the singular value decomposition and developing its theory [75]. Thanks to the pioneering efforts of Gene Golub, there exist efficient, stable algorithms to compute the singular value decomposition [39]. More recently, SVD started to play an important role in several scientific fields [28, 63, 83]. Its popularity also grew in parallel with the more and more efficient numerical methods. Due to the development of personal computers it became possible to handle larger-scale, multi-dimensional problems, and there is a greater demand for the higher order generalization of the SVD for tensors. Higher Order SVD (HOSVD) is used efficiently in independent component analysis (ICA) [58], as well as in the dimensionality reduction for higher-order factor analysis-type problems—thus reducing the computational complexity [57]—to name a few examples. The HOSVD concept was first published as a whole multi-dimensional SVD concept in 2000 [60], and the Workshop on Tensor Decompositions and Applications held in Luminy, Marseille, France, August 29–September 2, 2005 was the first event where the key topic was HOSVD.

This section briefly introduces the basic operators of tensor algebra used throughout this dissertation, and the main concept of Higher Order Singular Value Decomposition (HOSVD) tensors. Several research results have been published in this area, however this section is mainly based on Lathauwer's work [60].

2.1 Basic concept of tensor algebra

The starting point in the derivation of a multilinear singular value decomposition (SVD) for tensors, as multi-dimensional matrices, is to consider an appropriate generalization of the link between the column (row) vectors and the left (right) singular vectors of a matrix. In order to formalize this idea, we define the matrix representations of the tensor in which all the column (row, ...) vectors are stacked one after the other in the following way:

Definition 2.1 (n-mode matrix of tensor $\mathcal{A}$)**.** Assume an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$. The matrix $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times (I_{n+1} I_{n+2} \cdots I_N I_1 I_2 \cdots I_{n-1})}$ contains the element $a_{i_1, i_2, \ldots, i_N}$ at the position with row number i_n and column number equal to:

$$\begin{aligned}(i_{n+1}-1) I_{n+2} I_{n+3} \cdots I_N I_1 I_2 \cdots I_{N-1} + (i_{n+2}-1) I_{n+3} I_{n+4} \cdots I_N I_1 I_2 \cdots I_{n-1} + \cdots + \\ + \quad (i_N - 1) I_1 I_2 \cdots I_{n-1} + (i_1 - 1) I_2 I_3 \cdots I_{n-1} + (i_2 - 1) I_3 I_4 \cdots I_{n-1} + \cdots + i_{n-1}.\end{aligned}$$

Remark 2.1. The ordering of the column vectors can be arbitrarily determined. The only important thing is that in all cases the same ordering and reordering must be used systematically later on. In general, the rth column of n-mode matrix $\mathbf{A}_{(n)}$ is equivalent to the $I_1, I_2, \ldots, I_{n-1}, I_{n+1}, \ldots, I_N$-th vector of dimension n, where

$$r = \text{ordering}(i_1, i_2, \ldots, i_{n-1}, i_{n+1}, \ldots, i_N).$$

Figure 2.1 shows an example for the n-mode matrix of a 3rd-order tensor.

Rank property of Nth-order tensor There are major differences between matrices and higher-order tensors when rank properties are concerned. These differences directly affect the generalization of singular value decomposition based on matrices to Nth-order matrices. There are many theories in the literature concerning the rank properties of tensors. This section represents only that definition which is used in this dissertation henceforth. This definition generalizes the notion of column and row rank. If we refer in general to the column, row, ... vectors of an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ as its "n-mode vectors," defined as the I_n-dimensional vectors obtained from $\mathcal{A}$ by varying the index i_n and keeping the other indices fixed, then we have the following definition.

Definition 2.2 (n-mode rank of tensor $\mathcal{A}$)**.** The n-rank of $\mathcal{A}$, denoted by $R_n = \text{rank}_n(\mathcal{A})$, is the dimension of the vector space spanned by the n-mode vectors.

From computational point of view it directly follows that $\text{rank}_n(\mathcal{A}) = \text{rank}\left(\mathbf{A}_{(n)}\right)$, which is used for the calculation of n-rank.

Scalar product, orthogonality, norm of higher-order tensors In the HOSVD definition of Section 2.2, the structure constraint of diagonality imposed on the matrix of singular values will, in the second-order case, be replaced by a number of geometrical conditions. This requires a generalization of the well-known definitions of scalar product, orthogonality, and Frobenius-norm to tensors of arbitrary order. These generalizations can be defined in a straightforward way as follows.

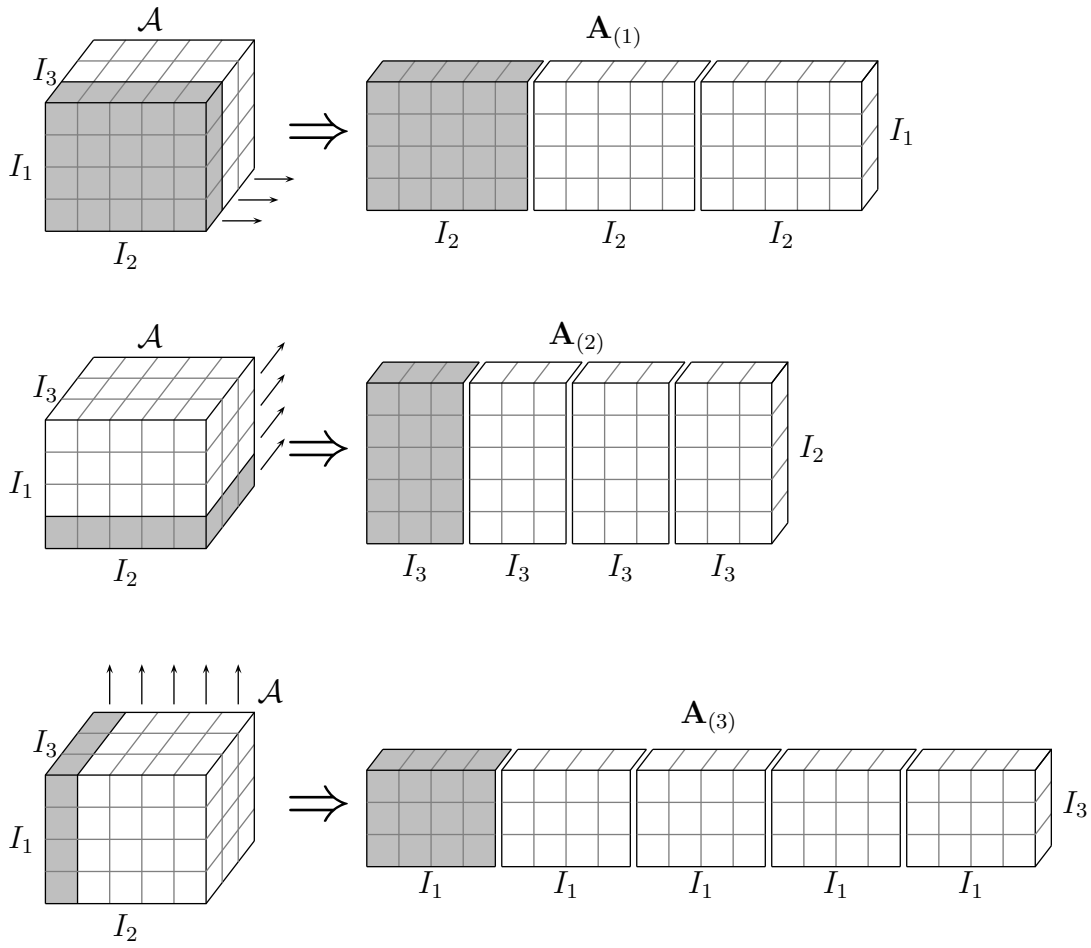

Figure 2.1: Illustration of 3-mode matrices of a 3rd-order tensor $\mathcal{A}$

Definition 2.3 (Scalar product)**.** The scalar product $\langle \mathcal{A}, \mathcal{B} \rangle$ of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as

$$\langle \mathcal{A}, \mathcal{B} \rangle \stackrel{\text{def}}{=} \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} b_{i_1 i_2 \ldots i_N} a_{i_1 i_2 \ldots i_N}.$$

Definition 2.4 (Orthogonality)**.** Arrays of which the scalar product equals 0 are orthogonal.

Definition 2.5 (Frobenius-norm)**.** The Frobenius-norm of a tensor $\mathcal{A}$ is given by

$$\|\mathcal{A}\| \stackrel{\text{def}}{=} \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}.$$

Multiplication of a higher-order tensor by a matrix. The HOSVD of a higher-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, in a way analogous to the HOSVD of matrices, will be defined by looking for orthogonal coordinate transformations of $\mathbb{R}^{I_1}, \mathbb{R}^{I_2}, \ldots, \mathbb{R}^{I_N}$ that induce a particular representation of the higher-order tensor. In this section we establish a notation for the multiplication of a higher-order tensor by a matrix. This will allow us to express the effect of basis transformations.

Let us first have a look at the matrix product $\mathbf{G} = \mathbf{U} \cdot \mathbf{F} \cdot \mathbf{V}^T$, involving matrices $\mathbf{F} \in \mathbb{R}^{I_1 \times I_2}$, $\mathbf{U} \in \mathbb{R}^{J_1 \times I_1}$, $\mathbf{V} \in \mathbb{R}^{J_2 \times I_2}$, and $\mathbf{G} \in \mathbb{R}^{J_1 \times J_2}$. To avoid working with "generalized transposes" in the multilinear case, we observe that the relationship between $\mathbf{U}$ and $\mathbf{F}$ and the relationship between $\mathbf{V}$ (not $\mathbf{V}^T$) and $\mathbf{F}$ are in fact completely similar: in the same way as $\mathbf{U}$ makes linear combinations of the rows of $\mathbf{F}$, $\mathbf{V}$ makes linear combinations of the columns of $\mathbf{F}$; in the same way as the columns of $\mathbf{F}$ are multiplied by $\mathbf{U}$, the rows of $\mathbf{F}$ are multiplied by $\mathbf{V}$; in the same way as the columns of $\mathbf{U}$ are associated with the column space of $\mathbf{G}$, the columns of $\mathbf{V}$ are associated with the row space of $\mathbf{G}$. This typical relationship will be denoted by means of the $\times_n$-symbol: $\mathbf{G} = \mathbf{F} \times_1 \mathbf{U} \times_2 \mathbf{V}$. In general, we have the following definition.

Definition 2.6 (n-mode product of a tensor by a matrix)**.** The n-mode product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J_n \times I_n}$, denoted by $\mathcal{A} \times_n \mathbf{U}$, is an $(I_1 \times I_2 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N)$-tensor of which the entries are given by

$$(\mathcal{A} \times_n \mathbf{U})_{i_1 i_2 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} \stackrel{\text{def}}{=} \sum_{i_n} a_{i_1 i_2 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} u_{j_n i_n}.$$

The multiple n-mode product of a tensor, such as $\mathcal{A} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \cdots \times_N \mathbf{U}_N$ can be shortly denoted as

$$\mathcal{A} \bigotimes_{n=1}^{N} \mathbf{U}_n.$$

From a computational point of view, the n-mode product of a tensor by a matrix $\mathcal{A} = \mathcal{B} \times_n \mathbf{U}$ can be defined as $\mathbf{A}_{(n)} = \mathbf{U}\mathbf{B}_{(n)}$.

The n-mode product satisfies the following properties.

Property 2.1. Given the tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and the matrices $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}, \mathbf{G} \in \mathbb{R}^{J_m \times I_m}, (n \neq m)$, one has

$$(\mathcal{A} \times_n \mathbf{F}) \times_m \mathbf{G} = (\mathcal{A} \times_m \mathbf{G}) \times_n \mathbf{F} = \mathcal{A} \times_n \mathbf{F} \times_m \mathbf{G}.$$

Property 2.2. Given the tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and the matrices $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}, \mathbf{G} \in \mathbb{R}^{K_n \times J_n}$, one has

$$(\mathcal{A} \times_n \mathbf{F}) \times_n \mathbf{G} = \mathcal{A} \times_n (\mathbf{G} \cdot \mathbf{F}).$$

2.2 Higher Order Singular Value Decomposition (HOSVD)

In this section an SVD model is proposed for Nth-order tensors. To facilitate the concept, we first recall the matrix SVD as follows.

Theorem 2.1 (Matrix SVD)**.** *Every real $(I_1 \times I_2)$-matrix $\mathbf{F}$ can be written as the product*

$$\mathbf{F} = \mathbf{U}_{(1)} \cdot \mathbf{S} \cdot \mathbf{V}_{(2)}^T = \mathbf{S} \times_1 \mathbf{U}_{(1)} \times_2 \mathbf{V}_{(2)} = \mathbf{S} \times_1 \mathbf{U}_{(1)} \times_2 \mathbf{U}_{(2)} = \mathbf{S} \bigotimes_{n=1}^{2} \mathbf{U}_{(n)}, \qquad (2.1)$$

in which

1. $\mathbf{U}_{(1)} = \left(\mathbf{u}_1^{(1)}\mathbf{u}_2^{(1)}\cdots\mathbf{u}_{I_1}^{(1)}\right)$ *is a unitary* $(I_1 \times I_1)$*-matrix,*

2. $\mathbf{U}_{(2)} = \left(\mathbf{u}_1^{(2)}\mathbf{u}_2^{(2)}\cdots\mathbf{u}_{I_1}^{(2)}\right)$ *is a unitary* $(I_2 \times I_2)$*-matrix,*

3. $\mathbf{S}$ *is an* $(I_1 \times I_2)$*-matrix with the properties of*

   (a) *pseudodiagonality:*

   $$\mathbf{S} = \operatorname{diag}\left(\sigma_1, \sigma_2, \ldots, \sigma_{\min(I_1,I_2)}\right), \tag{2.2}$$

   (b) *ordering:*

   $$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(I_1,I_2)} \geq 0. \tag{2.3}$$

The σ_i *are singular values of* $\mathbf{F}$ *and the vectors* $\mathbf{u}_i^{(1)}$ *and* $\mathbf{u}_i^{(2)}$ *are, respectively, an* i*th left and an* i*th right singular vector.*

Remark 2.2. The number of non-zero singular values σ_i equals to the rank of matrix $\mathbf{F}$.

Now we state the generalization of matrix SVD.

Theorem 2.2 (Higher Order SVD, HOSVD)**.** *Every real* $(I_1 \times I_2 \times \cdots \times I_N)$*-tensor* $\mathcal{A}$ *can be written as the product*

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{U}_{(1)} \times_2 \mathbf{U}_{(2)} \times_3 \cdots \times_N \mathbf{U}_{(N)} = \mathcal{S} \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_{(n)}, \tag{2.4}$$

in which

1. $\mathbf{U}_{(n)} = \left(\mathbf{u}_1^{(n)}\mathbf{u}_2^{(n)}\cdots\mathbf{u}_{I_n}^{(n)}\right)$*,* $n = 1 \ldots N$ *is a unitary* $(I_n \times I_n)$*-matrix,*

2. $\mathcal{S}$ *is a real* $(I_1 \times I_2 \times \cdots \times I_N)$*-tensor of which the subtensors* $\mathcal{S}_{i_n=\alpha}$ *obtained by fixing the* n*th index to* α*, have the properties of*

   (a) *all-orthogonality: two subtensors* $\mathcal{S}_{i_n=\alpha}$ *and* $\mathcal{S}_{i_n=\beta}$ *are orthogonal for all possible values of* n, α *and* β *subject to* $\alpha \neq \beta$*:*

   $$\langle \mathcal{S}_{i_n=\alpha}, \mathcal{S}_{i_n=\beta} \rangle = 0, \text{ when } \alpha \neq \beta, \tag{2.5}$$

   (b) *ordering:*

   $$\|\mathcal{S}_{i_n=1}\| \geq \|\mathcal{S}_{i_n=2}\| \geq \cdots \geq \|\mathcal{S}_{i_n=I_n}\| \geq 0, \tag{2.6}$$

   for all possible values of n*.*

The Frobenius-norms $\|\mathcal{S}_{i_n=i}\|$*, symbolized by* $\sigma_i^{(n)}$*, are* n*-mode singular values of* $\mathcal{A}$ *and the vector* $\mathbf{u}_i^{(n)}$ *is an* i*th* n*-mode singular vector. The decomposition is visualized for third-order tensors in Figure 2.2.*

Note that the HOSVD uniquely determines tensor $\mathcal{S}$, but the determination of matrices $\mathbf{U}_{(n)}$ may not be unique if there are equivalent singular values at least in one dimension.

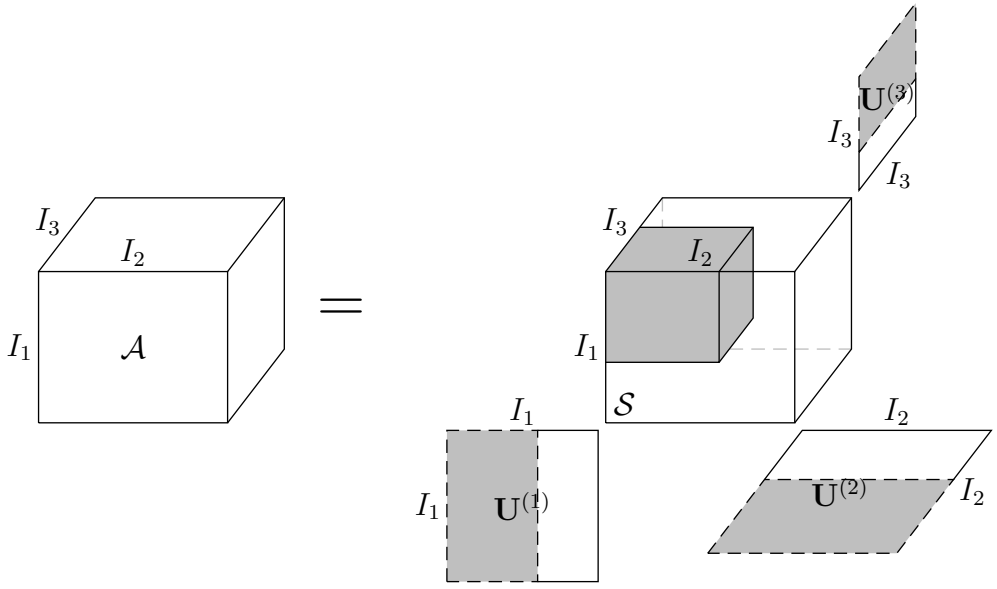

Figure 2.2: Visualization of the HOSVD for a third-order tensor

2.3 Approximation trade-off by HOSVD

In the following section we analyze the approximation property of Higher Order SVD.

Definition 2.7 (Exact Minimized form of Higher Order SVD (Exact MHOSVD))**.** The HOSVD is computed by executing SVD on each dimension of $\mathcal{S}$. If we discard the zero singular values and the related singular vectors $\mathbf{u}_{r_n+1\ldots I_n,n}$, where $r_n = \text{rank}_n(\mathcal{S})$, during the SVD computation of each dimension then we obtain the Exact Minimized form of HOSVD as

$$\mathcal{S} = \mathcal{D} \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n, \tag{2.7}$$

which has all the properties as in Theorem 2.2 except the size of $\mathbf{U}_n$ and $\mathcal{D}$. Here $\mathbf{U}_n$ has the size of $I_n \times r_n$ and $\mathcal{D}$ has the size of $r_1 \times \cdots \times r_N$.

Definition 2.8 (Non-Exact Minimized form of Higher Order SVD (Non-Exact MHOSVD))**.** If we discard non-zero singular values (not only the zero ones) and the corresponding singular vectors , then the decomposition only results an approximation of tensor $\mathcal{S}$ with the following property.

Property 2.3. Assume the HOSVD of tensor $\mathcal{A}$ is given according to Theorem 2.2, and the n-mode rank of $\mathcal{A}$ is R_n $(1 \leq n \leq N)$. Let us define $\hat{\mathcal{A}}$ by changing the corresponding elements of singular values $\sigma^{(n)}_{I'_n+1}, \sigma^{(n)}_{I'_n+2}, \ldots \sigma^{(n)}_{R_n}$ of tensor $\mathcal{S}$ to zero, for a given $I'_n < R_n$. In this case

$$\gamma = \|\mathcal{A} - \hat{\mathcal{A}}\|^2 \leq \sum_{i_1=I'_{1+1}}^{R_1} \left(\sigma^{(1)}_{i_1}\right)^2 + \sum_{i_2=I'_{2+1}}^{R_2} \left(\sigma^{(2)}_{i_2}\right)^2 + \cdots + \sum_{i_N=I'_{N+1}}^{R_N} \left(\sigma^{(N)}_{i_N}\right)^2. \tag{2.8}$$

This property is the Nth-order generalization of the connection between the singular value decomposition of a matrix and its best, lower ranked matrix approximation (in the sense of least square).

In the higher order case, the discarding of singular values in each dimension results in a lower rank along each dimension, while, contrarily to the singular value decomposition for matrices, the resulting tensor $\hat{\mathcal{A}}$, having a lowered rank along each of its dimensions, is not the best approximation to the given tensor $\mathcal{A}$. Irrespective of this fact, the descending order of the singular values indicates the increasing error during their discard. The upper limit of this error can be given by (2.8). The best approximation for a given rank reduction can be achieved by the proper modification of the elements of tensor $\hat{\mathcal{A}}$, for further details see [56, 59].

2.4 Computation of HOSVD

The n-mode singular matrix $\mathbf{U}_{(n)}$ (and the n-mode singular values) can directly be found as the left singular matrix (and the singular values) of an n-mode matrix of $\mathcal{A}$ (any matrix of which the columns are given by the n-mode vectors can be resorted to, as the column ordering is of no importance). Hence computing the HOSVD of an Nth-order tensor leads to the computation of N different matrix SVDs of matrices with size $(I_n \times I_1 I_2 \dots I_{n-1} I_{n+1} \dots I_N)$, $(1 \leq n \leq N)$.

$$\mathbf{A}_{(n)} = \mathbf{U}_{(n)} \Theta_{(n)} \left(\mathbf{V}_{(n)}\right)^T.$$

Afterwards, the core tensor $\mathcal{S}$ can be computed by bringing the matrices of singular vectors to the left side of (2.4)

$$\mathcal{S} = \mathcal{A} \times_1 \mathbf{U}_{(1)}^T \times_2 \mathbf{U}_{(2)}^T \times_3 \cdots \times_N \mathbf{U}_{(N)}^T = \mathbf{A} \bigotimes_{n=1}^{N} \mathbf{U}_{(n)}^T.$$

Chapter 3

Higher Order Singular Value Decomposition of Linear Parameter-Varying state-space models

3.1 Linear Parameter-Varying state-space model

Consider the following linear parameter-varying (LPV) state-space model:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}(\mathbf{p}(t))\mathbf{x}(t) + \mathbf{B}(\mathbf{p}(t))\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}(\mathbf{p}(t))\mathbf{x}(t) + \mathbf{D}(\mathbf{p}(t))\mathbf{u}(t), \end{aligned} \tag{3.1}$$

with input $\mathbf{u}(t) \in \mathbb{R}^k$, output $\mathbf{y}(t) \in \mathbb{R}^l$ and state vector $\mathbf{x}(t) \in \mathbb{R}^m$. The system matrix

$$\mathbf{S}(\mathbf{p}(t)) = \begin{pmatrix} \mathbf{A}(\mathbf{p}(t)) & \mathbf{B}(\mathbf{p}(t)) \\ \mathbf{C}(\mathbf{p}(t)) & \mathbf{D}(\mathbf{p}(t)) \end{pmatrix} \in \mathbb{R}^{(m+k)\times(m+l)} \tag{3.2}$$

is a parameter-varying object, where $\mathbf{p}(t) \in \Omega$ is a time varying N-dimensional parameter vector, and is an element of the closed hypercube $\Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_N, b_N] \subset \mathbb{R}^N$. Parameter $\mathbf{p}(t)$ can also include some elements of $\mathbf{x}(t)$. Therefore this type of model is considered to belong to the class of non-linear models. In the followings $O = m + k$ and $I = m + l$.

Definition 3.1 (Finite element TP model)**.** $\mathbf{S}(\mathbf{p}(t))$ of (3.2) is given for any parameter $\mathbf{p}(t)$ as the parameter-varying combination of linear time-invariant (LTI) system matrices $\mathbf{S}_{i_1 i_2 \ldots i_N} \in \mathbb{R}^{O\times I}$ also called *vertex systems*

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{S} \bigotimes_{n=1}^{N} \mathbf{w}_n(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}, \tag{3.3}$$

where row vector $\mathbf{w}_n(p_n) \in \mathbb{R}^{I_n}$ $n = 1, \ldots, N$ contains one bounded variable and continuous weighting functions $w_{n,i_n}(p_n)$, $(i_n = 1 \ldots I_n)$. The weighting function $w_{n,i_n}(p_n(t))$ is the i_nth weighting function defined on the nth dimension of Ω, and $p_n(t)$ is the nth element of vector $\mathbf{p}(t)$. $I_n < \infty$ denotes the number of the weighting functions used in the nth dimension

of Ω. Note that the dimensions of Ω are respectively assigned to the elements of the parameter vector $\mathbf{p}(t)$. The $(N+2)$-dimensional coefficient tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N \times O \times I}$ is constructed from LTI vertex systems $\mathbf{S}_{i_1 i_2 \ldots i_N} \in \mathbb{R}^{O \times I}$. For further details we refer to [5–7, 12].

3.2 HOSVD-based canonical form of LPV models

Consider an LPV model (3.1), which can be given in the finite element TP model form (3.3). Namely, the matrix function $\mathbf{S}(\mathbf{p}(t))$ can be given as:

$$\mathbf{S}(\mathbf{p}(t)) = \mathcal{S} \bigotimes_{n=1}^{N} \mathbf{w}_n(p_n(t)),$$

where $\mathbf{p}(t) \in \Omega$. We assume that, functions $w_{n,i_n}(p_n(t))$, $n = 1 \ldots N$ and $i_n = 1 \ldots I_n$ are linearly independent (in the means of $\mathbf{L}_2[a_n, b_n]$) over the intervals $[a_n, b_n]$, respectively. The linearly independent functions $w_{n,i_n}(p_n)(t)$ are determinable by the linear combinations of orthonormal functions (for instance by Gram–Schmidt-type orthogonalization method): thus, one can determine such a system of orthonormal functions for all n as $\varphi_{n,i_n}(p_n(t))$, $1 \leq i_n \leq I_n$, respectively defined over the intervals $[a_n, b_n]$, where all $\varphi_{n,k_n}(p_n(t))$, $1 \leq k_n \leq I_n$ are the linear combination of w_{n,i_n}, where i_n is not larger than k_n for all n. The functions w_{n,i_n} can respectively be determined in the same way by functions φ_{n,k_n}. Thus, one can see that if the form (3.3) of (3.1) exists then one can determine

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{C} \bigotimes_{n=1}^{N} \varphi_n(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}, \tag{3.4}$$

where tensor $\mathcal{C}$ has constant elements, and vectors $\varphi_n(p_n(t))$ consist of elements $\varphi_{n,k_n}(p_n(t))$.

Corollary 3.1. *We can assume, without the loss of generality, that functions w_{n,i_n} in the tensor-product representation of $\mathbf{S}(\mathbf{p}(t))$ are given in an orthonormal system:*

$$\forall n : \int_{a_n}^{b_n} w_{n,i_n}(p_n(t)) w_{n,j_n}(p_n(t)) \mathrm{d}x = \delta_{i_n,j_n}, \quad 1 \leq i_n, j_n \leq I_n,$$

where δ_{i_n,j_n} is a Kronecker-function ($\delta_{i_n,j_n} = 1$, if $i_n = j_n$ and $\delta_{i_n,j_n} = 0$, if $i_n \neq j_n$)-type

Having the resulting matrices $\mathbf{U}_n$ by executing exact MHOSVD, as given in Definition 2.7, on the first N-dimension of tensor $\mathcal{S}$ we can determine the following weighting functions:

$$\bar{\mathbf{w}}_n(p_n(t)) = \mathbf{w}_n(p_n(t)) \mathbf{U}_n$$

Then, based on (3.3) and (2.7) we arrive at:

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{D} \bigotimes_{n=1}^{N} \bar{\mathbf{w}}_n(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}, \tag{3.5}$$

Since the number of elements of tensor $\mathcal{D}$ in dimension n ($n = 1 \ldots N$) is equivalent to $r_n = \mathrm{rank}_n(S)$, we have r_n number of functions on dimension n in (3.5). Observe that

matrices $\mathbf{U}_n$ are orthonormal matrices and $w_{n,i_n}(p_n(t))$, $1 \leq i_n \leq I_n$ are also in orthonormal position for all $n = 1 \dots N$. Therefore, functions $\bar{w}_{n,1}(p_n(t)), \dots, \bar{w}_{n,r_n}(p_n(t))$ are also in orthonormal position for all n, since

$$\int_{a_n}^{b_n} \bar{\mathbf{w}}_n(p_n(t))\bar{\mathbf{w}}_n^T(p_n(t))\mathrm{d}p_n(t) = \mathbf{U}_n^T \left(\int_{a_n}^{b_n} \mathbf{w}_n(p_n(t))\mathbf{w}_n^T(p_n(t))\mathrm{d}p_n(t) \right) \mathbf{U}_n =$$

$$= \mathbf{U}_n^T \mathbf{I}_{r_n} \mathbf{U}_n = \mathbf{U}_n^T \mathbf{U}_n = \mathbf{I}_{r_n},$$

where matrix $\mathbf{I}_{r_n}$ denotes identity matrix with the size of $r_n \times r_n$. Based on the above and Corollary 3.1 we obtain the following theorem:

Theorem 3.1. *Consider (3.1) which have the form of (3.3). Then we can determine:*

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{D} \underset{n=1}{\overset{N}{\otimes}} \mathbf{w}_n(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}, \tag{3.6}$$

by generating the Exact Minimized form of HOSVD on the first N-dimension of $\mathcal{S}$. The resulting tensor $\mathcal{D}$ has the size of $r_1 \times \dots \times r_N \times O \times I$. The weighting functions have the following properties:

1. *The r_n number of weighting functions $w_{n,in}(p_n(t))$ contained in vector $\mathbf{w}_n(p_n(t))$ form an orthonormal system. The weighting function $\mathbf{w}_{i,n}(p_n(t))$ is an ith singular function on dimension $n = 1 \dots N$.*

   Tensor $\mathcal{D}$ has the following properties:

2. *Tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N \times O \times I}$ whose subtensors $\mathcal{D}_{i_n=\alpha}$ have the properties of*

   (a) *all-orthogonality: select one element of dimension $N+1$ and $N+2$ of tensor $\mathcal{D}$. The selected N-dimensional subtensor $\mathcal{D}'$ has all-orthogonality as: two subtensors $\mathcal{D}'_{i_n=\alpha}$ and $\mathcal{D}'_{i_n=\beta}$ are orthogonal for all possible values of n, α and $\beta : \left\langle \mathcal{D}'_{i_n=\alpha}, \mathcal{D}'_{i_n=\beta} \right\rangle = 0$ when $\alpha \neq \beta$,*

   (b) *ordering: $\left\| \mathcal{D}'_{i_n=1} \right\| \geq \left\| \mathcal{D}'_{i_n=2} \right\| \geq \dots \geq \left\| \mathcal{D}'_{i_n=I_n} \right\| \geq 0$ for all possible values of $n = 1 \dots N$.*

3. *The Frobenius-norms $\left\| \mathcal{D}'_{i_n=i} \right\|$, symbolized by $\sigma_i^{(n)}$, are n-mode singular values of $\mathcal{S}$.*

4. *$\mathcal{D}$ is termed core tensor consisting the LTI systems.*

Definition 3.2 (HOSVD-based canonical form of finite element TP model)**.** We call (3.6) the *HOSVD-based canonical form* of (3.3).

Remark 3.1. If there are equal singular values on any dimension when the Exact Minimized form of HOSVD is generated, then the canonical form is not unique. Obviously, if the non-zero singular values are different then the signum of the corresponding elements of the singular matrices may systematically vary. This means that the signum of the weighting functions may vary in the same way.

3.3 Numerical reconstruction of the HOSVD-based canonical form

For the sake of completeness, we give necessary concepts and theorems for the numerical reconstruction of the HOSVD-based canonical form without the detailed deduction in this section.

Let us decompose the intervals $[a_n, b_n]$, $n = 1 \dots N$ into M_n number of disjunct subintervals $\triangle_{n,m_n}$, $1 \leq m_n \leq M_n$ so as:

$$\xi_{n,0} = a_n < \xi_{n,1} < \ldots < \xi_{n,M_n} = b_n, \quad \triangle_{n,m_n} = [\xi_{n,m_n}, \xi_{n,m_n-1}),$$

Utilizing the above intervals, we can discretize the function $\mathbf{S}(\mathbf{p})$ at given points over the intervals, for example by letting

$$x_{n,m_n} \in \triangle_{n,m_n}, \quad 1 \leq m_n \leq M_n \tag{3.7}$$

Let us define a hyper-rectangular grid by elements x_{n,m_n}. We define all grid points by N element vector $\mathbf{g}$, whose elements are $\mathbf{g}_{m_1,\dots,m_N} = \begin{pmatrix} x_{1,m_1} & \cdots & x_{N,m_N} \end{pmatrix}$. Then we discretize function $\mathbf{S}(\mathbf{p}(t))$ for all grid points as:

$$\mathbf{B}_{m_1,\dots,m_N} = \mathbf{S}(\mathbf{g}_{m_1,\dots,m_N}).$$

Then we construct $N+2$-dimensional tensor $\mathcal{B}$ from matrices $\mathbf{B}_{m_1,\dots,m_N}$. Obviously the size of this tensor is $M_1 \times \cdots \times M_N \times O \times I$.

Furthermore, discretize vector valued functions $\mathbf{w}_n(p_n(t))$ over the discretization points x_{n,m_n} and construct matrices $\mathbf{W}_n$ from the discretized values as

$$\mathbf{W}_n = \begin{pmatrix} w_{n,1}(x_{n,1}) & w_{n,2}(x_{n,1}) & \cdots & w_{n,r_n}(x_{n,1}) \\ w_{n,1}(x_{n,2}) & w_{n,2}(x_{n,2}) & \cdots & w_{n,r_n}(x_{n,2}) \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1}(x_{n,M_n}) & w_{n,2}(x_{n,M_n}) & \cdots & w_{n,r_n}(x_{n,M_n}) \end{pmatrix} \tag{3.8}$$

Then tensor $\mathcal{B}$ can simply be given by (3.6) and (3.8) as

$$\mathcal{B} = \mathcal{D} \overset{N}{\underset{n=1}{\otimes}} \mathbf{W}_n \tag{3.9}$$

Theorem 3.2. *Assume that the decomposition (3.6) of function $\mathbf{S}(\mathbf{p}(t))$ exists, and we define M_n number of equidistant subinterval on each dimension $n = 1 \dots N$ over $[a_n, b_n]$, respectively for the discretization of $\mathbf{S}(\mathbf{p}(t))$. If we define sufficiently large M_n for all n then*

$$\operatorname{rank}_n(\mathcal{B}) = r_n.$$

Theorem 3.3. *Assume that functions $w_{n,i_n}(p_n)$, $1 \leq i_n \leq r_n$, are continuously derivable (at the endpoints of the interval we mean left and right hand side derivatives). If*

$$M_n > \bar{M}_n = 2r_n(b_n - a_n)^2 K_1 K_1'$$

then

$$\mathrm{rank}_n(\mathcal{B}) = r_n,$$

where

$$\begin{aligned} \max_{1\le i_n\le r_n}\ \max_{a_n\le p_n\le b_n} |w_{n,i_n}(p_n(t))| &= K_1, \\ \max_{1\le i_n\le r_n}\ \max_{a_n\le p_n\le b_n} |w'_{n,i_n}(p_n(t))| &= K'_1. \end{aligned}$$

Theorem 3.4. *Let the Exact Minimized form of HOSVD of discretized tensor $\mathcal{B}$ be:*

$$\mathcal{B} = \hat{\mathcal{D}} \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n, \tag{3.10}$$

where the size of $\mathbf{U}_n$ is $M_n \times r_n$ and the size of $\hat{\mathcal{D}}$ is $r_1 \times \cdots \times r_N \times O \times I$. We assume that the Exact Minimized from of HOSVD of $\mathcal{B}$ results in different non-zero singular values on each dimension. Then

$$\left(\sqrt{\triangle p_n}\mathbf{W}_n - \mathbf{U}_n\right)^T \left(\sqrt{\triangle p_n}\mathbf{W}_n - \mathbf{U}_n\right) \to 0 \tag{3.11}$$

while $M_n \to \infty$. Here $\triangle p_n = (b_n - a_n)/M_n$. Column-vectors $\mathbf{W}_{n,i_n}$ are determined to the extent of their sign.

Corollary 3.2. *Denote $\hat{w}_{n,i_n}(x_{n,m_n}) = u_{n,i_n,m_n}$, that is the m_nth element of the i_nth column of $\mathbf{U}_n$, $x_{n,m_n} \in \triangle_{n,m_n}$. Theorem 3.4 immediately leads to (functions w_{n,i_n} are determined to the extent of their signum, see Remark 3.1):*

$$\int_{a_n}^{b_n} (w_{n,i_n}(p_n) - \hat{w}_{n,i_n}(p_n(t)))^2 \mathrm{d}p_n \to 0,$$

$$M_n \to \infty, \quad i_n = 1 \ldots r_n$$

this means that the functions $\hat{w}_{n,i_n}(p_n(t))$ converge to $w_{n,i_n}(p_n)$ in the sense of $\mathbf{L}_2[a_n, b_n]$. As a matter of fact if the smoothness of (3.6) satisfies further conditions then we may derive even stronger theorems than Theorem 3.4.

In the same way:

Theorem 3.5. *Let the Exact Minimized form of HOSVD of discretized tensor $\mathcal{B}$ is:*

$$\mathcal{B} = \hat{\mathcal{D}} \overset{N}{\underset{n=1}{\otimes}} \mathbf{U}_n, \tag{3.12}$$

where the size of $\mathbf{U}_n$ is $M_n \times r_n$ and the size of $\hat{\mathcal{D}}$ is $r_1 \times \cdots \times r_N \times O \times I$. We assume that the Exact Minimized form of HOSVD of $\mathcal{B}$ results in different non-zero singular values on each dimension. Then

$$\hat{\mathcal{D}} \to \mathcal{D},$$

with $\forall n : M_n \to \infty$. Here $\mathcal{D}$ is form the HOSVD based canonical form. Namely, when we increase M_n to infinity the resulting $\hat{\mathcal{D}}$ converges to $\mathcal{D}$.

3.4 Computation of the HOSVD-based canonical form by TP model transformation

The TP model transformation was first introduced in [5, 12]. The goal of the TP model transformation is to transform a given state-space model (3.1) into TP model form (3.3). The transformation is a numerical method and has three key steps. The first step is the discretization of the given $\mathbf{S}(\mathbf{p}(t))$ over a huge number of points $\mathbf{p}(t) \in \Omega$. The discretized points are defined by a dense hyper-rectangular grid. The second step extracts the LTI vertex systems from the discretized systems. This step is specialized to find the HOSVD-based canonical form. The third step defines the continuous weighting functions to the LTI vertex systems.

Method 3.1 (TP model transformation)**.**
Step 1) Discretization

- Define the transformation space Ω as: $\mathbf{p}(t) \in \Omega : [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_N, b_N]$.
- Define a hyper-rectangular grid by arbitrarily located grid-lines in the transformation space Ω. For simplicity, define as equidistantly located grid-lines: $g_{n,m_n} = a_n + \frac{b_n - a_n}{M_n - 1}(m_n - 1)$, $m_n = 1 \dots M_n$. The numbers of the grid lines in the dimensions are M_n.
- Discretize the given function $\mathbf{S}(\mathbf{p}(t))$ over the grid-points:

$$\mathbf{S}^D_{m_1,m_2,\dots,m_N} = \mathbf{S}(\mathbf{p}_{m_1,m_2,\dots,m_N}) \in \mathbb{R}^{O \times I},$$

  where $\mathbf{p}_{m_1,m_2,\dots,m_N} = \begin{pmatrix} g_{1,m_1} & \dots & g_{N,m_N} \end{pmatrix}$. Superscript "D" means "discretized".
- Store the discretized matrices $\mathbf{S}^D_{m_1,m_2,\dots,m_N}$ into the tensor $\mathcal{S}^D \in \mathbb{R}^{M_1 \times M_2 \times \cdots \times M_N \times O \times I}$

Step 2) Extracting the LTI vertex systems
This step uses the previously defined MHOSVD (see Section 2.2) on the first N dimensions of tensor $\mathcal{S}^D$. While performing the MHOSVD we discard all zero or small singular values σ_k and their corresponding singular vectors in all $n = 1 \dots N$ dimensions. As a result we have

$$\mathcal{S}^D \underset{\gamma}{\approx} \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n,$$

where the error γ is bounded by Properties 2.3 as:

$$\gamma = \left(\left\| \mathcal{S}^D - \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n \right\|_{L_2} \right)^2 \leq \sum_k \sigma_k^2. \tag{3.13}$$

The resulting tensor $\mathcal{S}$, with the size of $(I_1 \times I_2 \times \cdots \times I_N \times O \times I)$, where $\forall n : I_n \leq r_n \leq M_n$, contains the LTI vertex systems, and is immediately substitutable into (3.3).
Step 3) Constructing continuous weighting function system

- The i_nth column vector $\mathbf{u}_{n,i_n=1\ldots I_n}$ of matrix $\mathbf{U}_n \in \mathbb{R}^{M_n \times I_n}$ determines one discretized weighting function $w_{n,i_n}(p_n(t))$ of variable $p_n(t)$. The values u_{n,m_n,i_n} of column i_n define the values of the weighting function $w_{n,i_n}(p_n(t))$ over the grid-lines $p_n(t) = g_{n,m_n}$:
$$w_{n,i_n}(g_{n,m_n}) = u_{n,m_n,i_n}.$$

- The weighting functions can be determined over any points by the help of the given $\mathbf{S}(\mathbf{p}(t))$. In order to determine the weighting functions in vector $\mathbf{w}_d(p_d)$, let p_k be fixed to the grid-lines as:
$$p_k = g_{k,1} \quad k = 1 \ldots N, \quad k \neq d.$$
Then for p_d:
$$\mathbf{w}_d(p_d) = (\mathbf{S}(\mathbf{p}(t)))_{(3)} \left(\left(\mathcal{S} \underset{k}{\otimes} \mathbf{u}_{k,1} \right)_{(n)} \right)^{+},$$
where vector $\mathbf{p}(t)$ consists of elements p_k and p_d as $\mathbf{p}(t) = \begin{pmatrix} g_{1,1} & \ldots & p_d & \ldots & g_{N,1} \end{pmatrix}$, and superscript "$^+$" denotes pseudo inverse and $\mathbf{u}_{k,1}$ is the first row vector of $\mathbf{U}_k$. The third-mode matrix $(\mathbf{S}(\mathbf{p}(t)))_{(3)}$ of matrix $\mathbf{S}(\mathbf{p}(t))$ is understood such that matrix $\mathbf{S}(\mathbf{p}(t))$ is considered as a three-dimensional tensor, where the length of the third-dimension is one. This practically means that the matrix $\mathbf{S}(\mathbf{p}(t))$ is stored in a one-row vector by placing the rows of $\mathbf{S}(\mathbf{p}(t))$ next to each other, respectively.

The error of the transformation can be estimated by the equation (3.13) given above. However the precise error between the resulting tensor-product by the transformation and the original model can be computed numerically over the dense hyper grid defined in space Ω.

From a practical point of view, it is worth mentioning that the computational time of the TP model transformation greatly depends on the computational time of the HOSVD.

Chapter 4

Different convex hulls of LPV models by TP model transformation

There are many ways to define the vertex systems. The applications of TP models specifies special requirements for the weighting functions. Such an example is the Parallel Distributed Compensation (PDC) based controller design framework, will be introduced in the next Section, that can be applied in cases when the vertex systems resulted by the TP model transformation give a convex hull of $\mathbf{S}(\mathbf{p}(t))$. In some cases even more conditions have to be satisfied. In PDC based controller design at least the convexity is necessary in theory, but as several application experiences show it is not enough in most cases. We can find examples when the tight convex hull of the system matrix $\mathbf{S}(\mathbf{p}(t))$ gave satisfactory results, while for observer design the convex hull extended with other properties made the problem solvable.

The type of the convex hull determined by the vertex system can be defined by the weighting functions. Therefore, we introduce weighting functions satisfying different convex hull types, and methods how they can be generated in Step 2 of the TP model transformation (Method 3.1).

4.1 Different types of weighting functions

First, let us define some possible types of the vector $\mathbf{w}(q)$ containing the weighting functions $w_i(q)$, $q \in \Omega$.

Definition 4.1 (Sum Normalized (SN) type)**.** The vector $\mathbf{w}(q)$ containing the weighting functions is *Sum Normalized* if the sum of the weighting functions for all $p \in \Omega$ is 1.

Definition 4.2 (Non-Negative (NN) type)**.** The vector $\mathbf{w}(q)$ containing the weighting functions is *Non-Negative* if the values of the weighting functions for all is non-negative.

Definition 4.3 (Normal (NO) type)**.** The vector $\mathbf{w}(q)$ containing the weighting functions is *Normal* if it is SN and NN type and the largest value of all weighting functions is 1. As well as, it is *close to NO (CNO)* if it is SN and NN type and the largest value of all weighting functions is 1 or close to 1.

Definition 4.4 (Relaxed Normal (RNO) type)**.** The vector $\mathbf{w}(q)$ containing the weighting functions is *Relaxed Normal* if the largest values of all weighting functions are all the same (if the weighting functions are SN and NN, then this value is always between 0 and 1).

Definition 4.5 (Inverted Normal (INO) type)**.** The vector $\mathbf{w}(q)$ containing the weighting functions is *Inverted Normal* if the smallest value of all weighting function is 0.

Henceforward, the superscript of the weighting functions $\mathbf{w}(q)$ shows its type, such as $\mathbf{w}^{\text{SN}}(q)$, $\mathbf{w}^{\text{NN}}(q)$, $\mathbf{w}^{\text{NO}}(q)$, $\mathbf{w}^{\text{CNO}}(q)$, $\mathbf{w}^{\text{INO}}(q)$, and $\mathbf{w}^{\text{RNO}}(q)$ mean that the type of the weighting functions are SN, NN, NO, CNO, INO, or RNO, respectively. The convex weighting functions that are at least SN and NN types are indicated as $\mathbf{w}^{\text{CO}}(q)$.

The geometrical interpretation of these weighting functions are the following. In case of SN and NN types weighting functions, the LTI vertex systems in the resulting TP model define convex hull of the system matrix $\mathbf{S}(\mathbf{p}(t))$. The SN, NN, and NO types of weighting functions define the tight convex hull of the LTI vertex systems in such a way that the most LTI vertex systems are the same with $\mathbf{S}(\mathbf{p}(t))$ at different points of space Ω, and the others are close to it (in the sense of L_2-norm). The SN, NN, and RNO types of weighting functions result a tight convex hull where the LTI vertex systems are at the same distance to system matrix $\mathbf{S}(\mathbf{p}(t))$ in the space Ω. In case of SN, NN, and INO types weighting functions, it is guaranteed that there are points $\mathbf{p}(t)$ in the space Ω. In case of INO type it is guaranteed that in the space Ω there exist such points $\mathbf{p}(t)$ where the system matrix $\mathbf{S}(\mathbf{p}(t))$ is defined by the convex combination of a set of the vertex systems only. If the weighting functions are also RNO type, then it is guaranteed that those vertex systems that contribute in the convex combination are equal distance to the system matrix $\mathbf{S}(\mathbf{p}(t))$ in space Ω.

4.2 Computation of the different convex hulls by TP model transformation

As the continuous weighting functions are generated from matrices $\mathbf{U}_n$ these matrices contain the discretized weighting functions and resulted by the HOSVD in Method 3.1 Step 3 of the TP model transformation. It is shown in this section how the matrices $\mathbf{U}_n$ can be produced that will result SN, NN, RNO, or INO type weighting functions in Step 3 of the TP model transformation. Therefore, let us redefine these weighting function types for matrices.

Definition 4.6 (SN type matrix)**.** A real matrix $\mathbf{U}$ is SN type, if the sum of all elements in all rows are 1.

Definition 4.7 (NN type matrix)**.** A real matrix $\mathbf{U}$ is NN type, if it does not have any negative element.

Definition 4.8 (NO type matrix)**.** A real matrix $\mathbf{U}$ is NO type, if it is SN and NN, and the largest value of all columns is 1. The matrix is close to NO type if it is SN and NN, and the largest value of all columns is 1, or close to 1.

Definition 4.9 (RNO type matrix)**.** A real matrix $\mathbf{U}$ is RNO type, if the largest values of all columns are the same (if the matrix is SN and NN type, then this value is always between 0 and 1).

Definition 4.10 (INO type matrix)**.** A real matrix $\mathbf{U}$ is INO type, if the smallest value of all columns is 0.

Papers [6, 15, 90, 91] give a constructive proof that says there exists an invertible transformation T that generates the specific type of matrix $\mathbf{U}_n$. The construction of SN, NN and NO type matrices is detailed in [15, 90], while for INO and RNO type paper [91] gives satisfactory details. The construction of the NO type matrix is not possible in all cases if the number of columns is limited. Certainly, it is possible by the increase of columns as the identity matrix is also NO type. However, from the point of view of computational complexity the goal is to give an NO type matrix with as few columns as possible. In case of TP model transformation the close to NO type matrix in almost all cases gives satisfactory results.

Method 4.1 (TP model transformation extended with different type of weighting functions)**.** In Step 2 of the TP model transformation (Method 3.1), let us extend the execution of HOSVD of tensor $\mathcal{S}^D$ on dimensions $n = 1 \dots N$ by the generation of SN, NN, NO, CNO, RNO, or INO type matrices $\mathbf{U}_n$. Then, define the LTI vertex systems

$$\mathcal{S} = \mathcal{S}^D \bigotimes_{n=1}^{N} \mathbf{U}_n^+.$$

So, we obtain

$$\mathcal{S}^D \underset{\gamma}{\approx} \mathcal{S} \bigotimes_{n=1}^{N} \mathbf{U}_n,$$

where the matrices $\mathbf{U}_n$ can be arbitrarily SN, NN, NO, RNO, or INO types. The approximation error γ still can be bounded by (3.13) as the generation of SN, NN, NO, RNO, or INO type matrices do not change the error. The continuous weighting functions are generated in Step 3 of the algorithm. The resulting weighting functions inherit the SN, NN, NO, RNO, or INO type of matrices $\mathbf{U}_n$.

Chapter 5

Tensor-Product model transformation based control design methodology

One of the control design frameworks widely adopted to LPV models is based on the following two steps. In the first step a polytopic model is derived from the LPV model and in the second step a controller is generated to the polytopic model.

The TP model transformation based control design methodology is based on this structure. In the first step it applies the TP model transformation (Method 4.1) to obtain a polytopic model. The second step of the design methodology is based on the Parallel Distributed Compensation design concept. The key idea of this concept is to find the controller in the same polytopic structure as that of the model. This polytopic structure combines the LTI feedback gains derived from the LTI systems of the polytopic model. The feedback gains can be derived by various techniques. One of the most powerful design techniques capable of optimizing various desired control specifications is based on linear matrix inequalities (LMI). Therefore, the PDC is typically applied with LMIs. The LMI design developed under the PDC framework requires that the weighting functions define the convex combination, namely the convex hull, of the LTI systems. Furthermore, paper [6] examines that the type of the convex hull may considerably relax the feasibility of the further LMI design (for instance tight convex hull). In order to have a complete view, we give a brief introduction to the basic concepts of PDC and LMI optimization.

5.1 Parallel Distributed Compensation based control design framework

The *Parallel Distributed Compensation framework* (PDC) was proposed by Tanaka and Wang in 1995, [87]. It was developed for Takagi–Sugeno (TS) fuzzy decision operator-based models (TS fuzzy models) with antecedent fuzzy sets defined in Ruspini-partition. Book [80] introduces the PDC design with a number of LMI theorems for various control design specifications. In the last decade the number of LMI design theorems developed for the PDC design framework exploded.

The closed formulation of the transfer function of this kind of TS fuzzy model is equivalent to the TP model form where the weighting functions represent the membership functions of the antecedent sets given in Ruspini-partition. Therefore, from now on let us define this specific Ruspini-partition TS fuzzy model equivalent TP model as convex finite element TP model.

Definition 5.1 (Convex finite element TP model)**.** The finite element TP model (3.3) is convex only if the weighting functions are $\mathbf{w}_n^{\mathrm{CO}}(\mathbf{p}(t))$, namely

$$\forall n, i, p_n(t) \quad : \quad w_{n,i}(p_n(t)) \in [0,1] \tag{5.1}$$

$$\forall n, p_n(t) \quad : \quad \sum_{i=1}^{I_n} w_{n,i}(p_n(t)) = 1 \tag{5.2}$$

In conclusion, the transfer functions of the Ruspini-partitioned TS fuzzy model is equivalent to finite element convex TP model form. This simply means that the PDC design is immediately applicable to finite element convex TP models. Therefore, let us introduce the PDC design framework in the followings for the finite element convex TP models.

The PDC framework starts with

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{S}^{\mathrm{CO}} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n^{\mathrm{CO}}(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}, \tag{5.3}$$

and determines one LTI feedback to each LTI component of the TP model:

$$\mathcal{S}^{\mathrm{CO}} \longrightarrow \mathcal{F}$$

The control value is computed in the same TP structure as:

$$\mathbf{u}(t) = -\left(\mathcal{F} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n^{\mathrm{CO}}(p_n(t)) \right) \mathbf{x}(t). \tag{5.4}$$

Here, the weighting functions $\mathbf{w}_n^{\mathrm{CO}}(p_n(t))$ are equivalent to the weighting functions of the the convex TP model (5.3), and tensor $\mathcal{F}$ contains the feedback forces in a similar way as tensor $\mathcal{S}$.

There are several ways for finding the feedback gains $\mathcal{F}$ of the vertex models. As the vertex models are LTI systems, the well-known controller design techniques for linear systems *e.g.* pole placement, Linear-quadratic (LQ) control, sliding mode control, etc. can be used to determine the feedback gains. However, more recently, there is an explosive growth in the appearance of scientific works that propose new and more advantageous linear matrix inequality systems that can readily be integrated into the PDC framework, in terms of different control properties [78, 80, 86, 87]. The stability theorems formulated by linear matrix inequalities can be chosen according to the desired multi-objective specifications. Such goals can be, for instance, the quadratic or asymptotic stability, decay rate control, or the different constraints formulated on state vectors, outputs, and control signals.

5.2 Linear Matrix Inequality in system control design

Linear Matrix Inequalities (LMIs) and LMI techniques have emerged as powerful design tools in areas ranging from control engineering to system identification and structural design. Three factors make LMI techniques appealing:

- A variety of design specifications and constraints can be expressed as LMIs.
- Once formulated in terms of LMIs, a problem can be solved exactly by efficient convex optimization algorithms (the "LMI solvers").
- While most problems with multiple constraints or objectives lack analytical solutions in terms of matrix equations, they often remain tractable in the LMI framework. This makes LMI-based design a valuable alternative to classical "analytical" methods.

The most significant advantage of LMIs is that it is easy to numerically specify and combine numerous design constraints, conditions, and goals. Many control problems and design specifications have LMI formulations [20]. This is especially true for Lyapunov-based analysis and design, but also for optimal LQG control, H_∞ control, covariance control, etc. Further applications of LMIs arise in estimation, identification, optimal design, structural design, matrix scaling problems, and so on. The main strength of LMI formulations is the ability to combine various design constraints or objectives in a numerically tractable manner.

A non-exhaustive list of problems addressed by LMI techniques includes the following:

- Robust stability of systems with LTI uncertainty (μ-analysis) [67, 74, 92]
- Quadratic stability of differential inclusions [21, 44]
- Lyapunov stability of parameter-dependent systems [35]
- Input/state/output properties of LTI systems (invariant ellipsoids, decay rate, etc.) [20]
- Multi-model/multi-objective state feedback design [4, 16, 20, 26, 51]
- Robust pole placement
- Optimal Linear Quadratic Gaussian control [20]
- Robust H_∞ control [34, 46]
- Multi-objective H_∞ synthesis [26, 51, 62]
- Control of stochastic systems [20]
- Weighted interpolation problems [20]

In the following section, a short introduction to linear matrix inequalities is given.

5.2.1 Definition of LMI

Before proceeding, we give the definition of a term used often in the literature. This will be followed by the introduction of the linear matrix inequalities.

Definition 5.2 (Affine function)**.** A function $f : \mathcal{S} \mapsto \mathcal{T}$ is *affine* if $f(x) = f_0 + T(x)$ where $f_0 \in \mathcal{T}$ and $T : \mathcal{S} \mapsto \mathcal{T}$ is a linear map, i.e.,

$$T(\alpha_1 x_1 + \alpha_2 x_2) = \alpha_1 T(x_1) + \alpha_2 T(x_2)$$

for all $x_1, x_2 \in \mathcal{S}$ and $\alpha_1, \alpha_2 \in \mathbb{R}$.

Hence $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is affine if and only if there exists $\mathbf{x}_0 \in \mathbb{R}^n$, such that the mapping $\mathbf{x} \mapsto f(\mathbf{x}) - f(\mathbf{x}_0)$ is linear. This means that all affine functions $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ can be represented as $f(\mathbf{x}) = f(\mathbf{x}_0) + T \cdot (\mathbf{x} - \mathbf{x}_0)$ where T is some matrix of dimension $m \times n$ and the dot $\cdot$ denotes multiplication. We will be interested in the case where $m = 1$ and denote by $\langle \cdot, \cdot \rangle$ the standard inner product in $\mathbb{R}^n$, that is, for $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$, $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle = \mathbf{x}_2^T \mathbf{x}_1$.

A *linear matrix inequality* is an expression of the form:

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + x_1 \mathbf{F}_1 + \cdots + x_m \mathbf{F}_m > 0, \tag{5.5}$$

where

1. $\mathbf{x} = (x_1, \ldots, x_m)$ is a vector of m real numbers called the *decision variables*.
2. $\mathbf{F}_0, \ldots, \mathbf{F}_m$ are real symmetric matrices, i.e., $\mathbf{F}_i = \mathbf{F}_i^T \in \mathbb{R}^{n \times n}$, $i = 0, \ldots, m$ for some $n \in \mathbb{Z}_+$-re.
3. the inequality > 0 in (5.5) means 'positive definite', i.e., $\mathbf{u}^T \mathbf{F}(\mathbf{x}) \mathbf{u} > 0$ for all $\mathbf{u} \in \mathbb{R}^n, \mathbf{u} \neq 0$. Equivalently, the smallest eigenvalue of $\mathbf{F}(\mathbf{x})$ is positive.

In more general terms

Definition 5.3 (Linear Matrix Inequalities)**.** A *linear matrix inequality* (LMI) is an inequality

$$\mathbf{F}(\mathbf{x}) > 0 \tag{5.6}$$

where $\mathbf{F}$ is an *affine function* mapping a finite dimensional vector space $\mathbb{V}$ to the set $\mathbb{S} := \{M \mid \exists n > 0 \text{ such that } \mathbf{M} = \mathbf{M}^T \in \mathbb{R}^{n \times n}\}$, of real symmetric matrices.

Remark 5.1. An affine mapping $\mathbf{F} : \mathbb{V} \mapsto \mathbb{S}$ necessarily takes the form $\mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \mathbf{T}(\mathbf{x})$ where $\mathbf{F} \in \mathbb{S}$ and $\mathbf{T} : \mathbb{V} \mapsto \mathbb{S}$ is a linear transformation. Thus if $\mathbb{V}$ is finite dimensional, say of dimension m, and $\{e_1, \ldots, e_m\}$ constitutes a basis for $\mathbb{V}$, then we can write

$$\mathbf{T}(\mathbf{x}) = \sum_{j=1}^{m} x_j \mathbf{F}_j$$

where the elements $\{x_1, \ldots, x_m\}$ are such that $\mathbf{x} = \sum_{j=1}^m x_j e_j$ and $\mathbf{F}_j = \mathbf{T}(e_j)$ for $j = 1, \ldots, m$. Hence we obtain (5.5) as a special case.

Remark 5.2. The same remark applies to affine mappings $\mathbf{F} : \mathbb{R}^{n\times n} \mapsto \mathbb{S}$. A simple example is the Lyapunov inequality $\mathbf{F}(\mathbf{X}) = \mathbf{A}^T\mathbf{X} + \mathbf{X}\mathbf{A} + \mathbf{Q} > 0$. Here, $\mathbf{A}, \mathbf{Q} \in \mathbb{R}^{n\times n}$ are assumed to be given and $\mathbf{X} \in \mathbb{R}^{n\times n}$ is unknown. The unknown variable is therefore a *matrix*. Note that this defines an LMI only if $\mathbf{Q}$ is symmetric. We can view this LMI as a special case of (5.5) by defining a basis $\mathbf{E}_1, \ldots, \mathbf{E}_m$ of $\mathbb{V}$ and writing $\mathbf{X} = \sum_{j=1}^m x_j\mathbf{E}_j$. Indeed,

$$\mathbf{F}(\mathbf{X}) = \mathbf{F}\left(\sum_{j=1}^m x_j\mathbf{E}_j\right) = \mathbf{F}_0 + \sum_{j=1}^m x_j\mathbf{F}(\mathbf{E}_j) = \mathbf{F}_0 + \sum_{j=1}^m x_j\mathbf{F}_j$$

which is of the form (5.5).

Remark 5.3. A *non-strict* LMI is a linear matrix inequality where $>$ in (5.5) and (5.6) is replaced by $\geq$. The matrix inequalities $\mathbf{F}(\mathbf{x}) < 0$, and $\mathbf{F}(\mathbf{x}) > \mathbf{G}(\mathbf{x})$ with $\mathbf{F}$ and $\mathbf{G}$ affine functions are obtained as special cases of Definition 5.3 as they can be rewritten as the linear matrix inequality $-\mathbf{F}(\mathbf{x}) > 0$ and $\mathbf{F}(\mathbf{x}) - \mathbf{G}(\mathbf{x}) > 0$.

5.2.2 Constraints expressed using LMI

The linear matrix inequality (5.6) defines a *convex constraint* on $\mathbf{x}$. That is, the set $\mathcal{S} := \{\mathbf{x} \mid \mathbf{F}(\mathbf{x}) > 0\}$ is convex. Indeed, if $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S}$ and $\alpha \in (0, 1)$ then

$$\mathbf{F}(\alpha x_1 + (1-\alpha)x_2) = \alpha\mathbf{F}(x_1) + (1-\alpha)\mathbf{F}(x_2) > 0,$$

where in the equality we used that $\mathbf{F}$ is affine and the inequality follows from the fact that $\alpha \geq 0$ and $(1-\alpha) \geq 0$.

Although the convex constraint $\mathbf{F}(\mathbf{x}) > 0$ on $\mathbf{x}$ may seem rather special, it turns out that many convex sets can be represented in this way. In this subsection we discuss some seemingly trivial properties of linear matrix inequalities which turn out to be of eminent help in the reduction of multiple constraints on an unknown variable to an equivalent constraint involving a single linear matrix inequality.

Definition 5.4 (System of LMIs)**.** A *system of linear matrix inequalities* is a finite set of linear matrix inequalities

$$\mathbf{F}_1(\mathbf{x}) > 0, \mathbf{F}_2(\mathbf{x}) > 0, \ldots, \mathbf{F}_k(\mathbf{x}) > 0. \tag{5.7}$$

It is a simple but essential property that every system of LMIs can be rewritten as one single LMI. Specifically, $\mathbf{F}_1(\mathbf{x}) > 0, \mathbf{F}_2(\mathbf{x}) > 0, \ldots, \mathbf{F}_k(\mathbf{x}) > 0$ if and only if

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} \mathbf{F}_1(\mathbf{x}) & 0 & \ldots & 0 \\ 0 & \mathbf{F}_2(\mathbf{x}) & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & \mathbf{F}_k(\mathbf{x}) \end{pmatrix} > 0.$$

The last inequality indeed makes sense as $\mathbf{F}(\mathbf{x})$ is symmetric for any $\mathbf{x}$. Furthermore, since the set of eigenvalues of $\mathbf{F}(\mathbf{x})$ is simply the union of the eigenvalues of $\mathbf{F}_1(\mathbf{x}), \ldots, \mathbf{F}_k(\mathbf{x})$, any $\mathbf{x}$ that satisfies $\mathbf{F}(\mathbf{x}) > 0$ also satisfies the system of LMI's (5.7) and vice versa.

A second important property amounts to incorporating *affine constraints* in linear matrix inequalities. By this, we mean that *combined constraints* (in the unknown $\mathbf{x}$) of the form

$$\begin{cases} \mathbf{F}(\mathbf{x}) > 0 \\ \mathbf{A}\mathbf{x} = \mathbf{b} \end{cases}$$

or

$$\begin{cases} \mathbf{F}(\mathbf{x}) > 0 \\ \mathbf{x} = \mathbf{A}\mathbf{y} + \mathbf{b} \text{ for some } \mathbf{y} \end{cases}$$

where the affine function $\mathbf{F} : \mathbb{R}^m \mapsto \mathbb{S}$ and matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$ are given can be *lumped* in one linear matrix inequality $\mathbf{F}(\mathbf{x}) > 0$. More generally, the combined equations

$$\begin{cases} \mathbf{F}(\mathbf{x}) > 0 \\ \mathbf{x} \in \mathcal{M} \end{cases} \tag{5.8}$$

where $\mathcal{M}$ is an *affine subset of* $\mathbb{R}^m$ i.e.,

$$\mathcal{M} = \mathbf{x}_0 + \mathcal{M}_0 = \{\mathbf{x}_0 + \mathbf{m}_0 \mid \mathbf{m}_0 \in \mathcal{M}_0\},$$

with $\mathbf{x}_0 \in \mathbb{R}^m$ and $\mathcal{M}_0$ a linear subspace of $\mathbb{R}^m$, can be rewritten in the form of one single linear matrix inequality $\mathbf{F}(\mathbf{x}) > 0$. To actually do this, let $\mathbf{e}_1, \ldots, \mathbf{e}_{m_0} \in \mathbb{R}^m$ be a basis of $\mathcal{M}_0$ and let $\mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \mathbf{T}(\mathbf{x})$ be decomposed as in Remark 5.1. Then (5.8) can be rewritten as

$$\begin{aligned} 0 \;<\; \mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \mathbf{T}\left(\mathbf{x}_0 + \sum_{j=1}^{m_0} x_j \mathbf{e}_j\right) &= \underbrace{\mathbf{F}_0 + \mathbf{T}(\mathbf{x}_0)}_{\text{constant part}} + \underbrace{\sum_{j=1}^{m_0} x_j \mathbf{T}(\mathbf{e}_j)}_{\text{linear part}} \\ &= \overline{\mathbf{F}}_0 + \mathbf{x}_1 \overline{\mathbf{F}}_1 + \cdots + \mathbf{x}_{m_0} \overline{\mathbf{F}}_{m_0} \\ &= \overline{\mathbf{F}}(\overline{\mathbf{x}}) \end{aligned}$$

where $\overline{\mathbf{F}}_0 = \mathbf{F}_0 + \mathbf{T}(\mathbf{x}_0), \overline{\mathbf{F}}_j = \mathbf{T}(\mathbf{e}_j)$ and $\overline{\mathbf{x}} = (x_1, \ldots, x_{m_0})$ are the coefficients of $\mathbf{x} - \mathbf{x}_0$ in the basis of $\mathcal{M}_0$. This implies that $\mathbf{x} \in \mathbb{R}^m$ satisfies (5.8) if and only if $\overline{\mathbf{F}}(\overline{\mathbf{x}}) > 0$. Note that the dimension m_0 of $\overline{\mathbf{x}}$ is smaller than the dimension m of $\mathbf{x}$.

A third property of LMIs is obtained from a simple exercise in algebra. It turns out to be possible to convert some *non-linear* inequalities to *linear inequalities*. Suppose that we partition a matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ as

$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{pmatrix},$$

where $\mathbf{M}_{11}$ has dimension $r \times r$. Assume that $\mathbf{M}_{11}$ is non-singular. Then the matrix $\mathbf{S} = \mathbf{M}_{22} - \mathbf{M}_{21}\mathbf{M}_{11}^{-1}\mathbf{M}_{12}$ is called the *Schur-complement* of $\mathbf{M}_{11}$ in $\mathbf{M}$. If $\mathbf{M}$ is symmetric then we have that

$$\begin{aligned} \mathbf{M} > 0 &\iff \begin{pmatrix} \mathbf{M}_{11} & 0 \\ 0 & \mathbf{S} \end{pmatrix} > 0 \\ &\iff \begin{cases} \mathbf{M}_{11} > 0 \\ \mathbf{S} > 0 \end{cases}. \end{aligned}$$

In conclusion we can state the following proposition.

Proposition 5.1 (Schur complement)**.** *Let* $\mathbf{F} : \mathbb{V} \mapsto \mathbb{S}$ *be an affine function which is partitioned according to*

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} \mathbf{F}_{11}(\mathbf{x}) & \mathbf{F}_{12}(\mathbf{x}) \\ \mathbf{F}_{21}(\mathbf{x}) & \mathbf{F}_{22}(\mathbf{x}) \end{pmatrix},$$

where $\mathbf{F}_{11}(\mathbf{x})$ *is square. Then* $\mathbf{F}(\mathbf{x}) > 0$ *if and only if*

$$\begin{cases} \mathbf{F}_{11}(\mathbf{x}) > 0 \\ \mathbf{F}_{22}(\mathbf{x}) - \mathbf{F}_{12}(\mathbf{x}) \left[\mathbf{F}_{11}(\mathbf{x})\right]^{-1} \mathbf{F}_{21}(\mathbf{x}) > 0 \end{cases}. \tag{5.9}$$

Note that the second inequality in (5.8) is a *non-linear* matrix inequality in $\mathbf{x}$. Using this result, it follows that non-linear matrix inequalities of the form (5.8) can be converted to linear matrix inequalities. In particular, it follows that non-linear inequalities of the form (5.8) define a convex constraint on the variable $\mathbf{x}$ in the sense that all x satisfying (5.8) define a convex set.

5.2.3 Application-oriented forms of LMI

As the publications listed in the introduction show, many optimization problems in control, identification and signal processing can be formulated (or reformulated) using linear matrix inequalities. Clearly, it only makes sense to cast these problems in an LMI setting if these inequalities can be solved in an efficient and reliable way. Since the linear matrix inequality $\mathbf{F}(\mathbf{x}) > 0$ defines a *convex constraint* on the variable $\mathbf{x}$, optimization problems involving the minimization (or maximization) of a performance function $f : \mathcal{S} \mapsto \mathbb{R}$ with $\mathcal{S} := \{\mathbf{x} \mid \mathbf{F}(\mathbf{x}) > 0\}$ belong to the class of *convex optimization problems*. Casting this in the setting of the previous section, it may be apparent that the full power of convex optimization theory can be employed if the performance function f is known to be convex.

Suppose that $\mathbf{F}, \mathbf{G}, \mathbf{H} : \mathbb{V} \mapsto \mathbb{S}$ are affine functions. There are three generic problems related to the study of linear matrix inequalities:

1. **Feasibility:** The test whether or not there exist solutions $\mathbf{x} \in \mathbb{V}$ of $\mathbf{F}(\mathbf{x}) > 0$ is called a *feasibility problem*. The LMI is called *feasible* if such $\mathbf{x}$ exists, otherwise the LMI $\mathbf{F}(\mathbf{x}) > 0$ is said to be *infeasible*.

2. **Optimization:** Let $f : \mathcal{S} \mapsto \mathbb{R}$ and suppose that $\mathcal{S} = \{\mathbf{x} \mid \mathbf{F}(\mathbf{x}) > 0\}$. The problem to determine
$$V_{\text{opt}} = \inf_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x})$$
is called an *optimization problem with an LMI constraint*. This problem involves the determination of V_{opt} and for arbitrary $\varepsilon > 0$ the calculation of an *almost optimal solution* $\mathbf{x}$ which satisfies $\mathbf{x} \in \mathcal{S}$ and $V_{\text{opt}} \leq f(\mathbf{x}) \leq V_{\text{opt}} + \varepsilon$.

3. **Generalized eigenvalue problem:** The generalized eigenvalue problem amounts to minimizing a scalar $\lambda \in \mathbb{R}$ subject to
$$\begin{cases} \lambda \mathbf{F}(\mathbf{x}) - \mathbf{G}(\mathbf{x}) > 0 \\ \mathbf{F}(\mathbf{x}) > 0 \\ \mathbf{H}(\mathbf{x}) > 0 \end{cases}.$$

5.2.4 Computation of LMIs

A major breakthrough in convex optimization lies in the introduction of interior-point methods. These methods were developed in a series of papers [48] and became of true interest in the context of LMI problems in the work of Yurii Nesterov and Arkadii Nemirovskii [65].

The mathematical apparatus used in the resolution is not covered in this dissertation due to its lengthiness and complexity. Detailed descriptions are given in [48, 65]. The interior-point method is the most popular optimization technique thanks to its efficiency. It is also widely used in commercial scientific applications such as MATLAB Optimization Toolbox and Robust Control Design Toolbox [37].

Part II

Theoretical Achievements

Chapter 6

Complexity relaxation of Tensor-Product model transformation

A crucial point of the TP model transformation is that its computational load explodes easily with the dimensionality of the parameter vector of LPV model. The reason for the computational explosion is that Step 2 of the transformation (see Section 3.4) executes Higher Order Singular Value Decomposition on a large size multi-dimensional tensor resulted by the discretization of a given LPV model. The number of dimensions of the tensor equals the number of elements in the parameter vector. If the number of these elements is high (this practically means more than 4 or 5) then a regular computer cannot execute HOSVD on the discretized tensor, hence, the TP model transformation neither, as well. For instance, in practical cases in order to generate a fine convex hull of the TP model at least 100 points along each dimension is necessary [6, 7, 14, 68, 69]. This results a discretized tensor with a size of 100^4 if the number of dimensions is 4.

This chapter proposes two ways for relaxing the computational load problem of the TP model transformation. The goal of both ways is to decrease the number of elements of the discretized tensors that directly result "the decrease of" the computational load of the HOSVD. The first proposes an approach to considerable decrease the discretized grid-density and modifies the algorithm of the TP model transformation accordingly. The key idea of the second way is to restrict the computation of the TP model transformation to the non-constant elements only, which means that TP model transformation is executed on a discretized subtensor only. These two main approaches modify the TP model transformation at different levels, thus the computational load can be decreased even more with their combination.

6.1 Decreasing the discretization grid density

The denser the hyper-rectangular grid Θ applied in the transformation space Ω, the more accurate will be the results obtained. The dense grid allows a more accurate numerical reconstruction of the weighting functions and a finer convex hull, hence, a more appropriate TP model (see Section 3.4). However, the complexity (number of elements) of the discretized tensor equals the complexity (number of sampling points) of the discretization grid Θ (see

Step 1 of Method 3.1). The computational load of HOSVD explodes with the complexity of the discretized tensor.

According to the above, we have two contradictory constraints to define the density of Θ. The first constraint forces us to use dense Θ to achieve an appropriate decomposition of $\mathbf{S}(\mathbf{p}(t))$ with fine convex hulls. The second constraint comes from the easily exploding computational load of the HOSVD.

The key idea comes from the practical considerations as we cannot apply more than a few weighting functions (up to about 4–5) per dimension, because the computational load of the further LMI based controller design also explodes easily with the number of the resulting LTI systems, and therefore we may not be able to solve them. Moreover, a few weighting functions per dimension are practically enough to have an exact or close-to-exact transformation. Thus, the first approach we limit the TP model transformation to the search of a few weighting functions by dimensions only.

6.1.1 The algorithm of the modified TP model transformation

The key point of the modification is that we apply two discretization grids Θ and Γ. Θ is a sparse grid upon which we discretize the $\mathbf{S}(\mathbf{p}(t))$ and generate $\mathbf{S}^{D,\Theta}$. Superscript "Θ" indicates that $\mathbf{S}^{D}$ is the discretization over Θ. Then we execute HOSVD on the "small sized" $\mathbf{S}^{D,\Theta}$ to yield the minimal number of LTI systems and do the approximation trade-off if it is necessary. So, Θ is defined according to the maximum computational capacity available for HOSVD. After this we introduce a new step capable of expanding the resulting decomposition to the dense grid Γ without generating $\mathbf{S}^{D,\Gamma}$ by discretization (which would be an extremely large tensor) and executing HOSVD on $\mathbf{S}^{D,\Gamma}$. Once we have the decomposition over Γ then we can generate "fine" convex hulls as in the original TP model transformation. The last step generates the continuous weighting functions in the same way as in the original TP model transformation.

Step 1) Discretization over a sparse grid Θ

- Define the transformation space Ω as: $\mathbf{p}(t) \in \Omega : [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_N, b_N]$.
- Define a hyper-rectangular N-dimensional sparse grid Θ using equidistantly located grid-lines: $\theta_{n,h_n} = a_n + \frac{b_n - a_n}{H_n - 1}(h_n - 1)$, $h_n = 1 \dots H_n$. The numbers of the grid lines in the dimensions are H_n. Notice here that the H_n must be larger or equal to the allowed maximum number of weighting functions in each dimension of the TP model to be generated. The reason for this is that the TP model transformation in Step 2 cannot generate more weighting functions then H_n, because the number of weighting functions are equal to the number of the remaining singular values in each dimension.
- Sample the given function $\mathbf{S}(\mathbf{p}(t))$ over the grid-points of Θ:
$$\mathbf{S}^{D,\Theta}_{h_1,h_2,\dots,h_N} = \mathbf{S}(\mathbf{p}_{h_1,h_2,\dots,h_N}) \in \mathbb{R}^{O \times I},$$
where $\mathbf{p}_{h_1,h_2,\dots,h_N} = \begin{pmatrix} \theta_{1,h_1} & \theta_{2,h_2} & \dots & \theta_{N,h_N} \end{pmatrix}$.
- Store the discretized matrices $\mathbf{S}^{D,\Theta}_{h_1,h_2,\dots,h_N}$ into the tensor $\mathcal{S}^{D,\Theta} \in \mathbb{R}^{H_1 \times H_2 \times \cdots \times H_N \times O \times I}$

Step 2) Approximation trade-off via HOSVD In this step we execute HOSVD on the first N dimensions of tensor $\mathcal{S}^{D,\Theta}$. While performing the HOSVD we discard all zero or small singular values σ_k and their corresponding singular vectors in all dimensions, as in the original TP model transformation. Assume that we keep I_n singular values in dimensions $n = 1 \dots N$. As a result we have

$$\mathcal{S}^{D,\Theta} \underset{\delta}{\approx} \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Theta}. \tag{6.1}$$

The error δ is bounded as (3.13). The resulting tensor $\mathcal{S}$, with the size of $(I_1 \times I_2 \times \cdots \times I_N \times O \times I)$, where $\forall n : I_n \leq H_n$, contains the first variant of the LTI vertex systems, and is immediately substitutable into (3.3). The size of $\mathbf{U}_n^{\Theta}$ is $H_n \times I_n$.

Step 3) Increasing the grid density to Γ This step expands the result of the second step to the dense discretization grid Γ. Define a hyper-rectangular dense grid Γ using equidistantly located grid-lines: $\gamma_{n,m_n} = a_n + \frac{b_n - a_n}{M_n - 1}(m_n - 1)$, $m_n = 1 \dots M_n$. The numbers of the grid lines in the dimensions are M_n. Define M_n in such a way that all grid points of Θ are in Γ, namely $\Theta \subset \Gamma$.

The goal is to generate the HOSVD of $\mathcal{S}^{D,\Gamma}$ as:

$$\mathcal{S}^{D,\Gamma} \underset{\varepsilon}{\approx} \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Gamma}, \tag{6.2}$$

where the size of $\mathbf{U}_n^{\Gamma}$ is $M_n \times I_n$ ($\forall n : M_n \gg H_n$) and the rows of the $\mathbf{U}_n^{\Gamma}$ are assigned to the grid-lines of Γ. We have $\varepsilon = 0$ if the rank of $\mathcal{S}^{D,\Gamma}$ is equal to maximum rank constraint I_n on dimensions $n = 1 \dots N$. If it has higher rank then we obtain $\varepsilon > 0$. However, we do not have $\mathcal{S}^{D,\Gamma}$ since its size and especially the computational load of HOSVD would transcend the capacity of a regular computer. Therefore, in order to avoid the execution of HOSVD on $\mathcal{S}^{D,\Gamma}$ we compute each row of $\mathbf{U}_n^{\Gamma}$ one by one from $\mathcal{S}$ and $\mathbf{S}(\mathbf{p}(t))$. We determine the m_dth row (assigned to the grid line γ_{d,m_d}) in matrix $\mathbf{U}_d^{\Gamma}$ in such a way that let p_k be fixed to one θ_{k,h_k} of the grid-lines in each dimensions $k = 1 \dots N, k \neq d$ of discretization grid Θ except the dth dimension as:

$$p_k = \theta_{k,h_k} \quad k = 1 \dots N, \quad k \neq d,$$

where $h_k \in \{1, \dots, H_k\}$ is arbitrarily selected. We may simply select $h_k = 1$ in practical cases if there is no special reason not to. Then we create vector

$$\mathbf{p}(t) = \begin{pmatrix} \theta_{1,h_1} & \theta_{2,h_2} & \dots & \gamma_{d,m_d} & \dots & \theta_{N,h_N} \end{pmatrix}.$$

The new row $\mathbf{u}_{d,m_d}^{\Gamma}$ of matrix $\mathbf{U}_d^{\Gamma}$ is computed as:

$$\mathbf{u}_{d,m_d}^{\Gamma} = (\mathbf{S}(\mathbf{p}(t)))_{(1)} \left(\left(\mathcal{S} \underset{k}{\otimes} \mathbf{u}_{k,h_k}^{\Theta} \right)_{(d)} \right)^{+}, \tag{6.3}$$

where superscript "$(\cdot)^{+}$" denotes pseudo inverse, and $\mathbf{u}_{k,h_k}^{\Theta}$ is the h_kth row vector of $\mathbf{U}_k^{\Theta}$. The first-mode matrix $(\mathbf{S}(\mathbf{p}(t)))_{(1)}$ of matrix $\mathbf{S}(\mathbf{p}(t))$ is understood such that matrix $\mathbf{S}(\mathbf{p}) \in \mathbb{R}^{O \times I}$ is considered as a three-dimensional tensor $\mathcal{S}(\mathbf{p}) \in \mathbb{R}^{1 \times O \times I}$, where the length of the first

dimension is one. $\mathbf{S}(\mathbf{p}(t))_{(1)}$ practically means that the matrix $\mathbf{S}(\mathbf{p}(t))$ is stored into one row vector by placing the rows of $\mathbf{S}(\mathbf{p}(t))$ next to each other, respectively, see [60].

As a result we now have a decomposition

$$\mathcal{S}^{D,\Gamma} \underset{\varepsilon}{\approx} \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Gamma},$$

in which the matrices $\mathbf{U}_n$ may not be the same as could be resulted by HOSVD in (6.2), however we kept the I_n rank constraint and yield the best approximation of the new rows in $\mathbf{U}_n$ in the sense of L_2 norm. This is guaranteed by the use of pseudo inverse. Note that if $\delta = 0$ in (6.1) and $\mathcal{S}^{D,\Gamma}$ has the same rank as $\mathcal{S}^{D,\Theta}$ in each dimension then $\varepsilon = 0$, see for instance the example in the next section.

Step 4) Generating different convex hulls This step executes SN (Sum Normalization), NN (Non Negativeness), and NO (Normality) or INO–RNO (Inverse-NO and Relaxed-NO) transformations [6, 18, 84, 90, 91] according to the desired type of the convex hull. The previous step results in $\mathbf{U}_n^{\Gamma}$. The goal is here to transform matrices $\mathbf{U}_n^{\Gamma}$ to $\bar{\mathbf{U}}_n^{\Gamma}$, where the bar over matrix $\mathbf{U}$ means that the matrix is SN, NN NO or of any other type according to the executed transformation. Note that the Yam's type SN transformation [90, 91] needs the discarded singular values computed by the HOSVD. Since we do not have the singular values of $\mathcal{S}^{D,\Gamma}$ we use transformation techniques proposed in papers [6] and [84]. These techniques do not use any information about the result of HOSVD, but matrices $\mathbf{U}^{\Gamma}$ only. As a matter of fact when we transform matrices $\mathbf{U}_n^{\Gamma}$ to $\bar{\mathbf{U}}_n^{\Gamma}$ then we must modify $\mathcal{S}$ to $\bar{\mathcal{S}}$ accordingly, in order to keep:

$$\mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Gamma} = \bar{\mathcal{S}} \underset{n}{\otimes} \bar{\mathbf{U}}_n^{\Gamma},$$

$\bar{\mathcal{S}}$ could be readily computed form a $\mathcal{S}^{D,\Gamma}$ as:

$$\bar{\mathcal{S}} = \mathcal{S}^{D,\Gamma} \underset{n}{\otimes} \left(\bar{\mathbf{U}}_n^{\Gamma}\right)^{+},$$

however, as mentioned we do not have $\mathcal{S}^{D,\Gamma}$, but we have $\mathcal{S}^{D,\Theta}$. Therefore, let us select those rows of $\bar{\mathbf{U}}_n^{\Gamma}$ which are also assigned to the sparse grid Θ and then store this information into matrix $\bar{\mathbf{U}}_n^{\Theta}$ (note that if $\bar{\mathbf{U}}_n^{\Gamma}$ is SN, NN, NO, RNO or INO–RNO type then $\bar{\mathbf{U}}_n^{\Theta}$ will also be SN, NN, NO, RNO or INO–RNO type, respectively). Thus, the tensor $\bar{\mathcal{S}}$ can be computed by:

$$\bar{\mathcal{S}} = \mathcal{S}^{D,\Theta} \underset{n}{\otimes} \left(\bar{\mathbf{U}}_n^{\Theta}\right)^{+}.$$

Finally we have

$$\mathcal{S}^{D,\Theta} \underset{\delta}{\approx} \bar{\mathcal{S}} \underset{n}{\otimes} \bar{\mathbf{U}}_n^{\Theta} = \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Theta}.$$

then

$$\mathcal{S}^{D,\Gamma} \underset{\varepsilon}{\approx} \bar{\mathcal{S}} \underset{n}{\otimes} \bar{\mathbf{U}}_n^{\Gamma} = \mathcal{S} \underset{n}{\otimes} \mathbf{U}_n^{\Gamma}.$$

Step 5) Generating the continuous weighting functions This step is equivalent with the last step of the original TP model transformation. It results in the continuous weighting functions such as

$$\mathbf{S}(\mathbf{p}(t)) \approx \bar{\mathcal{S}} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n(p_n(t)). \tag{6.4}$$

Finally we numerically check the accuracy of the resulting TP model over a huge number of points in Ω. We also check the SN, NN, NO, RNO and INO–RNO type of the weighting functions via numerical computation. In the case of exact transformation the best error we can achieve is in the range of about 10^{-13} that is caused by the numerical computation of the HOSVD.

6.1.2 Evaluation of the calculation complexity reduction

In this subsection we compare the computational complexity of the original and the modified TP model transformation. Since the computation problem is caused by the computational explosion of the HOSVD, we focus attention on the number of elements in $\mathcal{S}^D$ in both cases.

We denote the computational load of SVD of a matrix with R number of elements by $C_{\text{SVD}}(R)$. The original TP model transformation executes SVD N times on a matrix with R number of elements:

$$C_{\text{original}} = N \times C_{\text{SVD}}(O \times I \times \prod_{n=1}^{N} M_n)$$

In practical cases M_n is about 100, see papers [6–9, 13, 42, 68]. The modified TP model transformation executes SVD N times as well, however the number of the elements of the matrix whereupon SVD is executed could be considerable less than in the case of the original TP model transformation. The modified TP model transformation executes HOSVD on $\mathcal{S}^D$ with the size of $H_1 \times H_2 \times \cdots \times H_N \times O \times I$, where H_n is about 2–5. The computation of the modified algorithm is

$$C_{\text{modified}} = N \times C_{\text{SVD}}(O \times I \times \prod_{n=1}^{N} H_n)$$

The algorithm offers a polynomial computational complexity reduction.

In higher dimensions even the discretization step of the original TP model transformation fails, since the storage of $\mathcal{S}^D$ requires a huge memory capacity in memory; a requirement which exceeds the limits of today's computers.

The next section gives a 6-dimensional example where the original TP model transformation cannot be applied because of the overwhelming computational complexity, but where the modified TP model transformation can easily be executed.

6.1.3 Numerical example

In this section we demonstrate the complexity relaxation of the firsts approach on a numerical example.

Consider the following dynamic model:

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t))\mathbf{x}(t),$$

where

$$\mathbf{S}(\mathbf{p}(t)) = \begin{bmatrix} s_{1,1} & 2 & s_{1,3} & -1 & 0 & 1 & 0 & s_{1,8} \\ s_{2,1} & 0 & 1 & 3 & s_{2,5} & 0 & 0 & 1 \\ s_{3,1} & 1 & 3 & 0 & 0 & 5 & s_{3,7} & 1 \\ -2 & 0 & 3 & s_{4,4} & 0 & -2 & 0 & 1 \\ 0 & s_{5,2} & 5 & 10 & -2 & s_{5,6} & 0 & 1 \\ 0 & 3 & 0 & s_{6,4} & 1 & -1 & 0 & 1 \\ 5 & 1 & 0 & s_{7,4} & 0 & -1 & 0 & 1.5 \\ s_{8,1} & 1 & s_{8,3} & 1 & 0 & 1 & 0 & s_{8,8} \end{bmatrix} \tag{6.5}$$

$$\begin{aligned}
s_{1,1} &= x_1 + x_1^2 + x_2^3 + x_2^4 \\
s_{1,3} &= \cos(x_3) \\
s_{1,8} &= x_7(2 + x_5) \\
s_{2,1} &= 3 + x_2^2 + x_3^3 \\
s_{2,5} &= x_6 \cos(x_5)^2 \\
s_{3,1} &= \frac{x_1^3 + x_1^2}{1 - \sin(x_2)} + x_3^4 \cos(x_3) + 4 \\
s_{3,7} &= x_2 + x_4 \\
s_{4,4} &= \sin(x_4)\cos(x_5) \\
s_{5,2} &= x_1^2 x_6 \\
s_{5,6} &= x_6^2 \cos(x_4) \\
s_{6,4} &= x_1 x_3 x_4 \\
s_{7,4} &= x_8 \sin(x_4) \\
s_{8,1} &= x_6 \\
s_{8,3} &= \frac{\cos(x_8)}{1 - x_7} \\
s_{8,8} &= \frac{x_3^3}{2 - x_6 - x_7}
\end{aligned}$$

Observe that the above system matrix $\mathbf{S}(\mathbf{p}(t))$ nonlinearly depends on all elements of the state vector $\mathbf{x}(t) \in \mathbb{R}^6$. This means that $N = 6$ and $\mathbf{p}(t) = \mathbf{x}(t)$. Let the transformation space be $\Omega : [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_N, b_N]$, where $\forall n : a_n = -0.5$ and $\forall n : b_n = -a_n$. First we apply the original TP model transformation and we define 129 grid-lines for each dimension for discretization. The discretization result in $\mathcal{S}^D$ whose number of elements is $36 \cdot 129^6$. A regular software (for instance MATLAB) and a computer cannot store a tensor as large as $\mathcal{S}^D$, and even if it could, HOSVD could not be executed. In conclusion the original TP model transformation cannot be executed in the present case.

Let us apply the modified TP model transformation. We use the same Ω and we require the same grid density with 129 grid-lines for each dimension. First we define the sparse

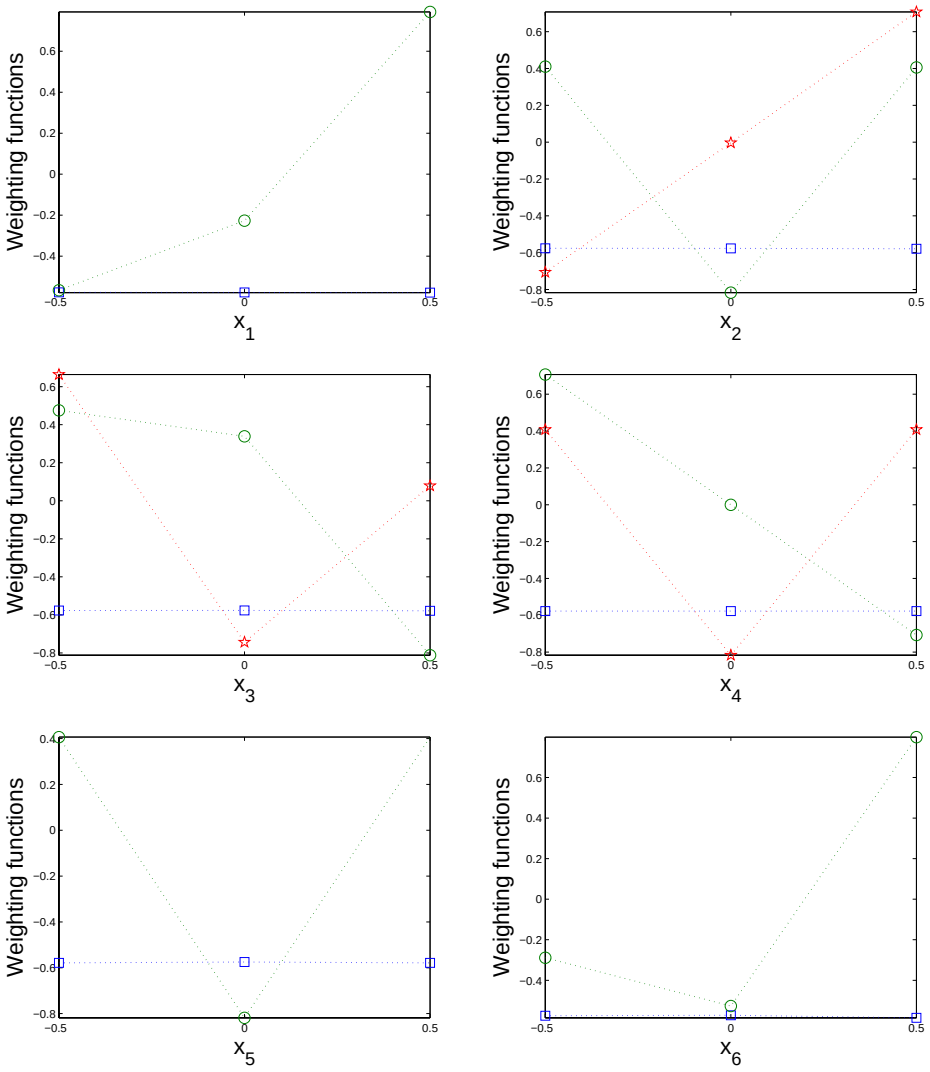

Figure 6.1: Discretized weighting functions over Θ

discretization grid. Here we have to consider two constraints. The first one is that the HOSVD (in MATLAB) can be executed when the number of grid-lines per dimension are about 3 in the present 6-dimensional case. The second restriction is that the computational load of LMI solver (to be executed in the case of controller design) explodes easily. Therefore, we restrict the computation of the TP model transformation to yield 3 weighting functions per dimension, that means maximum $3^6 = 729$ LTI systems. Thus, we define discretization grid Θ as $\forall n : H_n = 3$. The discretization results in $\mathcal{S}^{D,\Theta}$ whose number of elements is only $36 \cdot 3^6$. Executing HOSVD, we find that the rank of $\mathcal{S}^{D,\Theta}$ along each of its dimensions is 2, 3, 3, 3, 2, 2, respectively. Hence, the sizes of the resulting $\mathbf{U}_n$, $n = 1 \dots N$ are: 3×2, 3×3, 3×3, 3×3, 3×2 and 3×2. The size of $\mathcal{S}$ is $2 \times 3 \times 3 \times 3 \times 2 \times 2 \times 6 \times 6$. Figure 6.1 depicts the values of the columns of $\mathbf{U}_n$ (assigned to the discretization grid-lines) as the points of the discretized weighting functions. For easier visualization, we connected the points by straight line segments.

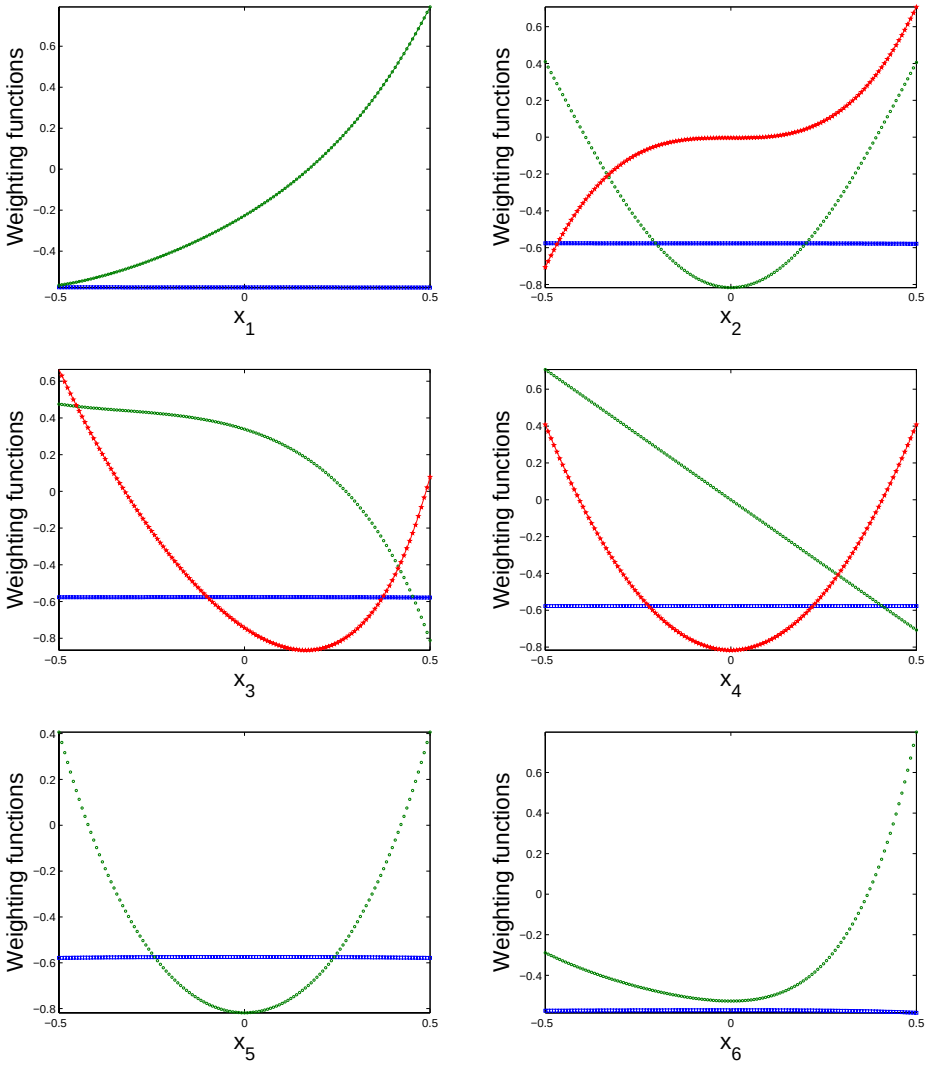

Figure 6.2: Discretized weighting functions over Γ

For Step 3 we define the dense grid Γ with 129 grid-lines for each dimension. The resulting weighting functions, discretized over 129 points, are depicted on Figure 6.2.

(If we execute Step 5 directly here then we can generate the continuous weighting functions and numerically check the error between the original and the resulting TP model over huge number of points. We find that the error, resulting from the numerical computation of HOSVD, is in the range of 10^{-13}. In conclusion we can say that the resulting TP model is exact and it includes a minimum number of 216 LTI systems.) As a next step we execute Step 4 and generate SN, NN, and NO type weighting functions as in the case of the original TP model transformation. By Step 5 we can generate the continuous weighting functions see Figure 6.3.

We can conclude from the above example that the original TP model transformation cannot be applied in case of this 6-dimensional problem because of its heavy computational load. The modified TP model transformation, however, can be executed on a regular PC and in the present example it yields an exact TP model.

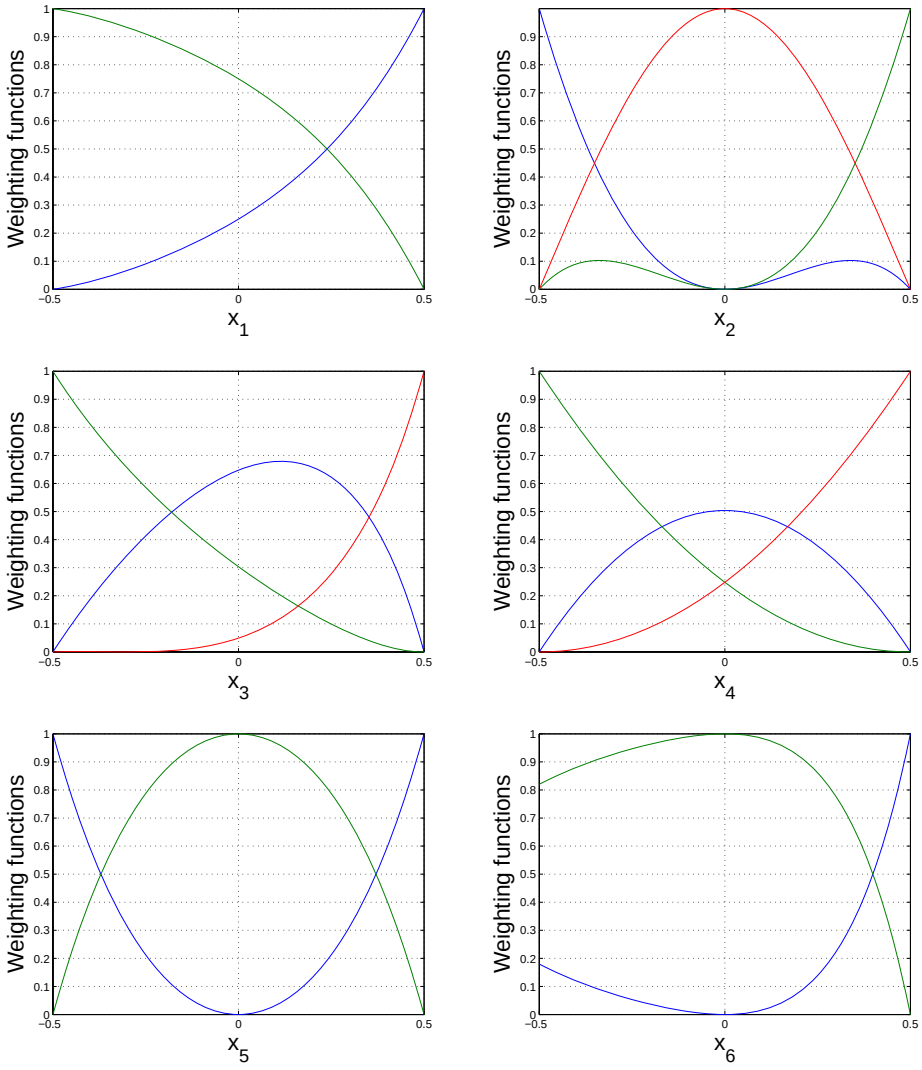

Figure 6.3: Continuous weighting functions over Γ

6.2 Avoiding the computation of constant elements in the LPV model

This approach deals with the matrix $\mathbf{S}(\mathbf{p}(t))$ and based on the fact that the constant, linear and nonlinear elements behave in different ways, have different effects, and have different computational cost consequences on the result of the TP model transformation. The TP model form of the constant elements can be determined without any computation, thus by separating the constant from linear, nonlinear elements and applying the TP model transformation for each set, the computational load can be reduced. A further refinement of this concept is when we separate the non-constant elements based on the degree of nonlinearity and degree of dependence on $\mathbf{p}(t)$.

6.2.1 The algorithm of the modified TP model transformation

The TP model transformation is capable of transforming an LPV model defined as (3.1) into TP model form (3.3):

$$\mathbf{S}(\mathbf{p}(t)) = \mathcal{S} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n(p_n(t)), \tag{6.6}$$

As the elements of the matrix $\mathbf{S}(\mathbf{p}(t))$ are independent, the TP model transformation can be performed for each element individually, and then we can write for an arbitrary (k,l) element of the matrix $\mathbf{S}(\mathbf{p}(t))$

$$\mathbf{S}_{k,l}(\mathbf{p}(t)) = \mathcal{S}^{k,l} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n^{k,l}(p_n(t)), \tag{6.7}$$

where $\mathcal{S}^{k,l}$ is an N-dimensional subtensor of $\mathcal{S}$ that is selected by fixating the last two coordinates of the $N+2$-dimensional $\mathcal{S}$ to (k,l), and $\mathbf{w}_n^{k,l}(p_n(t)), n = 1 \dots N$ is the corresponding weighting function set. The reason for executing the TP model transformation for the whole matrix $\mathbf{S}(\mathbf{p}(t))$ and not for each element separately is that the resulting weighting functions must be identical for all the elements as the transformation is trying to determine a solution.

However, if an arbitrary element $s_{k,l}(\mathbf{p}(t))$ is constant, namely it does not depend on any element of $\mathbf{p}(t)$, then for any arbitrary set of weighting functions $\mathbf{w}_n^{\text{arb}}(p_n(t)), n = 1 \dots N$ (the abbreviation "arb" means arbitrary) that is Sum-Normalized (SN) and Non-Negative (NN), namely satisfying the conditions given by Definition 5.1, we can write

$$\mathbf{S}_{k,l}(\mathbf{p}(t)) = \mathcal{S}^{s_{k,l}} \overset{N}{\underset{n=1}{\otimes}} \mathbf{w}_n^{\text{arb}}(p_n(t)), \tag{6.8}$$

where $\mathcal{S}^{s_{k,l}}$ is such an N-dimensional subtensor of $\mathcal{S}$ that is selected by setting the last two coordinates of the $N+2$-dimensional $\mathcal{S}$ to (k,l) and all values equal to $s_{k,l}$.

It is clear that the TP model transformation, applied to a constant element, is reduced to the simple construction of a subtensor, and the creation of a set of weighting functions that satisfies the SN and NN criteria. This set can be any weighting function set, for example, the weighting functions of a nonlinear element. So, for the constant elements the subtensors can be constructed in a straightforward way. In order to avoid the non-identical sets of weighting functions, the TP model transformation is performed on all linear and nonlinear elements simultaneously. The resulting weighting functions are appropriate for all elements (constant,

linear, and nonlinear) of the matrix $\mathbf{S}(\mathbf{p}(t))$. So, by doing the separation of the elements of the $\mathbf{S}(\mathbf{p}(t))$, performing the TP model transformation only on the non-constant set, which is computation intensive, and applying the resulting weighting functions to the constant set, the computational load of the TP model transformation can be reduced. The efficiency of this computation load relaxation is greatly depends on the ratio of constant and non-constant elements, the higher the number of constant elements the better relaxation can be achieved.

6.2.2 Evaluation of the calculation complexity reduction

In this subsection we compare the computational complexity of the original and the modified TP model transformation. Since the computation problem is caused by the computational explosion of the HOSVD, we focus attention on the number of elements in $\mathcal{S}^D$ in both cases.

We denote the computational load of SVD of a matrix with R number of elements by $C_{\mathrm{SVD}}(R)$. The original TP model transformation executes SVD N times on a matrix with R number of elements:

$$C_{\mathrm{original}} = N \times C_{\mathrm{SVD}}(O \times I \times \prod_{n=1}^{N} M_n)$$

The modified TP model transformation executes SVD N times as well, however the number of the elements of the matrix whereupon SVD is executed could be less than in the case of the original TP model transformation, because the SVD is executed only on the non-constant element of the system matrix. The modified TP model transformation executes HOSVD on the same grid density $\prod_{n=1}^{N} M_n$, but the system matrix is reduced to a vector containing only the non-constant element. Denote the number of non-constant elements by S_{nc}. The computation of the modified algorithm is

$$C_{\mathrm{modified}} = N \times C_{\mathrm{SVD}}(S_{\mathrm{nc}} \times \prod_{n=1}^{N} M_n)$$

The algorithm offers a linear computational complexity reduction.

6.2.3 Numerical example

Consider the following dynamic model:

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t))\mathbf{x}(t),$$

where

$$\mathbf{S}(\mathbf{p}(t)) = \begin{bmatrix} s_{1,1} & 2 & 2 & 4 & s_{1,5} & 8 \\ 2 & s_{2,2} & 3 & 1 & 4 & 2 \\ 2 & s_{3,2} & -2 & s_{3,4} & 7 & 3 \\ s_{4,1} & 3 & 8 & s_{4,4} & 9 & 3 \\ 1 & 2 & 0.5 & 4 & 5 & 0.2 \\ 0.6 & 2 & 7 & 4 & 5 & s_{6,6} \end{bmatrix} \tag{6.9}$$

$$\begin{aligned} s_{1,1} &= x_1 + x_1^2 + x_1^3 + x_1^4 \\ s_{1,5} &= 3 + x_1^2 + x_1^4 \\ s_{2,2} &= 3 + x_2^3 \\ s_{3,2} &= \frac{\sin(x_2)}{\cos(x_2)^2} \\ s_{3,4} &= 3\cos(x_2)\sin(x_2)^2 \\ s_{4,1} &= \cos(x_2) \\ s_{4,4} &= 1 + x_1^3 \\ s_{6,6} &= \cos(x_2) \end{aligned}$$

Observe that the above system matrix $\mathbf{S}(\mathbf{p}(t))$ nonlinearly depends on the first two elements of the state vector $\mathbf{x}(t) \in \mathbb{R}^6$. This means that $N = 2$ and $\mathbf{p}(t) = \{x_1(t), x_2(t)\}$. Let the transformation space be $\Omega : [a_1, b_1] \times [a_2, b_2]$, where $\forall n : a_n = -0.5$ and $\forall n : b_n = -a_n$. First we apply the original TP model transformation and we define 137 grid lines on each dimension for discretization. The discretization results in $\mathcal{S}^D$ whose number of elements is $36 \cdot 137^2$. However, if we apply the proposed modification, the discretization can be done only on the non-constant elements. In this case, we put all the non-constant elements to a vector $\mathbf{S}^{\text{TP}}(\mathbf{p}(t))$ like

$$\mathbf{S}^{\text{TP}}(\mathbf{p}(t)) = \begin{bmatrix} s_{1,1} \\ s_{1,5} \\ s_{2,2} \\ s_{3,2} \\ s_{3,4} \\ s_{4,1} \\ s_{4,4} \\ s_{6,6} \end{bmatrix} \tag{6.10}$$

and then we execute the TP model transformation on this reduced set. Thus the size of $\mathcal{S}^D$ becomes $8 \cdot 137^2$ which is a 78% reduction.

The results of the TP model transformation show that the example model can be exactly given as a combination of $4 \times 5 = 20$ LTI systems:

$$\mathbf{S}(\mathbf{p}(t)) = \mathcal{S} \bigotimes_{n=1}^{2} \mathbf{w}_n(p_n(t)), \tag{6.11}$$

The resulting weighting functions are depicted on Figure 6.4.

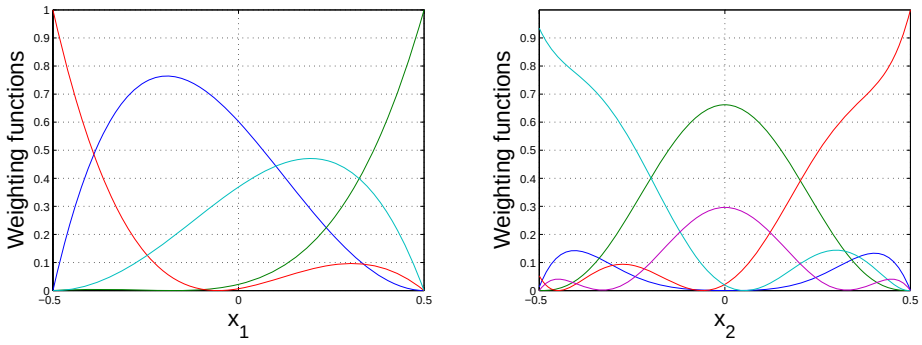

Figure 6.4: Weighting functions of x_1 and x_2

The size of the final tensor $\mathcal{S}$ is $4 \times 5 \times 6 \times 6$, namely it means it has $4 \times 5 = 20$ panes of 6×6. Each pane is constructed from the constant elements of $\mathbf{S}(\mathbf{p})$ and from the $4 \times 5 \times 8$-size tensor $\mathcal{S}^{\text{TP}}$ that contains the LTI values for the non-constant elements as

$$\mathcal{S}_{i,j,k,l} = \begin{cases} \mathbf{S}(\mathbf{p}(t))_{k,l} & \text{if} \quad \mathbf{S}(\mathbf{p}(t))_{k,l} = \text{constant} \\ \mathcal{S}^{\text{TP}}_{i,j,q} & \text{if} \quad \mathbf{S}(\mathbf{p}(t))_{k,l} \neq \text{constant} \end{cases}, \tag{6.12}$$

where i, j define a given pane, and k, l define a given element on the pane, $\mathcal{S}_{i,j,k,l}$ is the element k, l of the pane i, j in the tensor, $\mathbf{S}(\mathbf{p}(t))_{k,l}$ is the element k, l of the matrix. Because of the limited space, all the 20 LTI systems cannot be given, but for an arbitrary pane i, j of $\mathcal{S}$ the result is:

$$\mathbf{S}_{i,j} = \begin{bmatrix} \mathcal{S}^{\text{TP}}_{i,j,1} & 2 & 2 & 4 & \mathcal{S}^{\text{TP}}_{i,j,2} & 8 \\ 2 & \mathcal{S}^{\text{TP}}_{i,j,3} & 3 & 1 & 4 & 2 \\ 2 & \mathcal{S}^{\text{TP}}_{i,j,4} & -2 & \mathcal{S}^{\text{TP}}_{i,j,5} & 7 & 3 \\ \mathcal{S}^{\text{TP}}_{i,j,6} & 3 & 8 & \mathcal{S}^{\text{TP}}_{i,j,7} & 9 & 3 \\ 1 & 2 & 0.5 & 4 & 5 & 0.2 \\ 0.6 & 2 & 7 & 4 & 5 & \mathcal{S}^{\text{TP}}_{i,j,8} \end{bmatrix} \tag{6.13}$$

We can conclude form the above example that the modified TP model transformation gives a significant reduction on the computational load. However we must note that the degree of its reduction capability depends on the structure of the system matrix.

Part III

Applications

Chapter 7

TORA System

7.1 Introduction to the TORA system

The study is conducted through a output and state-feedback control design for the Translational Oscillations with an Eccentric Rotational Proof Mass Actuator (TORA) system, which was originally studied as a simplified model of a dual-spin spacecraft with mass imbalance to investigate the resonance capture phenomenon [47, 70]. The same plant was later studied involving the rotational proof-mass actuator for feedback stabilization of translational motion [23, 85]. The TORA system is also considered as a fourth-order benchmark problem [22, 24, 89]. The *International Journal of Robust and Nonlinear Control* published a series of studies about the control issue of the TORA system in Volume 8 in 1998 [72].

7.1.1 Nomenclature

- M = mass of cart
- k = linear spring stiffness
- m = mass of the proof-mass actuator
- I = moment of inertia of the proof-mass actuator
- e = distance between the rotation point and the center of the proof mass
- N = control torque applied to the proof mass
- F = is the disturbance force on the cart
- q = translational position of the cart
- θ = angular position of the rotational proof mass

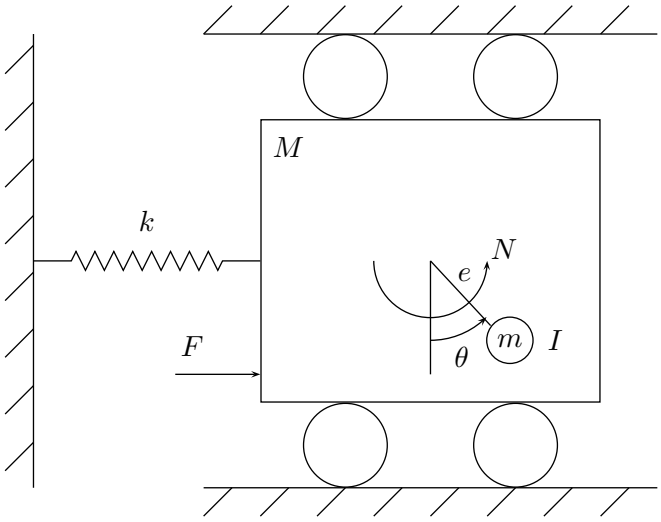

Figure 7.1: TORA system

7.1.2 Equations of motion

The TORA system is shown in Figure 7.1 with the notation defined above. The oscillation consists of a cart of mass M connected to a fixed wall by a linear spring of stiffness k. The cart is constrained to have one-dimensional travel in the horizontal plane. The rotating proof-mass actuator is attached to the cart. The control torque is applied to the proof mass. $\theta = 0^\circ$ is perpendicular to the motion of the cart, while $\theta = 90^\circ$ is aligned with the positive q direction. The equations of motion are given by [72]:

$$(M+m)\ddot{q} + kq = -me(\ddot{\theta}\cos\theta - \dot{\theta}^2\sin\theta) \tag{7.1}$$

$$(I+me^2)\ddot{\theta} = -me\ddot{q}\cos\theta + N \tag{7.2}$$

with the normalization [85]:

$$\xi \simeq \sqrt{\frac{M+m}{I+me^2}}q \quad \tau \simeq \sqrt{\frac{k}{M+m}}t \tag{7.3}$$

$$u \simeq \frac{M+m}{k(I+me^2)}N \tag{7.4}$$

the equations of motion become

$$\ddot{\xi} + \xi = \rho\left(\dot{\theta}^2\sin\theta - \ddot{\theta}\cos\theta\right) \tag{7.5}$$

$$\ddot{\theta} = -\rho\ddot{\xi}\cos\theta + u \tag{7.6}$$

where ξ is the normalized cart position, and u is the per unit control torque. τ is the normalized time whereupon the differentiation is understood. ρ is the coupling between the rotational and the translational motions:

$$\rho \simeq \frac{me}{\sqrt{(I+me^2)(M+m)}}. \tag{7.7}$$

The above equations can be given in the state-space model form

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))u(t), \tag{7.8}$$

$$\mathbf{y}(t) = \mathbf{c}(\mathbf{x}(t)),$$

where

$$\mathbf{f}(\mathbf{x}(t)) = \begin{pmatrix} x_2 \\ \frac{-x_1+\rho x_4^2 \sin(x_3)}{1-\rho^2\cos^2(x_3)} \\ x_4 \\ \frac{\rho\cos(x_3)(x_1-\rho x_4^2\sin(x_3))}{1-\rho^2\cos^2(x_3)} \end{pmatrix}, \quad \mathbf{g}(\mathbf{x}(t)) = \begin{pmatrix} 0 \\ \frac{-\rho\cos(x_3)}{1-\rho^2\cos^2(x_3)} \\ 0 \\ \frac{1}{1-\rho^2\cos^2(x_3)} \end{pmatrix}, \tag{7.9}$$

$$\mathbf{c}(\mathbf{x}(t)) = \begin{pmatrix} 0 & 0 & x_3 & 0 \\ 0 & 0 & 0 & x_4 \end{pmatrix}.$$

and $\mathbf{x}(t) = \begin{pmatrix} x_1(t) & x_2(t) & x_3(t) & x_4(t) \end{pmatrix}^T = \begin{pmatrix} \xi & \dot{\xi} & \theta & \dot{\theta} \end{pmatrix}^T$. Let us write the above equation in the typical form of linear parameter-varying state-space model as

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} \quad \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t), \tag{7.10}$$

where system matrix $\mathbf{S}(\mathbf{p}(t))$ contains:

$$\mathbf{S}(\mathbf{p}(t)) = \begin{pmatrix} \mathbf{A}(\mathbf{p}(t)) & \mathbf{B}(\mathbf{p}(t)) \end{pmatrix}$$

and $\mathbf{p}(t) = \begin{pmatrix} x_3(t) & x_4(t) \end{pmatrix} \in \Omega$ is time-varying 2nd-order parameter vector, thus

$$\mathbf{A}(x_3(t), x_4(t)) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{1}{1-\rho^2\cos^2(x_3)} & 0 & 0 & \frac{\rho x_4 \sin(x_3)}{1-\rho^2\cos^2(x_3)} \\ 0 & 0 & 0 & 1 \\ \frac{\rho\cos(x_3)}{1-\rho^2\cos^2(x_3)} & 0 & 0 & \frac{-x_4\rho^2\cos(x_3)\sin(x_3)}{1-\rho^2\cos^2(x_3)} \end{pmatrix} \tag{7.11}$$

$$\mathbf{B}(x_3(t)) = g(x(t)) \quad \mathbf{C} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The laboratory version of the TORA system is described in [24]. The nominal configuration of this version is given in Table 7.1.

7.2 HOSVD-based canonical form of the TORA system

We execute the TP model transformation on the LPV model (7.10) of the TORA. As a first step of the TP model transformation we have to define the transformation space Ω. If we see the simulations in the papers of special issue [72] and [31, 77, 79] we find that θ is always smaller than $0.85\,\mathrm{rad}$, and according to the maximum allowed torque $u = 0.1\,\mathrm{N}$ the system would not achieve larger $\dot{\theta}$ than $0.5\,\mathrm{rad}$, but could have a little overshoot, see [72, page 392] and control specifications section of this paper. Therefore, we define the transformation space as $\Omega = [-a, a] \times [-a, a]$ ($x_3(t) \in [-a, a]$ and $x_4(t) \in [-a, a]$), where $a = \frac{45}{180}\pi\,\mathrm{rad}$ (note

Table 7.1: Parameters of the TORA system

Description	Parameter	Value	Units
Cart mass	M	1.3608	kg
Arm mass	m	0.096	kg
Arm eccentricity	e	0.0592	m
Arm inertia	I	0.0002175	kg m^2
Spring stiffness	k	186.3	N/m
Coupling parameter	ρ	0.200	—

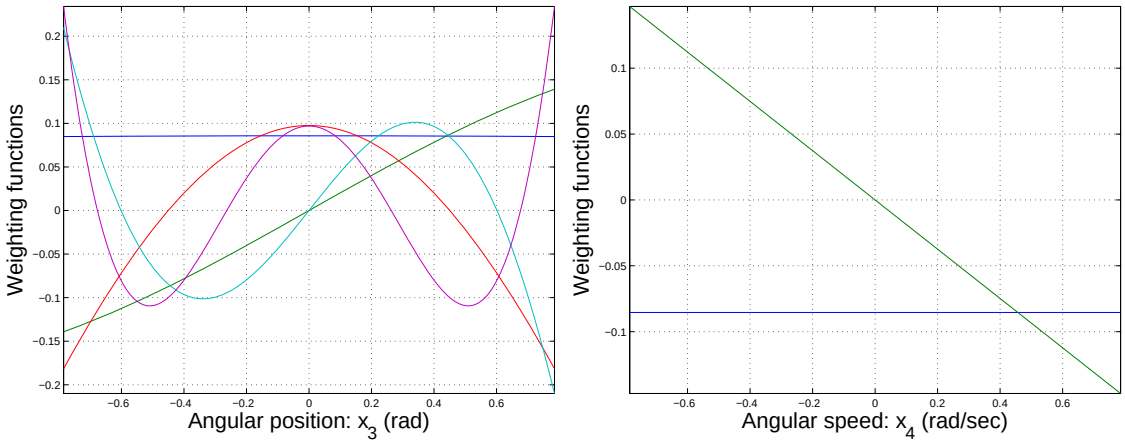

Figure 7.2: Weighting functions of the HOSVD-based canonical TP model of $x_3(t)$ and $x_4(t)$

that these intervals can be arbitrarily defined). The TP model transformation starts with the discretization over a rectangular grid. Let the density of the discretization grid be 137×137 on $(x_3(t) \in [-a, a]) \times (x_4(t) \in [-a, a])$. The result of the TP model transformation shows that TORA system can be exactly given in the HOSVD-based canonical polytopic model form with minimum $5 \times 2 = 10$ LTI vertex models

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{5} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (7.12)$$

The weighting functions are depicted in Figure 7.2.

7.2.1 Different finite element TP model forms of TORA system

In a mathematical point of view the TP model resulted in the previous section for TORA system is correct, however other TP model forms can be more advantageous in applications. These TP models are introduced in the followings.

TP MODEL 1 In order to have convex TP model to which the LMI control design conditions can be applied, let us generate convex TP model (SN and NN type weighting functions) by the TP model transformation which satisfies Definition 5.1. The results are depicted in Figure 7.3.

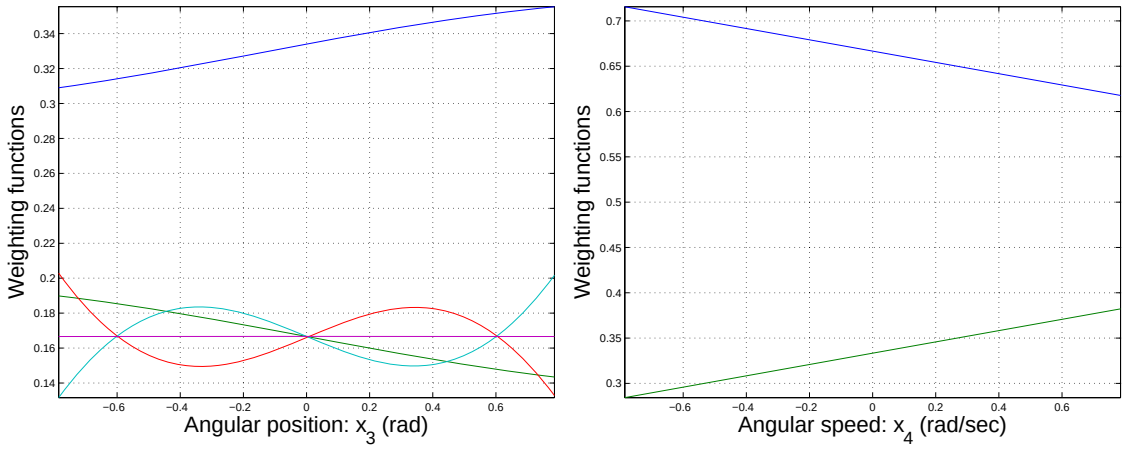

Figure 7.3: Weighting functions of TP model 1 for $x_3(t)$ and $x_4(t)$

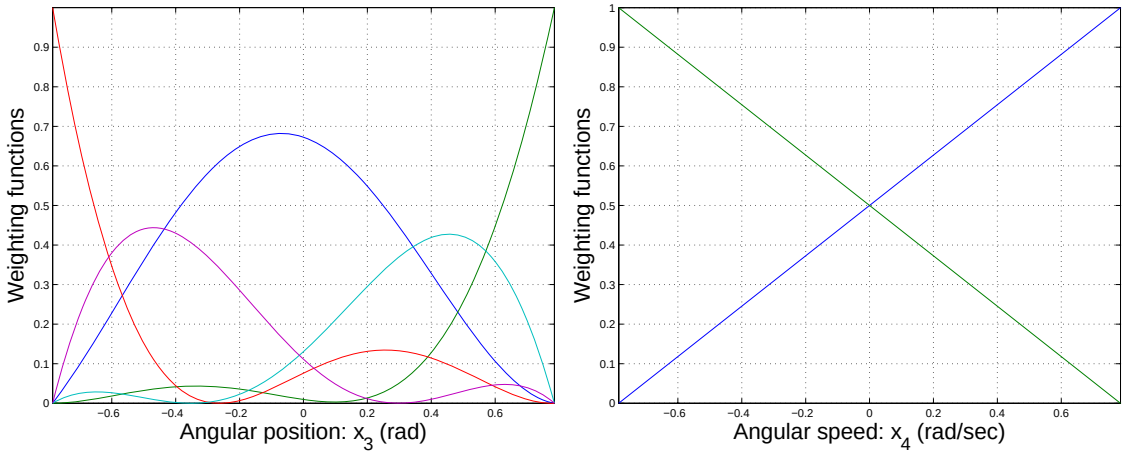

Figure 7.4: Weighting functions of TP model 2 for $x_3(t)$ and $x_4(t)$

TP MODEL 2 In many cases the convexity of the TP model is not enough, the further LMI controller design is not feasible. In order to relax the feasibility of the LMI conditions, let us define the tight convex hull of the LPV models via generating close to NO type weighting functions by the TP model transformation, see Figure 7.4.

The corresponding LTI system matrices of this TP model is also given, as the further controller design is based on it.

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{5} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right).$$

The LTI system matrices of the polytopic model are

$$\mathbf{A}_{1,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0493 & 0 & 0 & 0.0094 \\ 0 & 0 & 0 & 1.0000 \\ 0.2288 & 0 & 0 & -0.0010 \end{pmatrix} \quad \mathbf{B}_{1,1} = \begin{pmatrix} 0 \\ -0.2288 \\ 0 \\ 1.0493 \end{pmatrix}$$

$$\mathbf{A}_{1,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0493 & 0 & 0 & -0.0094 \\ 0 & 0 & 0 & 1.0000 \\ 0.2288 & 0 & 0 & 0.0010 \end{pmatrix} \quad \mathbf{B}_{1,2} = \begin{pmatrix} 0 \\ -0.2288 \\ 0 \\ 1.0493 \end{pmatrix}$$

$$\mathbf{A}_{2,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0204 & 0 & 0 & 0.1133 \\ 0 & 0 & 0 & 1.0000 \\ 0.1443 & 0 & 0 & -0.0160 \end{pmatrix} \quad \mathbf{B}_{2,1} = \begin{pmatrix} 0 \\ -0.1443 \\ 0 \\ 1.0204 \end{pmatrix}$$

$$\mathbf{A}_{2,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0204 & 0 & 0 & -0.1133 \\ 0 & 0 & 0 & 1.0000 \\ 0.1443 & 0 & 0 & 0.0160 \end{pmatrix} \quad \mathbf{B}_{2,2} = \begin{pmatrix} 0 \\ -0.1443 \\ 0 \\ 1.0204 \end{pmatrix}$$

$$\mathbf{A}_{3,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0204 & 0 & 0 & -0.1133 \\ 0 & 0 & 0 & 1.0000 \\ 0.1443 & 0 & 0 & 0.0160 \end{pmatrix} \quad \mathbf{B}_{3,1} = \begin{pmatrix} 0 \\ -0.1443 \\ 0 \\ 1.0204 \end{pmatrix}$$

$$\mathbf{A}_{3,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0204 & 0 & 0 & 0.1133 \\ 0 & 0 & 0 & 1.0000 \\ 0.1443 & 0 & 0 & -0.0160 \end{pmatrix} \quad \mathbf{B}_{3,2} = \begin{pmatrix} 0 \\ -0.1443 \\ 0 \\ 1.0204 \end{pmatrix}$$

$$\mathbf{A}_{4,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0328 & 0 & 0 & 0.1439 \\ 0 & 0 & 0 & 1.0000 \\ 0.1884 & 0 & 0 & -0.0272 \end{pmatrix} \quad \mathbf{B}_{4,1} = \begin{pmatrix} 0 \\ -0.1884 \\ 0 \\ 1.0328 \end{pmatrix}$$

$$\mathbf{A}_{4,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0328 & 0 & 0 & -0.1439 \\ 0 & 0 & 0 & 1.0000 \\ 0.1884 & 0 & 0 & 0.0272 \end{pmatrix} \quad \mathbf{B}_{4,2} = \begin{pmatrix} 0 \\ -0.1884 \\ 0 \\ 1.0328 \end{pmatrix}$$

$$\mathbf{A}_{5,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0222 & 0 & 0 & -0.1578 \\ 0 & 0 & 0 & 1.0000 \\ 0.1572 & 0 & 0 & 0.0282 \end{pmatrix} \quad \mathbf{B}_{5,1} = \begin{pmatrix} 0 \\ -0.1572 \\ 0 \\ 1.0222 \end{pmatrix}$$

$$\mathbf{A}_{5,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ -1.0222 & 0 & 0 & 0.1578 \\ 0 & 0 & 0 & 1.0000 \\ 0.1572 & 0 & 0 & -0.0282 \end{pmatrix} \quad \mathbf{B}_{5,2} = \begin{pmatrix} 0 \\ -0.1572 \\ 0 \\ 1.0222 \end{pmatrix}$$

The LMI stability theorems phrased under the PDC framework based controller design use linear indexing for the LTI systems. The multidimensional indexing of the LTI systems in the polytopic model can be linearized. For demonstrational purposes and better understanding Figure 7.5 and 7.6 show the weighting functions of TP model 2 in a 2nd-order form.

TP MODEL 3 Let us define further types of the weighting functions and define their INO–RNO type, see Figure 7.7. This INO–RNO type convex hull may relax the observer design in the output feedback control problem, see paper [6].

The weighting functions can be derived analytically in some special cases, but as the model become more complex, the analytical derivations needs more expertise and it can

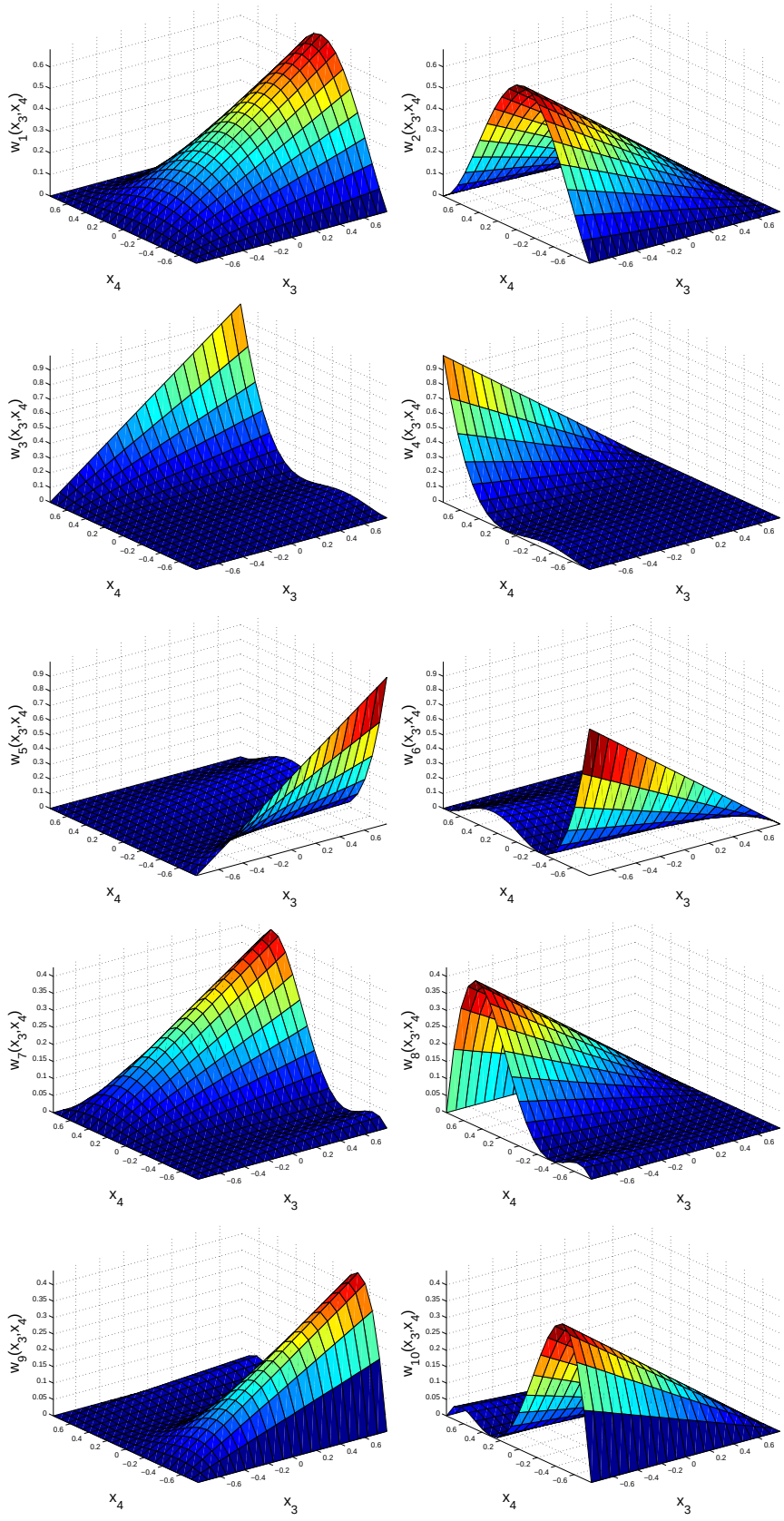

Figure 7.5: 2nd-order weighting functions of TP model 2

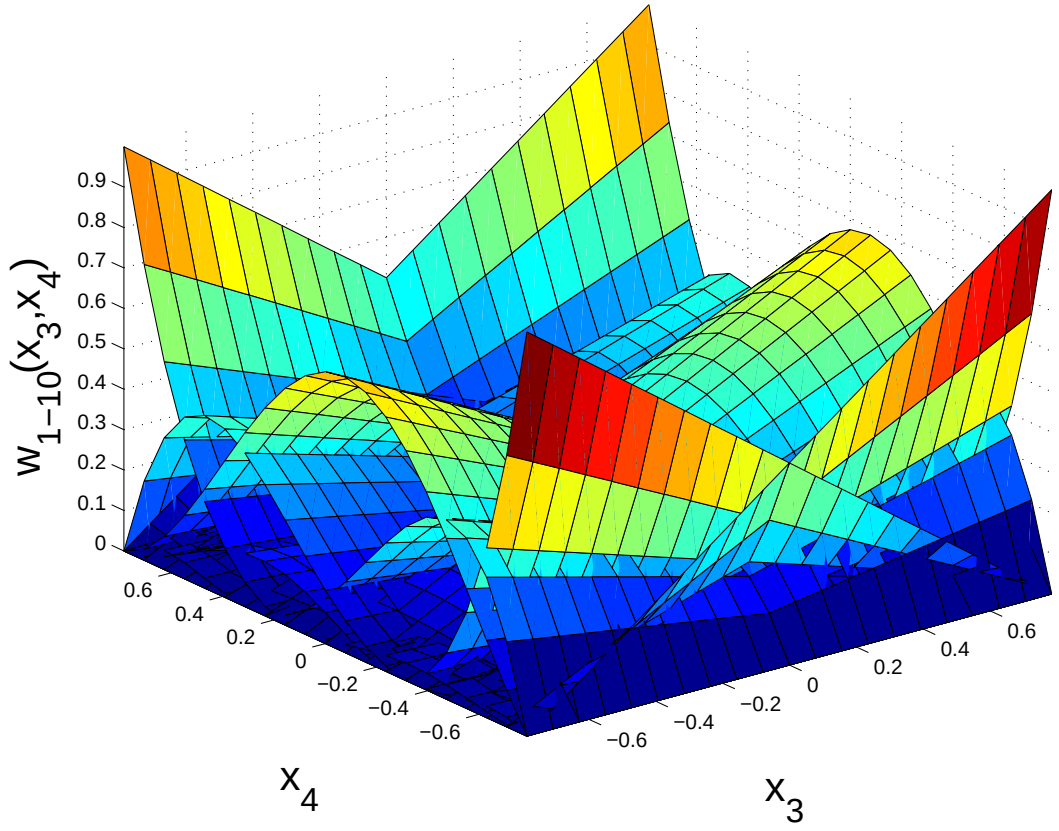

Figure 7.6: 2nd-order weighting functions of TP model 2 in a common coordinate system

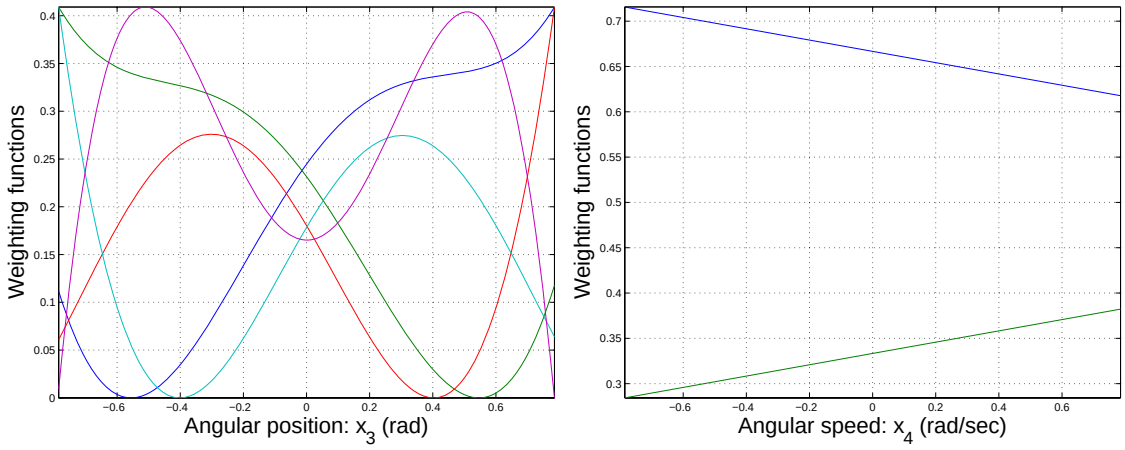

Figure 7.7: Weighting functions of TP model 3 for $x_3(t)$ and $x_4(t)$

easily be a very complex and hard task. Moreover, the analytical derivations of the tight convex hull or INO–RNO type weighting functions need the analytical solution of the tight convex hull problem that is unavailable in general. Finding the minimal number of LTI systems or the orthogonal weighting functions or LTI systems could also be very complex via analytical derivations. However all these features can readily be guaranteed by the Higher Order Singular Value Decomposition (whose analytic derivation is also unsolved in general) executed as the core of the TP model transformation. The TP model transformation requires a few minutes on a regular personal computer and is not dependent on the actual analytical form of the given LPV model. If the model is changed we can simply execute the TP model transformation again.

7.2.2 Complexity relaxed TP models

During the TP model transformation based controller design procedure complexity issues can occur that can inhibit the derivation of the controller, or the complexity of the resulting controller is so high that it is impossible to handle in real world operation. The TP model transformation based control design framework offers trade-off techniques that help us to control the model complexity and approximation accuracy challenge.

Besides the exact TP model, we also generate complexity relaxed TP models of the TORA system. In later section for each TP model a corresponding controller is designed. We analyze how the complexity relaxation changes the behavior of the TP models, and influence the controller performances.

In the previous subsection several exact, finite element TP model of the TORA system has been introduced. In the followings we generate close to NO types weighting function, but in each TP model we reduce the number of weighting function, thus the complexity of the model.

As it has been already shown, the rank of the system matrix $\mathbf{S}(\mathbf{p}(t))$ in the dimension of $x_3(t)$ is 5, whilst in the dimension of $x_4(t)$ is 2. The nonzero singular values in each dimension is

$$\begin{aligned} \sigma_{1,1} &= 341.31 & \qquad \sigma_{2,1} &= 341.31 \\ \sigma_{1,2} &= 5.5948 & \qquad \sigma_{2,2} &= 5.5948 \\ \sigma_{1,3} &= 3.8334 \\ \sigma_{1,4} &= 0.085134 \\ \sigma_{1,5} &= 0.041615 \end{aligned}$$

TP MODEL 2b The complexity of the TP model can be reduced in the dimension of $x_3(t)$ by discarding some singular values. Note that in the dimension of $x_4(t)$ the reduction is not possible since convexity (Definition 5.1) requires at least two weighting functions. Hence, let us keep the four largest singular values of dimension $x_3(t)$. It results an approximation of the

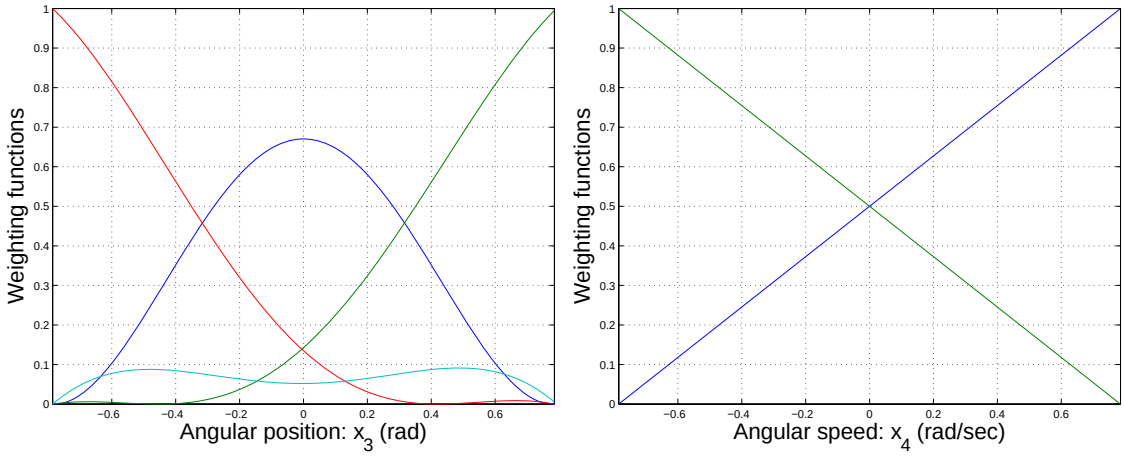

Figure 7.8: Close to NO type weighting functions of the reduced TP model of 8 LTI systems

TORA system that is composed of $4 \times 2 = 8$ LTI systems. Figure 7.8 shows the weighting functions of the tight convex hull of the reduced LPV model.

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{4} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (7.13)$$

Equation (3.13) gives only an upper bound for the approximation error that is calculated by the discarded singular values. In case of TP MODEL 2b it is $\sigma_{1,5} = 0.041615$. In order to measure the actual modeling approximation error by numerical checking, the difference of the analytical model and TP models, in terms of L_2 norm, were calculated over $10\,000$ random sample points. The numerical approximation error of TP MODEL 2b is 0.0023.

TP MODEL 2c In the dimension of $x_3(t)$ further reduction is possible. By discarding the two smallest singular values, namely $\sigma_{1,4}$ and $\sigma_{1,5}$, we get a more complexity relaxed TP model of the TORA system. The resulting system equation is

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{3} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (7.14)$$

In this case the upper bound of the approximation error is 0.126749, while the maximal numerical error is 0.0024. In Figure 7.9 the resulting weight functions are illustrated.

TP MODEL 2d By keeping only the two largest singular values in dimension $x_3(t)$, the most reduced TP model of the TORA system realizes. The system equation of this model is

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{2} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (7.15)$$

In this case the upper bound of the approximation error is 3.960149, while the maximal measured error is 0.21513. In Figure 7.10 the resulting weight functions are illustrated.

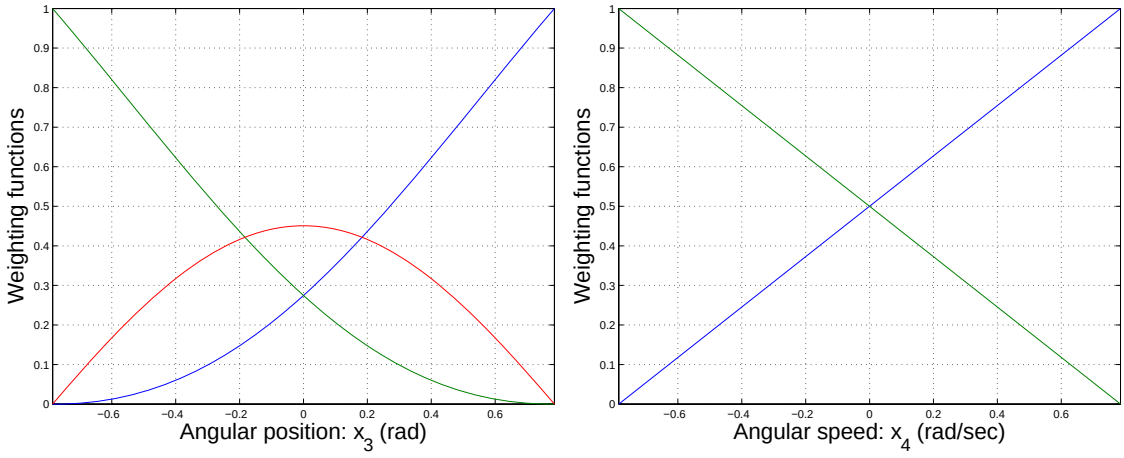

Figure 7.9: Close to NO type weighting functions of the reduced TP model of 6 LTI systems

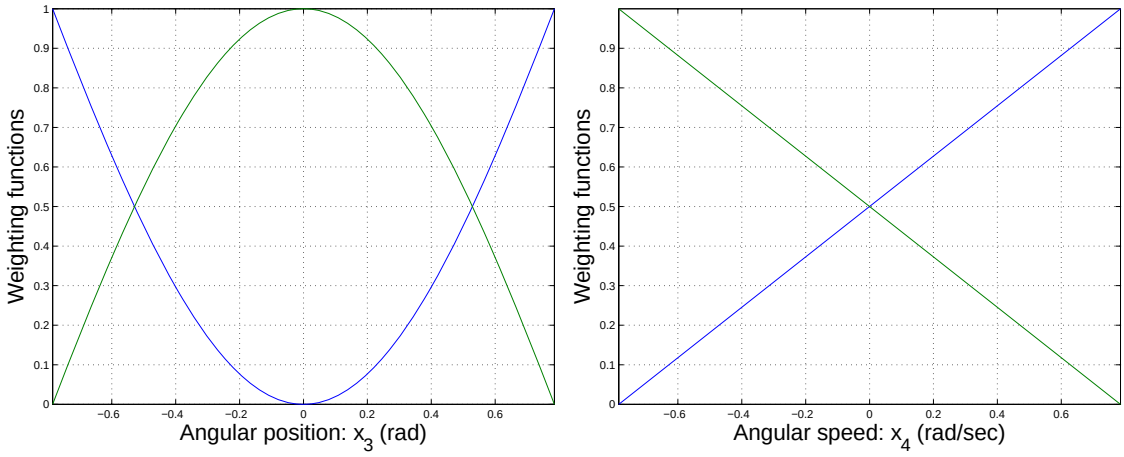

Figure 7.10: Close to NO type weighting functions of the reduced TP model of 4 LTI systems

Comparison of the resulting TP models The main conclusion of the comparison is the complexity of the model can be drastically reduced without causing unacceptable approximation error as the results, summarized in Table 7.2 on page 71, show. The second conclusion is that the estimated error bound from the singular values are much worse than the actual approximation error. As a matter of fact we have to be careful when selecting the non-exact approximation. In most cases, and in the case of TORA, the approximated model is suitable, but there are some systems, *e.g.* chaotic systems, when even slight changes can cause drastic differences. It is also worth noticing here that usually the error of typical identification techniques is in a larger range than the discarded singular values.

7.3 Multi-objective state-feedback controller design

7.3.1 Control specifications

According to the papers in the special issue [72] and papers [31,77,79] we summarize here the typical design specifications [72, page 309] of the benchmark problem of the TORA system:

Design a controller that satisfies the following criteria:

- The closed-loop system is stable (in this paper we aim at achieving asymptotic stability).
- The closed-loop system exhibits good settling behavior for a class of initial conditions.
- The physical configuration of the system necessities the constraint $|q| \leq 0.025\,\mathrm{m}$.
- The control value is limited by $N \leq 0.100\,\mathrm{Nm}$, although somewhat higher torques can be tolerated for short periods.

7.3.2 Asymptotic stability of the TORA system

This section derives controllers for different specifications by applying the TP model 2 and the LMI stability theorems. Having the solution of the LMIs, the feedback gains are computed by (7.18), and the control value is computed by (5.4). In the present case it is:

$$u(t) = -\left(\sum_{i=1}^{5}\sum_{j=1}^{2} w_{1,i}(x_3(t))w_{2,j}(x_4(t))\mathbf{F}_{i,j}\right)\mathbf{x}(t),$$

where

$$\mathbf{F}_{i,j} = \mathbf{F}_r, \quad \textit{and} \quad r = 2(i-1)+j$$

We compare the control results to various different alternative solutions [31,72,77,79]. An important difference from other controller design methods is that they are analytical solutions, while the present solution is automatically derived via numerical methods in a few minutes on a regular computer without analytical interaction. Further design specifications such as parameter uncertainty and robust control, etc. can be readily included in the design, if needed, by selecting other LMIs from the control literature. If the dynamic model is modified then the design process can be repeated in a few minutes unlike the analytical solutions which may lead to a hard work again even in case of a small modification.

CONTROLLER 0 In order to guarantee asymptotic stability, let us substitute the LTI systems of TP model 2 into the following LMI system. The derivation and the proof of the theorem is fully detailed in [80].

Theorem 7.1 (Asymptotic stability)**.** *Polytopic model (5.3) with control value (5.4) is asymptotically stable if there exist* $\mathbf{X} > 0$ *and* $\mathbf{M}_r$ *satisfying equations*

$$-\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} + \mathbf{M}_r^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_r > \mathbf{0} \tag{7.16}$$

for all r and

$$-\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} - \mathbf{X}\mathbf{A}_s^T - \mathbf{A}_s\mathbf{X} + \\ +\mathbf{M}_s^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_s + \mathbf{M}_r^T\mathbf{B}_s^T + \mathbf{B}_s\mathbf{M}_r \geq \mathbf{0}. \tag{7.17}$$

for $r < s \leq R$, except the pairs (r, s) such that $w_r(\mathbf{p}(t))w_s(\mathbf{p}(t)) = 0, \forall \mathbf{p}(t)$, and where the feedback gains are determined form the solutions $\mathbf{X}$ and $\mathbf{M}_r$ as

$$\mathbf{F}_r = \mathbf{M}_r\mathbf{X}^{-1}. \tag{7.18}$$

We find that the LMIs are feasible. The simulation result is depicted on Figure 7.11, where $\mathbf{x}(0) = \begin{pmatrix}0.023\,\mathrm{m} & 0 & 0 & 0\end{pmatrix}$. We can compare the control result to stable fuzzy control on Figures 1–4 of [79, page 318]. Note that the parameters on the figures of [79] are per unit and $\mathbf{x}(0) = \begin{pmatrix}0.0195\,\mathrm{m} & 0 & 0 & 0\end{pmatrix}$. Other difference is that $\varepsilon = 0.05$ (we used $\varepsilon = 0.2$). The stabilization time is about 100 per unit that is 8.84 s. If we use the same ε and initials then we can conclude that CONTROLLER 0 is slightly faster and uses smaller control value.

CONTROLLER 1 In order to achieve better response of the controller we may design decay rate control by the next theorem

Theorem 7.2 (Decay rate control)**.** *Assume the polytopic model (5.3) with controller (5.4). The largest lower bound on the decay rate by quadratic Lyapunov function is guaranteed by the solution of the following generalized eigenvalue minimizations problem (GEVP):*

$$\underset{X, M_1, \ldots, M_R}{\text{maximize}}\, \alpha \quad \text{subject to}$$

$$\begin{aligned} \mathbf{X} &> \mathbf{0}, \\ -\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} + \mathbf{M}_r^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_r - 2\alpha\mathbf{X} &> \mathbf{0}, \\ -\mathbf{X}\mathbf{A}_r^T - \mathbf{A}_r\mathbf{X} - \mathbf{X}\mathbf{A}_s^T - \mathbf{A}_s\mathbf{X} + \mathbf{M}_s^T\mathbf{B}_r^T + \mathbf{B}_r\mathbf{M}_s & \\ +\mathbf{M}_r^T\mathbf{B}_s^T + \mathbf{B}_s\mathbf{M}_r - 4\alpha\mathbf{X} &\geq \mathbf{0} \end{aligned}$$

for $r < s \leq R$, except the pairs (r, s) such that $w_r(\mathbf{p}(t))w_s(\mathbf{p}(t)) = 0, \forall \mathbf{p}(t)$, and where the feedback gains are determined form the solutions by (7.18).

The simulation result is depicted on Figure 7.12 with $\mathbf{x}(0) = \begin{pmatrix}0.023\,\mathrm{m} & 0 & 0 & 0\end{pmatrix}$. If we compare Figures 7.11 and 7.12 we can observe that the system is stabilized significantly faster, however the control value is considerably larger.

In the followings, we set constraints on the control value and the output to meet the control specifications listed in Section 7.3.1.

CONTROLLER 2 In order to satisfy the constraints defined in Section 7.3.1, the following LMIs are added to the previous ones.

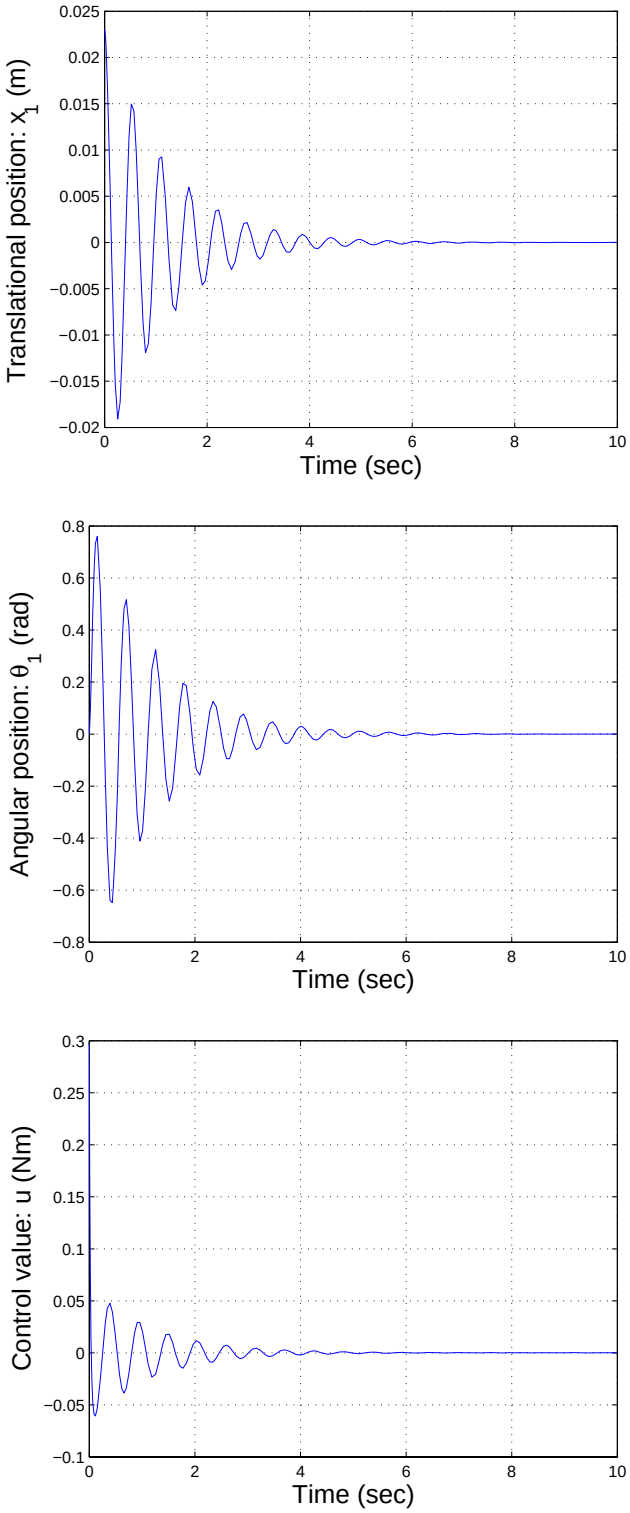

Figure 7.11: CONTROLLER 0: Asymptotic stability control design, $\mathbf{x}(0) = (0.023\,0\,0\,0)$

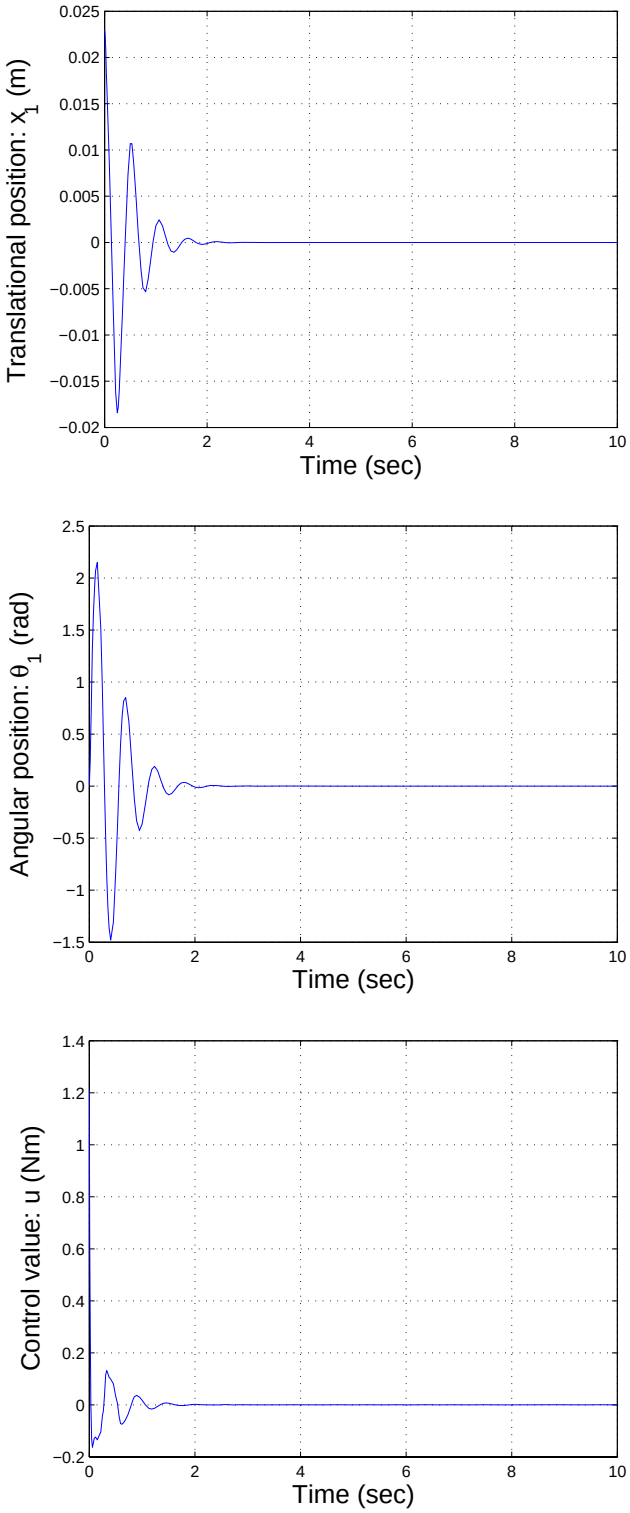

Figure 7.12: CONTROLLER 1: Decay rate control design, $\mathbf{x}(0) = (0.023\,0\,0\,0)$

Theorem 7.3 (Constraint on the control value)**.** *Assume that* $\|\mathbf{x}(0)\| \leq \phi$*, where* $\mathbf{x}(0)$ *is unknown, but the upper bound* ϕ *is known. The constraint* $\|\mathbf{u}(t)\|_2 \leq \mu$ *is enforced at all times* $t \geq 0$ *if the LMIs*

$$\begin{aligned} \phi^2\mathbf{I} &\leq \mathbf{X} \\ \begin{pmatrix} \mathbf{X} & \mathbf{M}_i^T \\ \mathbf{M}_i & \mu^2\mathbf{I} \end{pmatrix} &\geq \mathbf{0} \end{aligned}$$

hold.

Theorem 7.4 (Constraint on the output)**.** *Assume that* $\|\mathbf{x}(0)\| \leq \phi$*, where* $\mathbf{x}(0)$ *is unknown, but the upper bound* ϕ *is known. The constraint* $\|\mathbf{y}(t)\|_2 \leq \lambda$ *is enforced at all times* $t \geq 0$ *if the LMIs*

$$\begin{aligned} \phi^2\mathbf{I} &\leq \mathbf{X} \\ \begin{pmatrix} \mathbf{X} & \mathbf{X}\mathbf{C}_i^T \\ \mathbf{C}_i\mathbf{X} & \lambda^2\mathbf{I} \end{pmatrix} &\geq \mathbf{0} \end{aligned}$$

hold.

Thus we solve these LMIs for the constraints together with the LMIs of Theorem 7.1 to guarantee asymptotic stability. The simulation result is depicted on Figure 7.13 with $\mathbf{x}(0) = \begin{pmatrix} 0.023\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. Comparing Figures 7.11 and 7.13 we can conclude that the maximum control value of CONTROLLER 2 ($0.0802\,\mathrm{Nm}$) is significantly smaller than the control value of CONTROLLER 1 ($1.2113,\mathrm{Nm}$) and even of CONTROLLER 0 ($0.2968\,\mathrm{Nm}$), see Figure 7.11. This is obviously expected by solving the LMIs of Theorem 7.3 and 7.4. We can find simulation results of different controllers presented in the special issue [72].

Figure 2 of [72, page 324] shows the result of a simple linear controller (per unit parameters), where $\mathbf{x}(0) = \begin{pmatrix} 0.0195\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. The stabilization time is more than 40 per unit that is $3.1\,\mathrm{s}$. Having the same initials CONTROLLER 2 stabilizes the system in about $2.5\,\mathrm{s}$. The control value is not published in [72, page 324].

Figures 3 and 4 of [72, page 363] present control results of different controllers for initial vector $\mathbf{x}(0) = \begin{pmatrix} 0.03\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. Here the controllers are derived based on nonlinear passive control synthesis. In order to be comparable let us simulate the response of the CONTROLLER 2 for the same initials, see Figure 7.14. We can observe that the CONTROLLER 2 stabilizes the system in about $6\,\mathrm{s}$ while the stabilization time on Figures 3 and 4 of [72, page 363] is about $13\,\mathrm{s}$ and $20\,\mathrm{s}$ depending on the controller applied. The maximum control value of CONTROLLER 2 is $0.0802\,\mathrm{Nm}$ while on the Figures 3 and 4 on the page 363 are between $0.03\,\mathrm{Nm}$ and $0.05\,\mathrm{Nm}$ depending on the controller applied. For instance, the full order control system is stabilized in about $13\,\mathrm{s}$ with maximum $0.035\,\mathrm{Nm}$ control value. The conclusion is that the CONTROLLER 2 is significantly faster, but generates twice as large control value in the first moments.

Figures 2 and 3 on page 394 show the effectiveness of a measurement-scheduled control for $\mathbf{x}(0) = \begin{pmatrix} 0.023\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. CONTROLLER 2 on Figure 7.13 shows a slightly faster stabilization and applies slightly smaller maximum ($0.0802\,\mathrm{Nm}$) control value than the controller of Figures 2 and 3 on page 394. We can conclude that these control results are very similar.

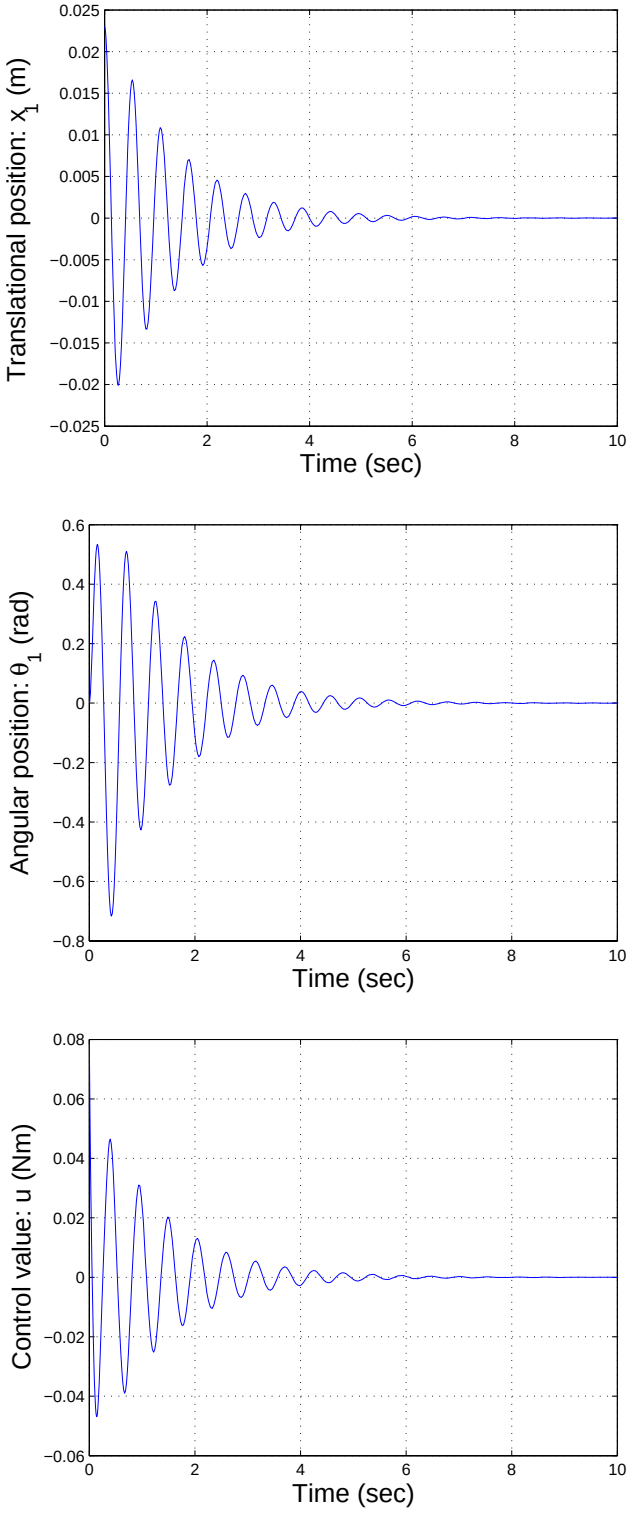

Figure 7.13: CONTROLLER 2: Asymptotic stability control design with constraints, $\mathbf{x}(0) = (0.023\,0\,0\,0)$

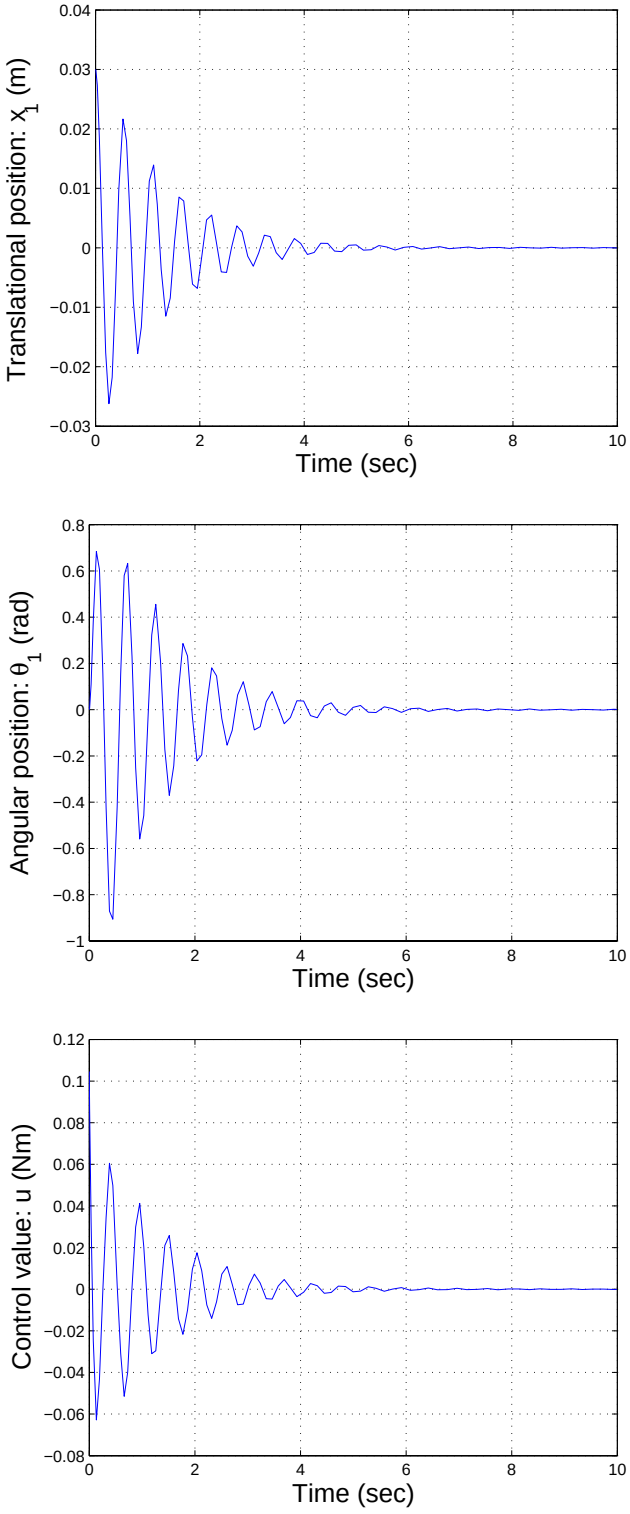

Figure 7.14: CONTROLLER 2: Asymptotic stability control design with constraints, $\mathbf{x}(0) = (0.03\,0\,0\,0)$

Figure 3 and 5 on page 412 present the control results of state-dependent Riccati Equation based optimal solutions for $\mathbf{x}(0) = \begin{pmatrix} 0.0058\,\text{m} & 0 & 0 & 0 \end{pmatrix}$. Note that the values on the figures are dimensionless. The stabilization time is about $1.9\,\text{s}$ and the maximum control value is $0.0531\,\text{Nm}$. CONTROLLER 2 stabilizes the system in about $3.3\,\text{s}$ with maximum $0.0802\,\text{Nm}$. Figures 2 and 3 of paper [31] shows the saturated passivity based control results. This simulation uses the same initials as our simulation shown on Figure 7.13. We can conclude that the results are very similar in the sense of settling time and the maximum of the control value.

7.3.3 Disturbance rejection performance

The disturbance rejection performance is also a measure to analyze the robustness of a controller. Some articles of Special issue [72] on the TORA system proposes different simulations and solutions for disturbance rejection.

For disturbance rejection performance the following extended TORA model is used. The notation is the same as used in Section 7.1. The extended state-space model form with disturbance is

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))u(t) + \mathbf{d}(\mathbf{x})w(t), \tag{7.19}$$
$$\mathbf{y} = \mathbf{c}(\mathbf{x}(t)),$$

where

$$\mathbf{f}(\mathbf{x}(t)) = \begin{pmatrix} x_2 \\ \frac{-x_1+\varepsilon x_4^2 \sin x_3}{1-\varepsilon^2 \cos^2 x_3} \\ x_4 \\ \frac{\varepsilon \cos x_3 (x_1-\varepsilon x_4^2 \sin x_3)}{1-\varepsilon^2 \cos^2 x_3} \end{pmatrix}, \quad \mathbf{g}(\mathbf{x}(t)) = \begin{pmatrix} 0 \\ \frac{-\varepsilon \cos x_3}{1-\varepsilon^2 \cos^2 x_3} \\ 0 \\ \frac{1}{1-\varepsilon^2 \cos^2 x_3} \end{pmatrix}, \quad \mathbf{d}(\mathbf{x}) = \begin{pmatrix} 0 \\ \frac{1}{1-\varepsilon^2 \cos^2 x_3} \\ 0 \\ \frac{-\varepsilon \cos x_3}{1-\varepsilon^2 \cos^2 x_3} \end{pmatrix} \tag{7.20}$$

$$\mathbf{c}(\mathbf{x}(t)) = \begin{pmatrix} x_1 & 0 & 0 & 0 \\ 0 & 0 & x_3 & 0 \end{pmatrix},$$

$\mathbf{x}(t) = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}^T = \begin{pmatrix} \xi & \dot{\xi} & \theta & \dot{\theta} \end{pmatrix}^T$ and $w(t)$ is the disturbance.

The disturbance rejection property of CONTROLLER 2 designed in Section 7.3.2 in introduced and compared to other results in the followings. Simulation of the disturbance rejection of the controller derived via measurement-scheduled based technique is presented on the Figure 6 of special issue [72, page 395]. In that simulation $w(t) = 1.8\sin(10t)$ is added to the system for all $t > 0$, where $\mathbf{x}(0) = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$. The results are shown in Figure 7.15. The system is stabilized in about $6\,\text{s}$ and about $1.7 \cdot 10^{-3}\,\text{m}$ amplitude remains. CONTROLLER 2 is also stabilized in about $6\,\text{s}$ for the same initials and disturbance, however the remaining amplitude is significantly smaller: $0.4 \cdot 10^{-3}\,\text{s}$.

Figure 13 of [72, page 419] shows the disturbance rejection of a controller design via state dependent Riccati Equation (SDRE) technique. The disturbance $w(t) = 0.01\sin(5t)$ is added to the system for all $t > 0$, where $\mathbf{x}(0) = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$. Note that this figure shows dimensionless values. For comparison Figure 7.16 shows the results of CONTROLLER 2 for the same initials and disturbance. When we compare Figure 7.16 to Figure 13 of [72, page 419] we can observe that the resulting performances are very similar.

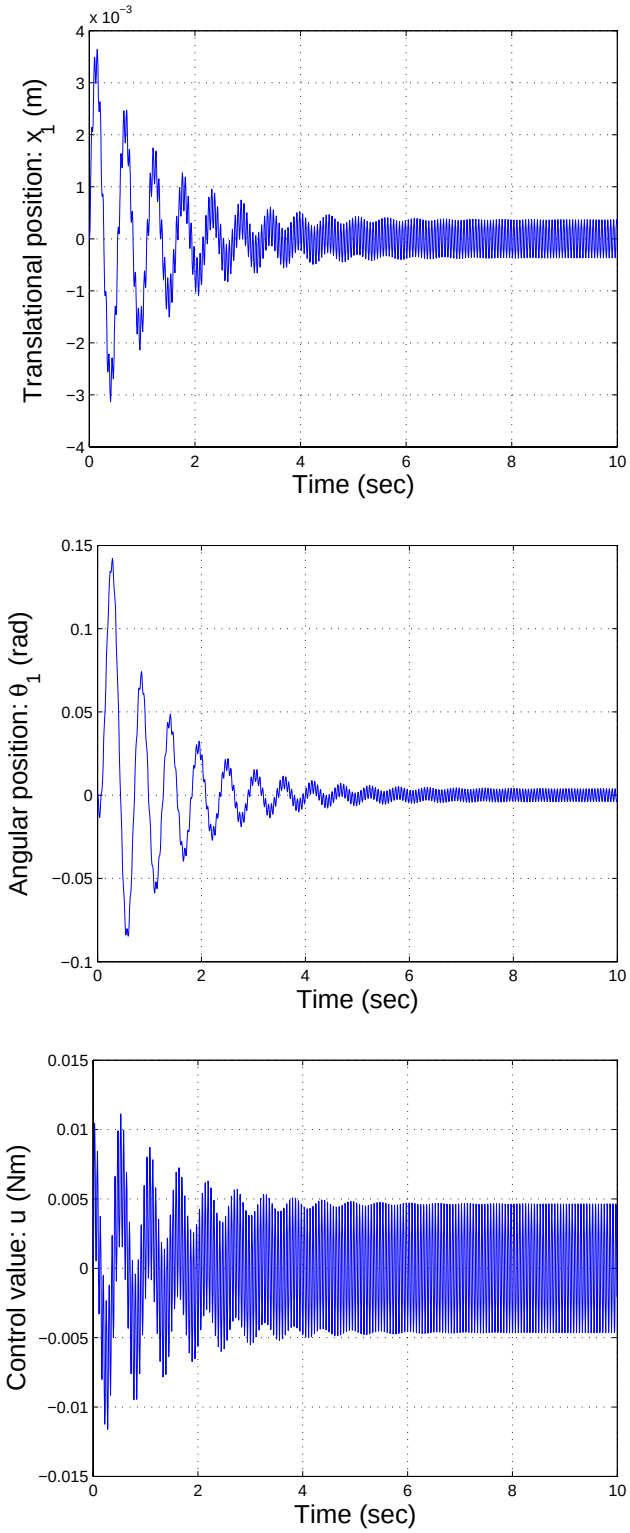

Figure 7.15: Controller 2. Asymptotic stability control design with constraints and disturbance $w(t) = 1.8\sin(10t)$, $\mathbf{x}(0) = (0\,0\,0\,0)$

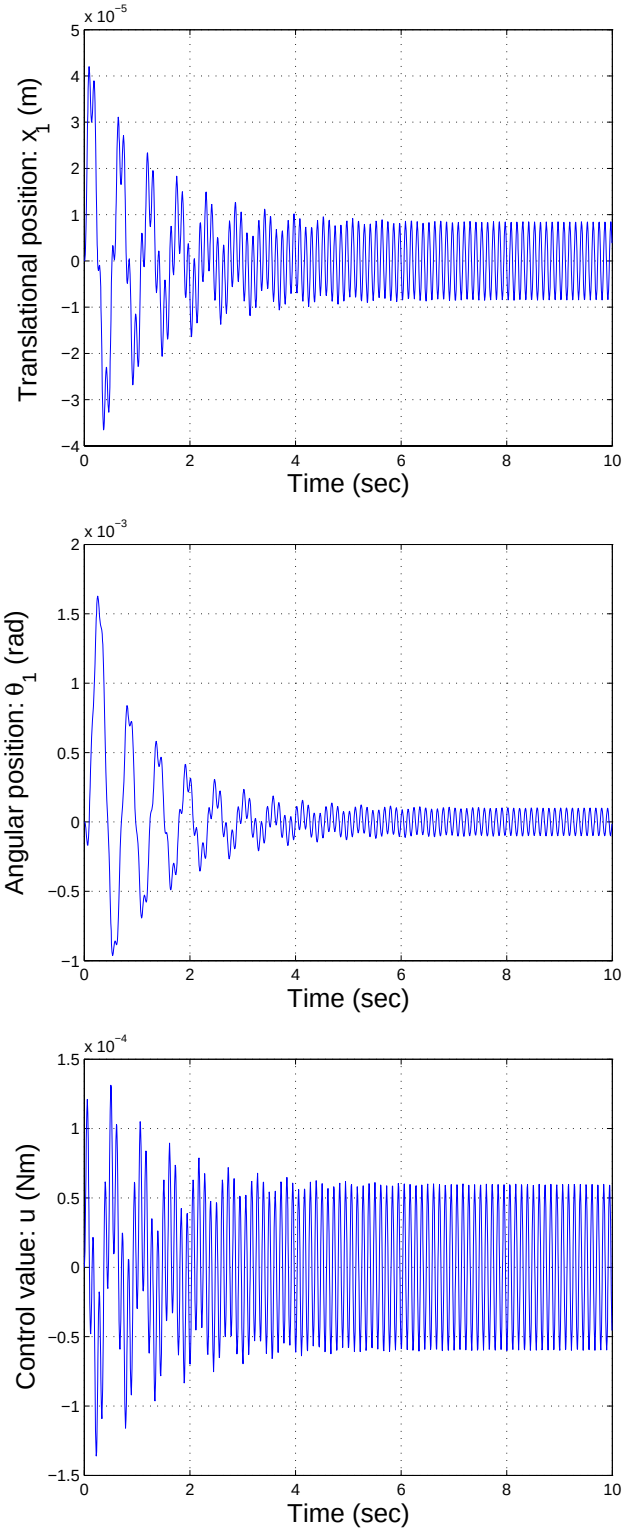

Figure 7.16: Controller 2. Asymptotic stability control design with constraints and disturbance $w(t) = 0.01\sin(5t)$, $\mathbf{x}(0) = (0\,0\,0\,0)$

7.3.4 Evaluation of complexity relaxed TP model based controller design

For the present evaluation of complexity and approximation accuracy challenge, we apply the control specification given in Section 7.3.1.

We compose a joint LMI system of Theorem 7.1 and Theorems 7.3 and 7.4 to guarantee the stability issues and constraints defined in the above control specification.

CONTROLLER 4 We consider this controller as the reference controller, and the response of the rest of the controllers are compared to this. The controller is designed for TP MODEL 2. This requires the solution of 67 LMI equations.

CONTROLLER 4b The feedback gains for the controller of TP MODEL 2b is obtained by the feasible solution of an LMI system containing 46 LMI equations.

CONTROLLER 4c The LMI system of TP MODEL 2c consists of 29 LMIs. The feedback gains of the controller, that is derived from the feasible solution of LMI system, is the following:

CONTROLLER 4d We applied the same LMI system to TP MODEL 2d that results a system of 16 LMI equations. The feedback gains of the controller is obtained from the feasible solution of the LMI system.

Simulation results and comparison of derived controllers In the simulation the system's initial configuration was $\mathbf{x}(0) = \begin{pmatrix} 0.023\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. The results of the four controllers are shown and are plotted together for better visualization in Figure 7.17. Figure 7.18 shows some parts of the simulation results magnified in order to highlight the differences. We can see on the figures, there are only slight differences between the responses of the different controllers, practically we can say that the results are identical despite of the applied reduction during the TP model transformation. The main reason behind this fact is that the strength of influence of the LTI models is proportional to the magnitude of the singular values. Therefore, the magnitude of differences between the designed controllers is also strongly related to the magnitude of the singular values. For illustration, we should analyze carefully the responses of the controller of the exact model (indicated with "CTRL 4" in the figures), and the controller CONTROLLER 4b. The difference is so small, because the difference of the exact TP model, TP MODEL 4 and the TP MODEL 4b from which the controllers were derived is also really small. The contribution of the neglected LTIs to the TP model is proportional to the ratios of the singular values, and $\sigma_{1,5}$ has only an effect of 0.012%. If we analyze the response of CONTROLLER 4c, we cannot see more significant difference, because the contribution of $\sigma_{1,4}$ is also around 0.025%, thus together with $\sigma_{1,5}$ it is still around 0.037% in total. A slightly significant change can be observer in the response of CONTROLLER 4d. In case the sum of the discarded singular values has an effect of about 1.16%, thus a bigger results in the response of the controller.

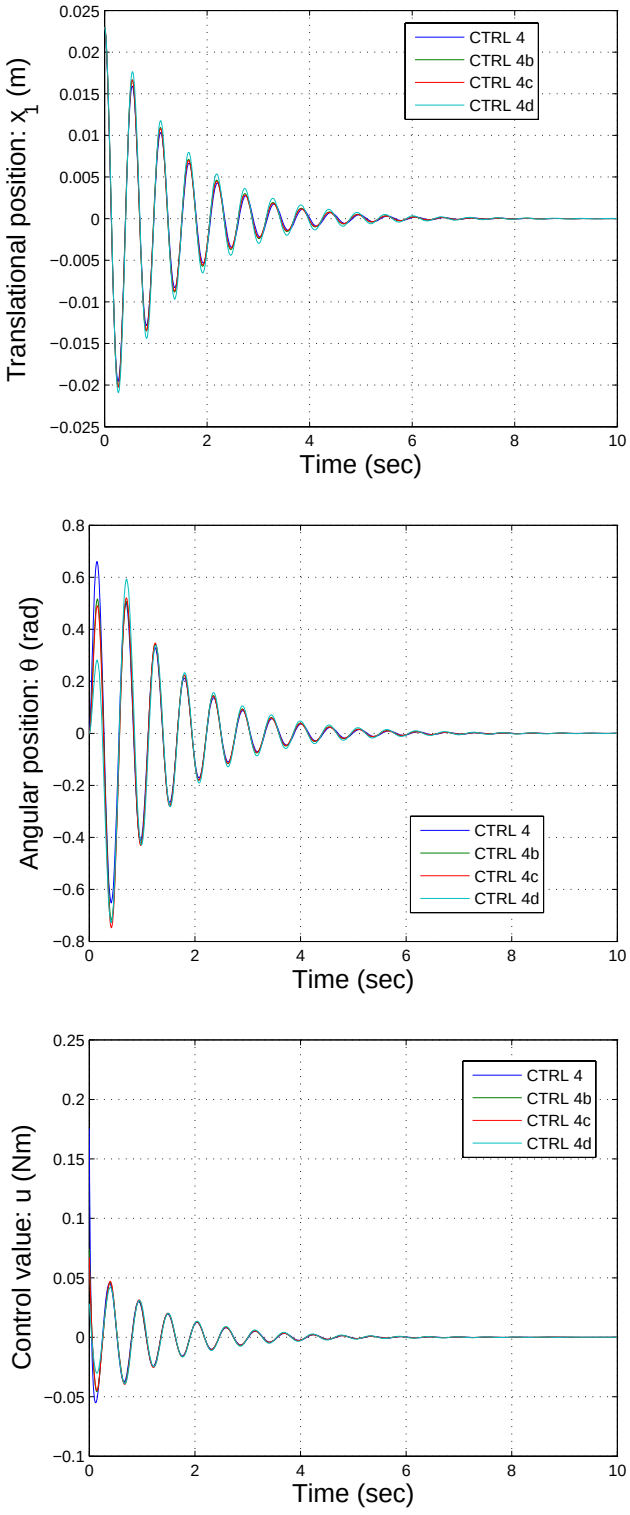

Figure 7.17: Asymptotic stability controller design of exact and reduced TP models

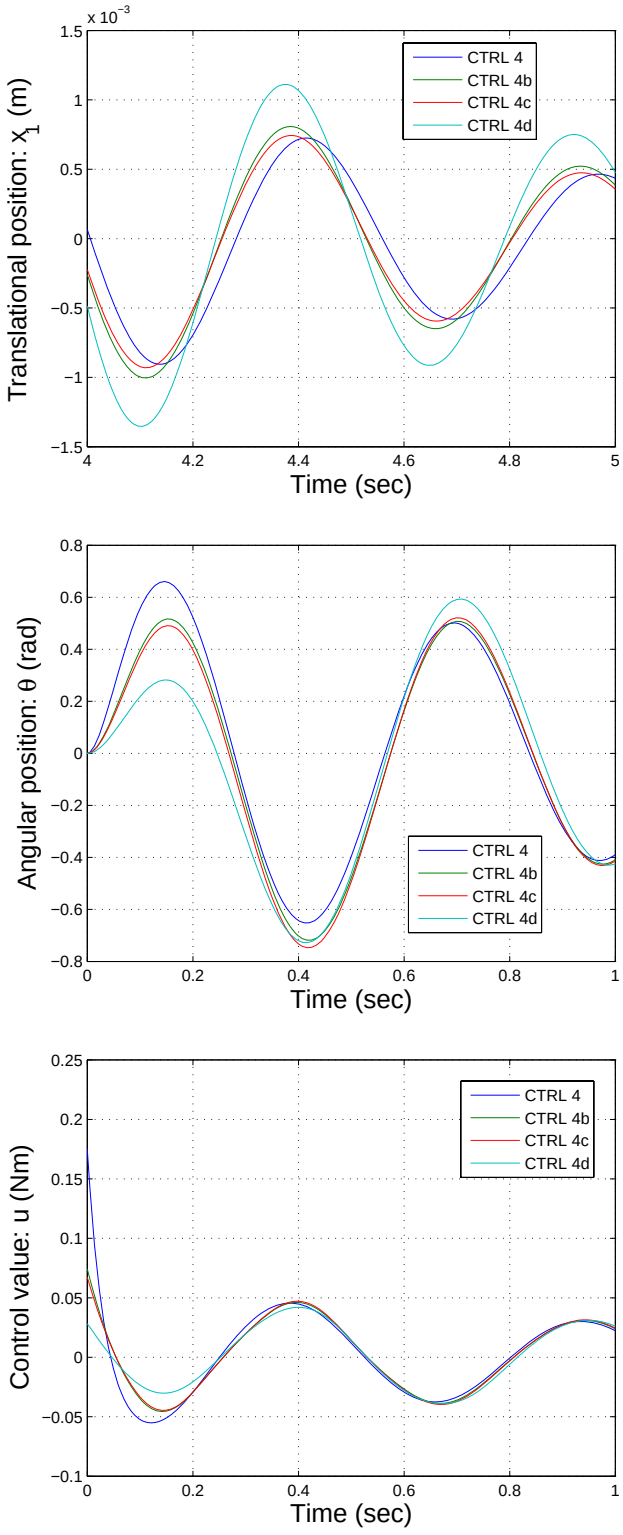

Figure 7.18: Magnification of Figure 7.17 for emphasizing the differences of controllers

Table 7.2: Summary of approximation trade-off of the TORA system

Number of singular values kept	Number of LTIs	Reduction ratio of model transformation	Upper-bound of estimated error	Measured maximal L_2 error	Number of LMIs of the controller	Reduction ration of the number of LMIs
5	10	0%	0	10^{-12}	67	0%
4	8	20%	0.00416	0.0023	46	31%
3	6	40%	0.12675	0.0024	29	56%
2	4	60%	3.96015	0.21513	16	76%

Another important issue concerning these results is that in order to derive CONTROLLER 4, an LMI system of 67 LMIs has to be solved, whilst 16 LMIs can describe CONTROLLER 2, the controller of the most reduced TP model that is a 76% of reduction. The TORA system is a simple model, the number of LMIs is moderate, but by defining more constraints, applying more complex controller specifications, such as decay rate control, observer design, etc., and if the TP model of the system consists of more LTI systems, the number of LMIs can easily explode to such a manner that is difficult to handle [81].

Table 7.2 shows a comprehensive chart on the approximation trade-off of the TORA system.

Again we have to be careful with the approximation trade-off. In case of exact TP models, the feasible solution of the LMI system are proofed to guarantee the stability and defined controller specifications for original model. In this case the solution is trackable through the LMIs and TP model transformation. On the other hand, if the TP model is only an approximation of the original model, then in mathematical sense we can only say that the stability and control specification are guaranteed only for the approximated model described by the TP model. Even in case of comprehensive series of simulation the controller shows stabilization capability, it is not trackable mathematically. For instance in case of a dynamic system with chaotic behavior, a small error can cause explosion in the system. However, in most cases the controllers derived from the approximated models are satisfactory as the modeling error of typical identification techniques in in larger range than the approximation error.

7.4 Observer based output-feedback controller design

7.4.1 Asymptotically stable observer design for the TORA system

In real-world control problems, if it often the case that the complete information of the states of a system is not always available. In such cases, one need to resort to output-feedback design methods such as observer-based designs.

As in all observer designs, TP model transformation based observers are required to satisfy

$$\mathbf{x}(t) - \hat{\mathbf{x}}(t) \to 0 \quad \text{as} \quad t \to \infty,$$

where $\hat{\mathbf{x}}(t)$ denotes the state vector estimated by a TP model transformation based observer. This condition guarantees that the steady-state error between $\mathbf{x}(t)$ and $\hat{\mathbf{x}}(t)$ converges to 0. As in the case of controller design, the PDC concept is employed to arrive at the following observer structure:

$$\begin{aligned}\dot{\hat{\mathbf{x}}}(t) &= \mathbf{A}(\mathbf{p}(t))\hat{\mathbf{x}}(t) + \mathbf{B}(\mathbf{p}(t))\mathbf{u}(t) + \mathbf{K}(\mathbf{p}(t))(\mathbf{y}(t) - \hat{\mathbf{y}}(t)) \\ \hat{\mathbf{y}}(t) &= \mathbf{C}(\mathbf{p}(t))\hat{\mathbf{x}}(t),\end{aligned}$$

That is in TP model form:

$$\begin{aligned}\dot{\hat{\mathbf{x}}}(t) &= \mathcal{A} \underset{n}{\otimes} \mathbf{w}(p_n(t))\hat{\mathbf{x}}(t) + \mathcal{B} \underset{n}{\otimes} \mathbf{w}_n(p_n(t))\mathbf{u}(t) + \\ &\quad + \mathcal{K} \underset{n}{\otimes} \mathbf{w}(p_n(t))(\mathbf{y}(t) - \hat{\mathbf{y}}(t)) \\ \hat{\mathbf{y}}(t) &= \mathcal{C} \underset{n}{\otimes} \mathbf{w}(p_n(t))\hat{\mathbf{x}}(t).\end{aligned} \tag{7.21}$$

At this point, we should emphasize that the vector $\mathbf{p}(t)$ does not contain values form the estimated state-vector $\hat{\mathbf{x}}(t)$ in our case, since $p_1(t)$ equals to angular position (x_3) and $p_2(t)$ equals to angular speed ($x_4(t)$). These variables are observable. We estimate only state-values $x_1(t)$ and $x_2(t)$. Consequently, the goal, in the present case, is to determine gains in tensor $\mathcal{K}$ for (7.21).

CONTROLLER 5 For this goal, the following LMI theorem can be derived:

Theorem 7.5 (Globally and asymptotically stable observer)**.** *Assume the polytopic model (5.3) with controller (5.4) and observer structure (7.21). This controller is globally and asymptotically stable if there exists such* $\mathbf{P}_1 > \mathbf{0}, \mathbf{P}_2 > \mathbf{0}$ *and* $\mathbf{M}_{1i}, \mathbf{N}_{2i}$ $(i = 1, \ldots, R)$ *satisfying equations*

$$\begin{aligned}\mathbf{P}_1\mathbf{A}_i^T - \mathbf{M}_{1i}^T\mathbf{B}_i^T + \mathbf{A}_i\mathbf{P}_1 - \mathbf{B}_i\mathbf{M}_{1i} &< \mathbf{0}, \\ \mathbf{A}_i^T\mathbf{P}_2 - \mathbf{C}_i^T\mathbf{N}_{2i}^T + \mathbf{P}_2\mathbf{A}_i - \mathbf{N}_{2i}\mathbf{C}_i &< \mathbf{0}, \\ \mathbf{P}_1\mathbf{A}_i^T - \mathbf{M}_{1j}^T\mathbf{B}_i^T + \mathbf{A}_j\mathbf{P}_1 - \mathbf{B}_i\mathbf{M}_{1j} + \mathbf{P}_1\mathbf{A}_j^T - \mathbf{M}_{1i}^T\mathbf{B}_j^T + \mathbf{A}_j\mathbf{P}_1 - \mathbf{B}_j\mathbf{M}_{1i} &< \mathbf{0}, \\ \mathbf{A}_i^T\mathbf{P}_2 - \mathbf{C}_j^T\mathbf{N}_{2i}^T + \mathbf{P}_2\mathbf{A}_i - \mathbf{N}_{2i}\mathbf{C}_j + \mathbf{A}_j^T\mathbf{P}_2 - \mathbf{C}_i^T\mathbf{N}_{2j}^T + \mathbf{P}_2\mathbf{A}_j - \mathbf{N}_{2j}\mathbf{C}_i &< \mathbf{0}\end{aligned}$$

for $i < j \leq R$*, except the pairs* (i, j) *such that* $w_i(\mathbf{p}(t))w_j(\mathbf{p}(t)) = 0, \forall \mathbf{p}(t)$*, and where* $\mathbf{M}_{1i} = \mathbf{F}_i\mathbf{P}_1$ *and* $\mathbf{N}_{2i} = \mathbf{P}_2\mathbf{K}_i$*.*

The matrices $\mathbf{P}_1, \mathbf{P}_2, \mathbf{M}_{1i}$ *and* $\mathbf{N}_{2i}$ *can be found using convex optimization techniques involving LMIs if they exist. The feedback gains and the observer gains can then be obtained as* $\mathbf{F}_i = \mathbf{M}_{1i}\mathbf{P}_1^{-1}$ *and* $\mathbf{K}_i = \mathbf{P}_2^{-1}\mathbf{N}_{2i}$*.*

We apply now Theorem 7.5 and Theorems 7.3 and 7.4 for system constraints to TP MODEL 2 of the TORA system. We define the matrix $\mathbf{C}$ for all r from

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t),$$

that is in present case:

$$\mathbf{C}_r = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The LMIs of Theorem 7.5, applied to TP MODEL 2, are feasible.

In conclusion the state values $x_1(t)$ and $x_2(t)$ are estimated by (7.21) as:

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{A}(\mathbf{p}(t))\hat{\mathbf{x}}(t) + \mathbf{B}(\mathbf{p}(t))u(t) +$$

$$\left(\sum_{i=1}^{5}\sum_{j=1}^{2} w_{1,i}(x_3(t))w_{2,j}(x_4(t))\mathbf{k}_{i,j}\right)(\mathbf{y}(t) - \hat{\mathbf{y}}(t)),$$

where

$$\mathbf{y}(t) = \begin{pmatrix} x_3(t) \\ x_4(t) \end{pmatrix} \quad \text{and} \quad \hat{\mathbf{y}}(t) = \begin{pmatrix} \hat{x}_3(t) \\ \hat{x}_4(t) \end{pmatrix} \quad \text{and} \quad \mathbf{p}(t) = \begin{pmatrix} x_3(t) \\ x_4(t) \end{pmatrix},$$

The design process, similar to the state-feedback controller design, does not need any analytic interaction and takes a few minutes only on regular computers. We can easily guarantee various design specification beyond stability by selecting proper LMI conditions.

7.4.2 Simulation results

To demonstrate the performance of the output-feedback control system, numerical experiments are presented in this section. In order to be comparable to other published results, the numerical examples are performed with the system parameters listed in Table 7.1, control specifications detailed in Section 7.3.1, and for initial conditions $\mathbf{x}(t) = \begin{pmatrix} 0.023\,\mathrm{m} & 0 & 0 & 0 \end{pmatrix}$. The initial observer state is $\hat{\mathbf{x}}(t) = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$. Figure 7.19 shows that the system is asymptotically stabilized. We can see that the stabilization time is a bit longer with the observer than without the observer as depicted in Figure 7.13. Figure 7.20 shows how the estimated state variables $\hat{x}_1(t)$ and $\hat{x}_2(t)$ converge to the state variables $x_1(t)$ and $x_2(t)$, while Figure 7.21 shows how the error $\mathbf{y}(t) - \hat{\mathbf{y}}(t)$ converges to zero.

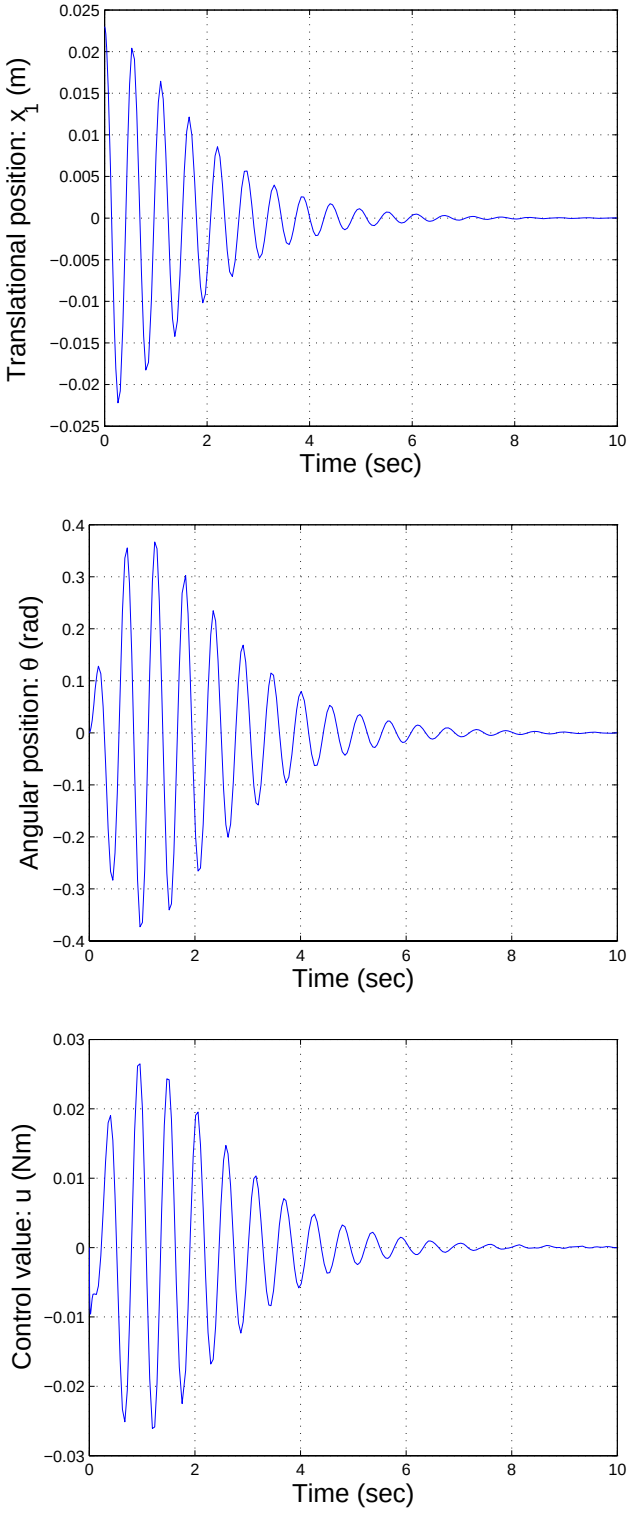

Figure 7.19: CONTROLLER 5: Observer based asymptotic stability control design with constraints, $\mathbf{x}(0) = (0.023\,0\,0\,0)$

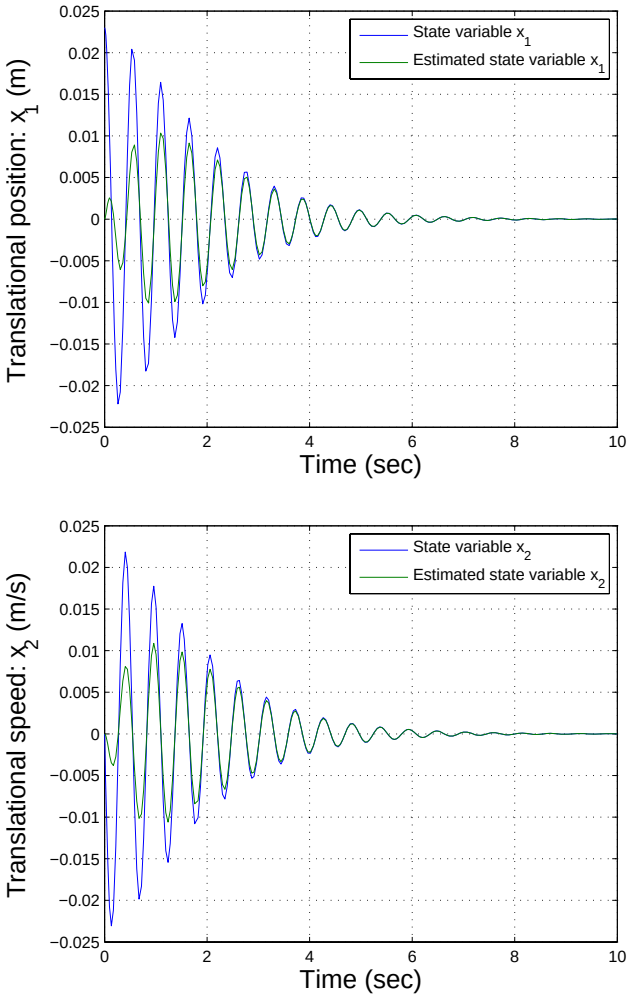

Figure 7.20: Convergence of estimated state variables x_1 and x_2 to state variables

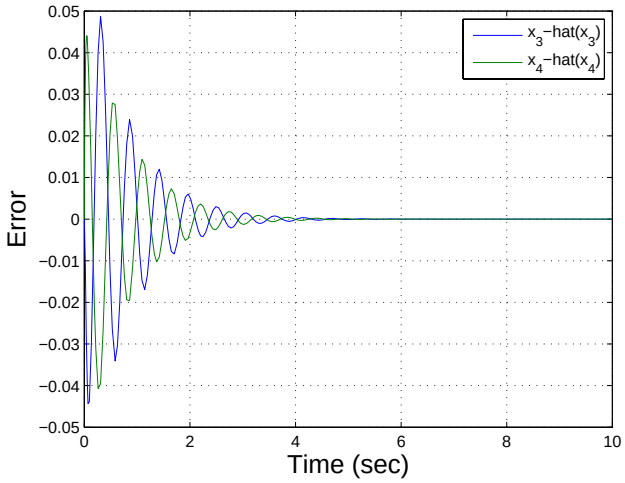

Figure 7.21: Convergence of $\mathbf{y} - \hat{\mathbf{y}}$ to zero

Chapter 8

Single Pendulum Gantry

The Single Pendulum Gantry system, an example of a translational electromechanical system, is used for educational purposes at University of Zagreb, Croatia. It is an experimental testbed, and the goal is to design, compare and evaluate several controller approaches. For more details about the testbed, please, refer to [53, 54].

8.1 Equation of motion

Let us consider the stabilization problem as shown in Figure 8.1. Only a brief discussion is presented here, for detailed description, please, refer to [53, 54]. Letting $\mathbf{x}(t) = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}^T = \begin{pmatrix} x_c & \dot{x}_c & \alpha & \dot{\alpha} \end{pmatrix}^T$, the equations of motion in linear parameter-varying state-space form is:

$$\dot{\mathbf{x}}(t) = \mathbf{A}(\mathbf{x}(t))\mathbf{x}(t) + \mathbf{B}(\mathbf{x}(t))u(t), \tag{8.1}$$

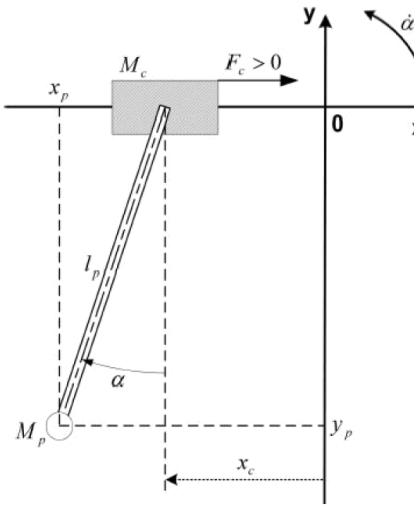

Figure 8.1: Schematic of the Single Pendulum Gantry model

Table 8.1: Parameters of the SPG system

Description	Parameter	Value	Units
Equivalent viscous damping coefficient	B_{eq}	5.4	N ms/rad
Viscous damping coefficient	B_p	0.0024	N ms/rad
Planetary gearbox efficiency	η_g	1	—
Motor efficiency	η_m	1	—
Gravitational constant of earth	g	9.81	m/s^2
Pendulum moment of inertia	I_p	0.0078838	kg m^2
Rotor moment of inertia	J_m	3.9001e-007	kg m^2
Planetary gearbox gear ratio	K_g	3.71	—
Back electro-motive force constant	K_m	0.0076776	Vs
Motor torque constant	K_t	0.007683	Nm/A
Pendulum length from pivot to COG	l_p	0.3302	m
Lumped mass of the cart system	M_c	1.0731	kg
Pendulum mass	M_p	0.23	kg
Motor armature resistance	R_m	2.6	Ω
Motor pinion radius	r_mp	0.00635	m

where

$$\mathbf{A}(\mathbf{x}(t)) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & a_1/a_x & a_2/a_x & a_3/a_x \\ 0 & 0 & 0 & 1 \\ 0 & a_4/a_x & a_5/a_x & a_6/a_x \end{pmatrix}, \mathbf{B}(\mathbf{x}(t)) = \begin{pmatrix} 0 \\ b_1/a_x \\ 0 \\ a_2/b_x \end{pmatrix},$$

$$\begin{aligned}
a_1 &= -(I_p + M_p l_p^2)\left(\frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2} + B_{eq}\right) \\
a_2 &= \frac{M_p^2 l_p^2 g \cos(x_3) sin(x_3)}{x_3} \\
a_3 &= (M_p^2 l_p^3 + l_p M_p l_p)\sin(x_3)x_4 + M_p l_p B_p \cos(x_3) \\
a_4 &= M_p l_p \cos(x_3)\left(B_{eq} - \frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2}\right) \\
a_5 &= \frac{-(M_c + M_p) M_p l_p \sin(x_3)}{x_3} \\
a_6 &= -(M_c + M_p)B_p - M_p^2 l_p^2 \cos(x_3)\sin(x_3)x_4 \\
a_x &= (M_c + M_p)I_p + M_c M_p l_p^2 + M_p^2 l_p^2 \sin^2(x_3) \\
b_1 &= -(I_p M_p l_p)^2 \frac{\eta_g K_g \eta_m K_t}{R_m r_{mp}} \\
b_2 &= -M_p l_p \cos(x_3)\frac{\eta_g K_g \eta_m K_t}{R_m r_m p}
\end{aligned}$$

The parameters of the experimental system are given in Table 8.1.

8.2 TP model representations of the Single Pendulum Gantry

Observe that the nonlinearity is caused by state values $x_3(t)$ and $x_4(t)$. The operation range of the pendulum's tip which is considered as the load of the gantry crane system is limited to ± 25 deg for safety reasons *i.e.* the swinging of the load should be limited, and the angular acceleration for the motor is maximum $0.7\,\frac{\text{rad}}{\text{s}}$. For the TP model transformation we define the transformation space as $\Omega = [\frac{-27}{180}\pi, \frac{27}{180}\pi] \times [-0.8, 0.8]$ (note that these intervals can be arbitrarily defined). Let the density of the discretization grid be 137×137. The discretization results in $\mathbf{A}^D_{i,j}$ and $\mathbf{B}^D_{i,j}$, where $i, j = 1 \dots 137$. Then we construct the matrix $\mathbf{S}^D_{i,j} = \begin{pmatrix} \mathbf{A}^D_{i,j} & \mathbf{B}^D_{i,j} \end{pmatrix}$, and after that the tensor $\mathcal{S}^D \in \mathbb{R}^{137\times137\times4\times5}$ from $\mathbf{S}^D_{i,j}$. The TP transformation is executed by the beta version of the MATLAB Toolbox for TP Model Transformation. If we execute HOSVD on the first two dimensions of $\mathcal{S}^D$ then we find that the rank of $\mathcal{S}^D$ on the first two dimensions are 7 and 2 respectively.

The singular values are as follows in the dimension x_3: $\sigma_{1,1} = 1609.4$, $\sigma_{1,2} = 206.72$, $\sigma_{1,3} = 12.604$, $\sigma_{1,4} = 10.719$, $\sigma_{1,5} = 2.3109$, $\sigma_{1,6} = 0.14075$, $\sigma_{1,7} = 0.001854$, and in the dimension x_4: $\sigma_{2,1} = 1622.7$, $\sigma_{2,2} = 10.965$. This means that the SPG system can be exactly given as convex combination of $7 \times 2 = 14$ linear vertex models (the L_2 numerical error of the TP model transformation for exact model is less than 10^{-12}). The TP model transformation describes SPG system as:

$$\dot{\mathbf{x}}(t) = \mathbf{S}(t)(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} = \sum_{i=1}^{7} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (8.2)$$

As in most cases it is to expensive in computational sense to work with 14 LTI models, and in real world situations the sensor and actuators accuracy is much worse than the modeling accuracy, it is possible to reduce the model. If we only keep the four largest singular values in dimension $x_3(t)$ and keep the two singular values in dimension $x_4(t)$, the system can be reduced to 8 LTI models. The theoretical maximum of L_2 approximation error is the sum of the discarded singular values that means $\sigma_{1,5} + \sigma_{1,6} + \sigma_{1,7} = 2.4535$ however by checking the actual L_2 error for 10000 test points, an average maximal error of 0.080307 is received. This means that by adding a maximal theoretical error of 0.13%, the measurement shows that the actual additional error is not significant at all, the complexity of the system is reduced by 43%. The resulting system model is as follows, and its weighting functions are depicted in Figure 8.2.

$$\dot{\mathbf{x}}(t) = \mathbf{S}(\mathbf{p}(t)) \begin{pmatrix} \mathbf{x}(t) \\ u(t) \end{pmatrix} \approx \sum_{i=1}^{4} \sum_{j=1}^{2} w_{1,i}(x_3(t)) w_{2,j}(x_4(t)) \left(\mathbf{A}_{i,j}\mathbf{x}(t) + \mathbf{B}_{i,j}u(t) \right). \quad (8.3)$$

The LTI system matrices of the model are:

$$\mathbf{A}_{1,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.2630 & 1.2457 & -0.0192 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 22.8870 & -24.2374 & -0.0311 \end{pmatrix} \quad \mathbf{B}_{1,1} = \begin{pmatrix} 0 \\ 1.4794 \\ 0 \\ -3.0061 \end{pmatrix}$$

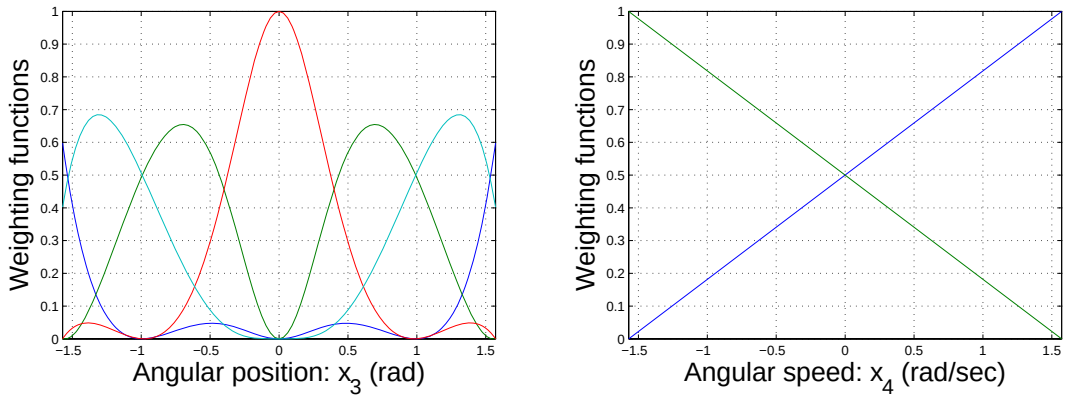

Figure 8.2: Weighting functions of the TP model on dimensions $x_3(t)$ and $x_4(t)$

$$\mathbf{A}_{1,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.2906 & 1.2657 & 0.0270 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 23.1794 & -24.3744 & -0.1306 \end{pmatrix} \quad \mathbf{B}_{1,2} = \begin{pmatrix} 0 \\ 1.4830 \\ 0 \\ -3.0455 \end{pmatrix}$$

$$\mathbf{A}_{2,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.8223 & 1.6427 & 0.0052 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 28.5811 & -26.9299 & -0.0852 \end{pmatrix} \quad \mathbf{B}_{2,1} = \begin{pmatrix} 0 \\ 1.5528 \\ 0 \\ -3.7540 \end{pmatrix}$$

$$\mathbf{A}_{2,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -12.4388 & 2.1008 & 0.0066 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 35.3681 & -30.0863 & -0.0901 \end{pmatrix} \quad \mathbf{B}_{2,2} = \begin{pmatrix} 0 \\ 1.6338 \\ 0 \\ -4.6455 \end{pmatrix}$$

$$\mathbf{A}_{3,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.2630 & 1.2457 & 0.0275 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 22.8870 & -24.2374 & -0.1316 \end{pmatrix} \quad \mathbf{B}_{3,1} = \begin{pmatrix} 0 \\ 1.4794 \\ 0 \\ -3.0061 \end{pmatrix}$$

$$\mathbf{A}_{3,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.2906 & 1.2657 & -0.0185 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 23.1794 & -24.3744 & -0.0324 \end{pmatrix} \quad \mathbf{B}_{3,2} = \begin{pmatrix} 0 \\ 1.4830 \\ 0 \\ -3.0455 \end{pmatrix}$$

$$\mathbf{A}_{4,1} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -11.8223 & 1.6427 & 0.0053 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 28.5811 & -26.9299 & -0.0855 \end{pmatrix} \quad \mathbf{B}_{4,1} = \begin{pmatrix} 0 \\ 1.5528 \\ 0 \\ -3.7540 \end{pmatrix}$$

$$\mathbf{A}_{4,2} = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -12.4388 & 2.1008 & 0.0063 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 35.3681 & -30.0863 & -0.0894 \end{pmatrix} \quad \mathbf{B}_{4,2} = \begin{pmatrix} 0 \\ 1.6338 \\ 0 \\ -4.6544 \end{pmatrix}$$

A linearized model is selected for the conventional state-feedback control design

$$\mathbf{A}_{\text{lin}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & -11.651 & 1.521 & 0.0049 \\ 0 & 0 & 0 & 1 \\ 0 & 26.845 & -26.109 & -0.0841 \end{pmatrix} \quad \mathbf{B}_{lin} = \begin{pmatrix} 0 \\ 1.530 \\ 0 \\ -3.526 \end{pmatrix} \tag{8.4}$$

8.2.1 Controller design

We compare the control performances to various different alternative solutions.

Conventional controller based on pole placement

CONTROLLER 1 (Pole Placement for the linearized system) The poles of the closed loop linearized system (8.4) with state-feedback are selected with the consideration of about 2 sec settling time and 5% overshoot in the following way

$$\mathbf{Poles} = \begin{pmatrix} -1.8182 + 1.9067i & -1.8182 - 1.9067i & -20 & -40 \end{pmatrix} \tag{8.5}$$

The state feedback control is

$$u = -\mathbf{F}^{\text{lin}} x, \quad \mathbf{F}^{\text{lin}} = \begin{pmatrix} 160 & 88 & -210 & 23 \end{pmatrix} \tag{8.6}$$

Derivation of TP based controllers

In the present case the controller (5.4) has the following form:

$$u = -\left(\sum_{i=1}^{4} \sum_{j=1}^{2} w_{1,i}(x_3) w_{2,j}(x_4) \mathbf{F}_{i,j} \right) x, \tag{8.7}$$

Two methods are presented to define the feedback gains $\mathbf{F}_{i,j}$ for the eight systems.

CONTROLLER 2 (Pole Placement for TP based system) The feedback gains $\mathbf{F}_{i,j}$ are selected separately for the all systems to place closed loop system poles to (8.5).

$$\begin{aligned}
\mathbf{F}_{1,1}^{\text{PP}} &= \begin{pmatrix} 158.45 & 86.771 & -205.71 & 22.625 \end{pmatrix} \\
\mathbf{F}_{1,2}^{\text{PP}} &= \begin{pmatrix} 172.65 & 95.228 & -243.48 & 29.380 \end{pmatrix} \\
\mathbf{F}_{2,1}^{\text{PP}} &= \begin{pmatrix} 171.96 & 94.775 & -241.20 & 29.014 \end{pmatrix} \\
\mathbf{F}_{2,2}^{\text{PP}} &= \begin{pmatrix} 140.72 & 76.248 & -162.00 & 15.788 \end{pmatrix} \\
\mathbf{F}_{3,1}^{\text{PP}} &= \begin{pmatrix} 158.45 & 86.772 & -205.72 & 22.626 \end{pmatrix} \\
\mathbf{F}_{3,2}^{\text{PP}} &= \begin{pmatrix} 172.65 & 95.185 & -243.21 & 29.392 \end{pmatrix} \\
\mathbf{F}_{4,1}^{\text{PP}} &= \begin{pmatrix} 171.96 & 94.811 & -241.43 & 28.999 \end{pmatrix} \\
\mathbf{F}_{4,2}^{\text{PP}} &= \begin{pmatrix} 140.72 & 76.238 & -161.95 & 15.783 \end{pmatrix}
\end{aligned}$$

CONTROLLER 3 (LMI for TP based system) We design here a controller capable of asymptotically stabilizing the SPG and satisfies the given constraints. By using the LMI solver of MATLAB Robust Control Toolbox, the following feasible solution and feedback gains are obtained for the controller:

$$
\begin{aligned}
\mathbf{F}_{1,1}^{\mathrm{LMI}} &= \begin{pmatrix} 118.3947 & 51.3126 & -45.6237 & 16.4703 \end{pmatrix} \\
\mathbf{F}_{1,2}^{\mathrm{LMI}} &= \begin{pmatrix} 118.0638 & 51.3291 & -46.1783 & 16.4069 \end{pmatrix} \\
\mathbf{F}_{2,1}^{\mathrm{LMI}} &= \begin{pmatrix} 117.5669 & 52.2320 & -50.2620 & 15.9900 \end{pmatrix} \\
\mathbf{F}_{2,2}^{\mathrm{LMI}} &= \begin{pmatrix} 141.1224 & 63.2115 & -56.1795 & 19.1934 \end{pmatrix} \\
\mathbf{F}_{3,1}^{\mathrm{LMI}} &= \begin{pmatrix} 118.2570 & 51.2608 & -45.6394 & 16.4747 \end{pmatrix} \\
\mathbf{F}_{3,2}^{\mathrm{LMI}} &= \begin{pmatrix} 118.2075 & 51.3926 & -46.1995 & 16.3999 \end{pmatrix} \\
\mathbf{F}_{4,1}^{\mathrm{LMI}} &= \begin{pmatrix} 117.5665 & 52.2318 & -50.2620 & 15.9900 \end{pmatrix} \\
\mathbf{F}_{4,2}^{\mathrm{LMI}} &= \begin{pmatrix} 141.1182 & 63.2101 & -56.1805 & 19.1918 \end{pmatrix}
\end{aligned}
$$

8.2.2 Experimental results

The experimental results with the three controllers are presented in Figure 8.3–8.5. The reference was a pulse train. In the first set of plots (Figure 8.3) the time functions of the reference and the load (tip of the pendulum) position is shown. In the second set of plots (Figure 8.4), the time functions of the angle of the load are shown. As it was expected, the performances of **CONTROLLER 1** and **CONTROLLER 2** are quite similar since they are set to have the same poles. The 5% overshoot can be observed well. The **CONTROLLER 3** seems to be faster but there are no significant difference among the three responses. However, **CONTROLLER 3** does not produce an overshoot, so it sticks a bit before reaching goal position. A more flexible LMI stability theorem can offer such characteristics. Around 4 sec you can also see a small jump on the plots. These are coming from the mechanical damage of the trail. The main difference appears in the control activity. According to Figure 8.5, the **CONTROLLER 3** has the most smooth time functions.

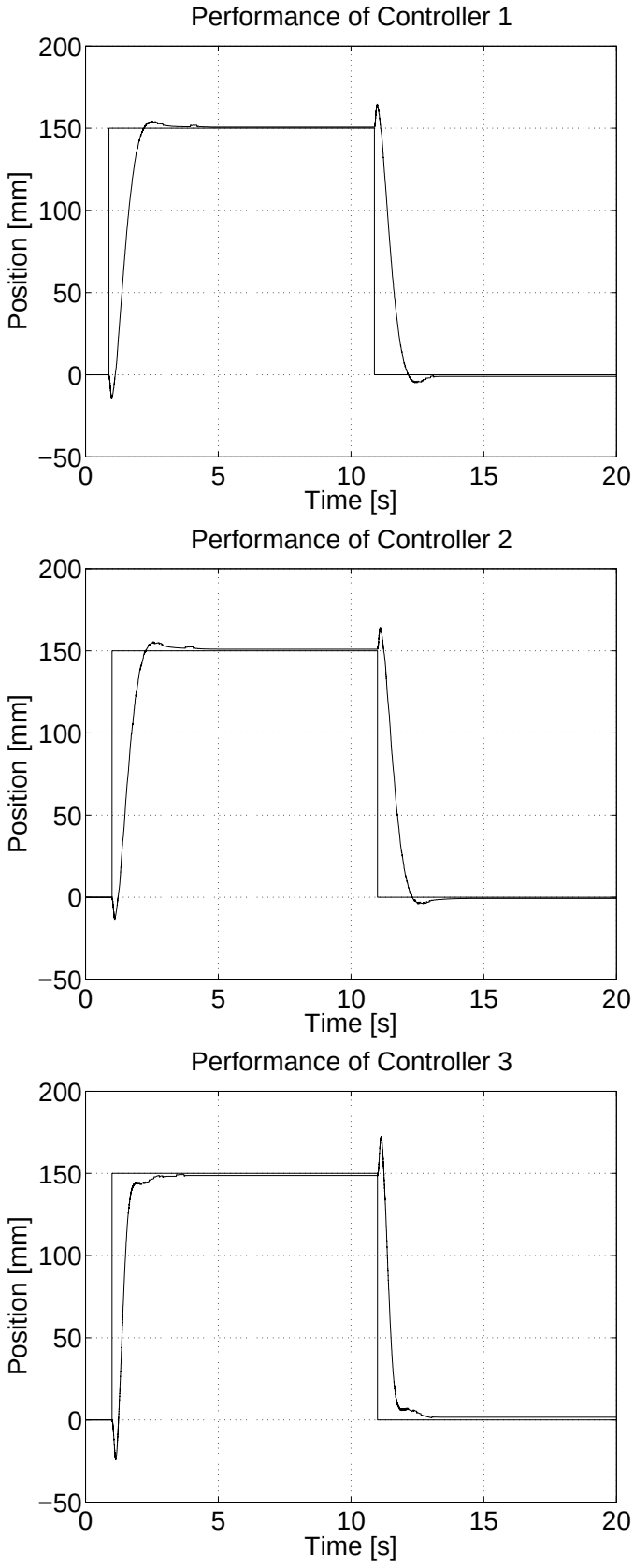

Figure 8.3: The position of the load M_p, comparison of the performances of three controllers

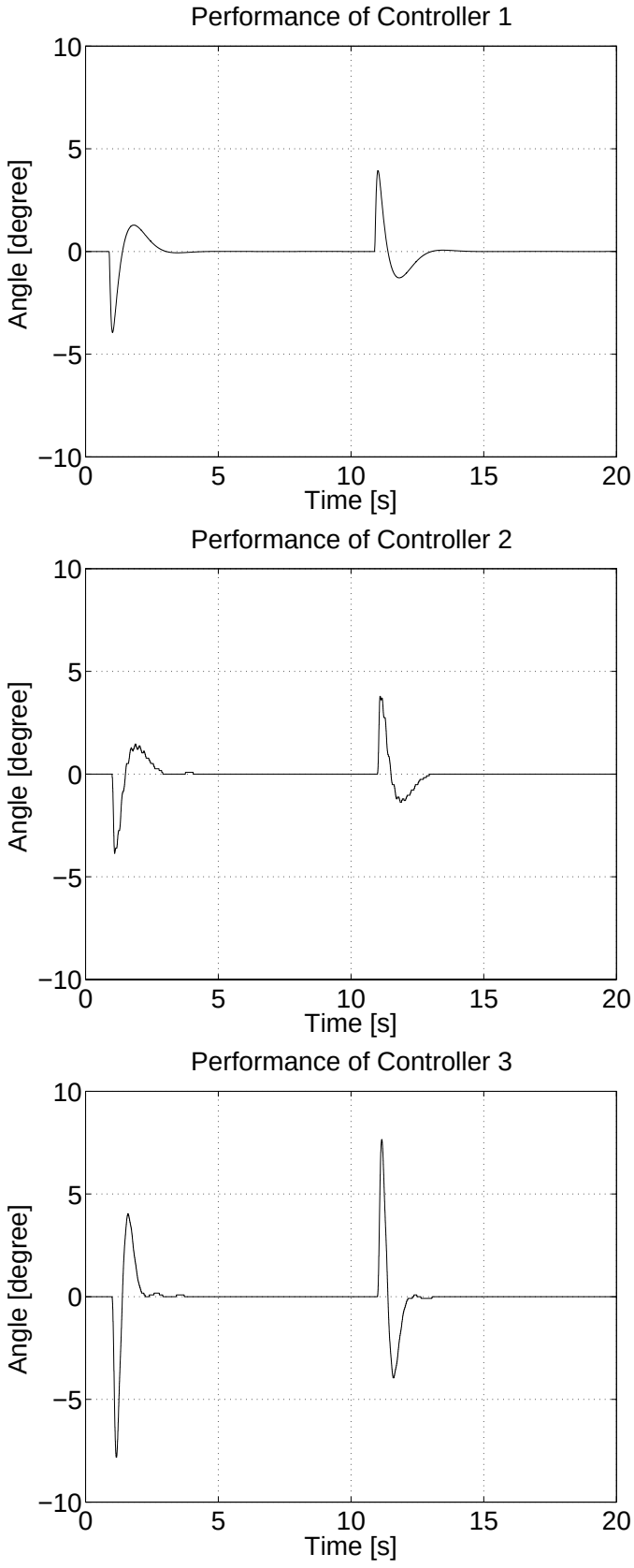

Figure 8.4: The the angle of the load (M_p), comparison of the performances of three controllers

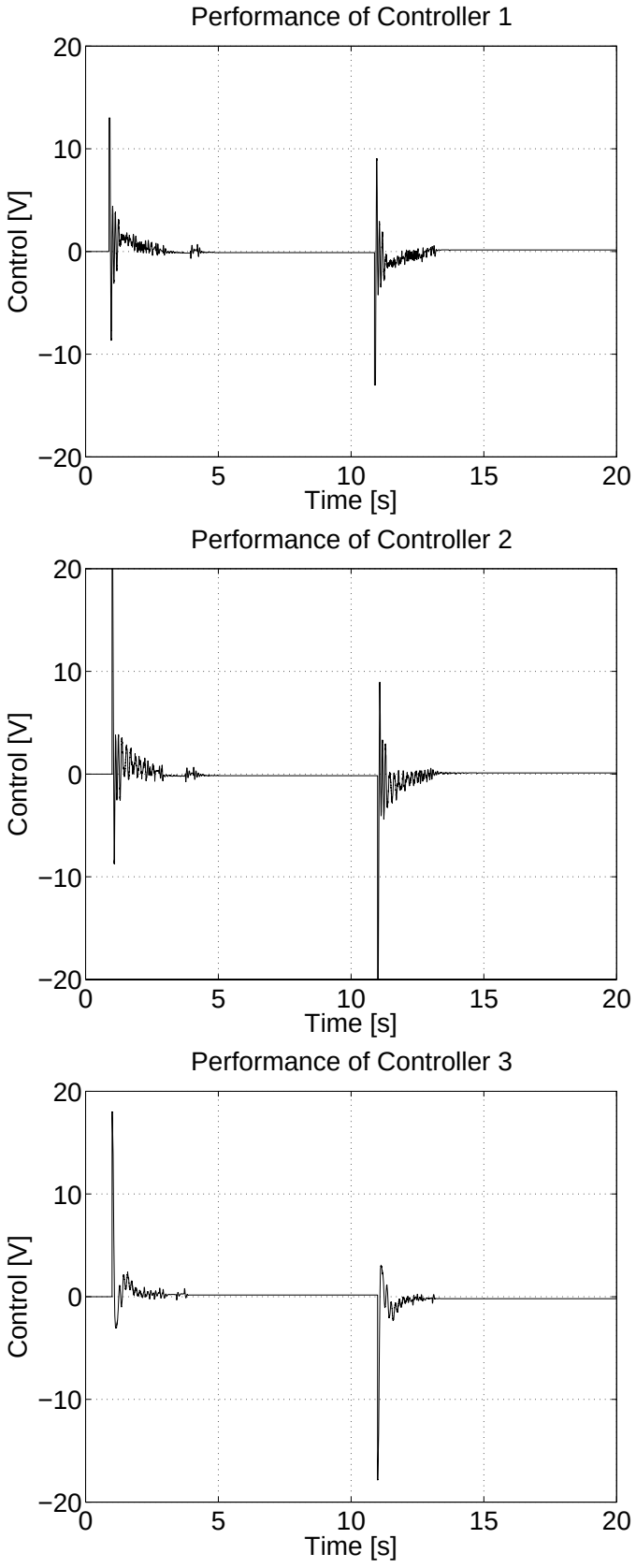

Figure 8.5: The control signal, comparison of the performances of three controllers

Part IV

Conclusion

Chapter 9

Theses

Thesis 1: Computational complexity relaxation of the Tensor Product model transformation by decreasing the discretization grid density [P–1]

I proposed a computational complexity relaxed TP model transformation that is executable on a sparse discretization grid of the system matrix that immediately leads to a computational complexity reduction. However, the utilization of the sparse discretization grid may degrade the accuracy of the convex hull searching. In order to compensate this problem, I also proposed a theoretical modification of the TP model transformation that is capable of extending the TP model to a dense discretization grid. I also developed a numerical implementation of the modified TP model transformation. The proposed implementation is ready to use in real-world applications, I made it available as a toolbox to MATLAB. The detailed algorithm is discussed in Section 6.1.1 on page 34.

I gave an estimate that the proposed relaxation polynomially reduces the computational load of the TP model transformation. In practical cases the expected number of weighting functions are small, the grid density can be reduced by several orders. The efficiency of the proposed modification is evaluated in Section 6.1.2 on page 37.

Thesis 2: Computational complexity relaxation of the Tensor Product model transformation by separating the constant and non-constant elements [P–2]

I made a suggestion for the relaxation of the computational load in TP model transformation by avoiding the computation of constant elements in the LPV model. The key idea of this approach is the separation of constant and non-constant elements of the discretized system matrix. For the numerical implementation, I developed an algorithm that decompose the system matrix into constant and non-constant matrices, then execute a modified TP model transformation on each discretized set, and finally reconstruct into the TP model form. I must

emphasize here that the proposed modification can only be applied to convex TP models, it cannot be used for HOSVD-based canonical form. The proposed method is ready to use in real-world applications, I made it available as a toolbox to MATLAB. The details are discussed in Section 6.2.1 on page 42.

I showed that the proposed separation of the constant and non-constant elements of the discretized system matrix can significantly reduce the computational load in cases when the ratio of constant elements in the discretized matrix is high. The proposed method gives a linear complexity reduction. In practical cases the ratio of the constant elements in the system matrix is about 70–80%, thus the proposed method offers a good relaxation in these cases. Section 6.2.3 on page 43 shows a numerical example for it.

Thesis 3: Solving complex control problems by Tensor Product model transformation based controller design [P–3, 5, 6, 12, 16–18]

Thesis 3.1 I proved that the finite element HOSVD-based canonical forms of the TORA and SPG systems exist as a parameter-varying weighting combination of ten, fourteen vertex systems, respectively. These new forms are given as

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = \mathcal{S} \bigotimes_{n=1}^{2} \mathbf{w}_n(p_n(t)) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{pmatrix}. \tag{9.1}$$

In this regards, I proved that different types of convex finite element TP models (Sum-Normalized and Non-Negative, tight convex, Inverted Normalized and Relaxed Normalized) exist and can be given as a convex combination of minimum ten and fourteen vertex systems, respectively for TORA and SPG systems.

In Section 7.2 on page 49 I derived all these finite element TP models for TORA system, whilst Section 8.2 on page 78 gives the TP models for SPG system. These TP model forms are new representations, they have not been published before in the scientific literature. This new representation opens a new way for LMI based controller design methodologies.

Thesis 3.2 I proved by the Lyapunov stability theorems formulated in terms of linear matrix inequalities in Section 7.2.1 and 8.2 that the continuous finite element convex polytopic model of TORA and SPG systems are controllable and observable in the space Ω.

I proved that the TP models (9.1) of the TORA and SPG systems with SN, NN, and NO type weighting functions satisfies the conditions of the Parallel Distributed Compensation based controller design. Utilizing this controller design technique I derived controllers. I proved by the LMI theorems under PDC framework that the derived controllers guarantee multi-objective control performances such as asymptotic stability and given constraints on control value and on the output vector. Further details are discussed in Section 7.3 on page 58 for TORA system and in Section 8.2.2 on page 81 for SPG system.

I also showed that the tight convex TP model of TORA system immediately applicable for observer structure based output feedback control design. Utilizing this observer structure we are capable of estimating unmeasurable state vectors, and guarantee that this output feedback control design satisfies multi-objective control performances such as asymptotic stability and given constraints on control value and on the output vector. Further details are discussed in Section 7.4 on page 71.

Thesis 3.3 I carried out the analysis of the trade-off property between the computational complexity and approximation accuracy of different types of convex (SN, NN, and CNO type weighting functions) polytopic models of the TORA system. I showed by numerical simulations that there is no significant difference in control performance of the different complexity reduced TP models, whilst the computational complexity of the TP model is reduced by 60% and the controller is 76% smaller than the original. It is also important to emphasize that not only the complexity of the controller reduced significantly, but also the complexity load of the whole control design process became much smaller as the decrease of the linear time-invariant systems directly polynomially reduce the number of LMI terms. Section 7.3.4 on page 68 gives more details information.

Nomenclature

A few comments are appropriate on the notation used in this dissertation. To facilitate the distinction between scalars, vectors, matrices, and higher-order tensors, the type of a given quantity will be reflected by its representation: scalars are denoted by lower-case letters $(a, b, \dots; \alpha, \beta, \dots)$ (italic shaped), vectors are written as bold-face lower case letters $(\mathbf{a}, \mathbf{b}, \dots)$, matrices correspond to bold-face capitals $(\mathbf{A}, \mathbf{B}, \dots)$, and tensors are written as calligraphic letters $(\mathcal{A}, \mathcal{B}, \dots)$. This notation is consistently used for lower-order parts of a given structure. For example, the entry with row index i and column index j in a matrix $\mathbf{A}$, i.e., $(\mathbf{A})_{ij}$, is symbolized by a_{ij} (also $(\mathbf{a})_i = a_i$ and $(\mathbf{A})_{i_1 i_2 \dots i_N} = a_{i_1 i_2 \dots i_N}$); furthermore, the ith column vector of a matrix $\mathbf{A}$ is denoted as $\mathbf{a}_i$, i.e., $\mathbf{A} = (\mathbf{a}_1 \mathbf{a}_2 \dots)$. To enhance the overall readability, we have made one exception to this rule: as we frequently use the characters i, j, r, and n to represent of indices (counters), I, J, R, and N will be reserved to denote the index upper bounds, unless stated otherwise. The pseudo inverse of matrix $\mathbf{A}$ is indicated by $\mathbf{A}^+$.

Symbols used in the dissertation

- $\mathcal{F}$ = feedback gains of the vertex systems
- $\mathbf{S}(\mathbf{p}(t))$ = the system matrix of linear parameter-varying state-space model
- $\mathcal{S}$ = coefficient tensor of the finite element TP model constructed from the vertex system
- σ = singular value
- Θ = discretization grid density
- $\mathbf{u}(t)$ = control value
- $\mathbf{U}_n$ = matrix representation of the weighting functions
- $\mathbf{w}(p)$ = weighting function of the finite element TP model. The superscript of the weighting functions $\mathbf{w}(p)$ shows its type, such as $\mathbf{w}^{\text{SN}}(p)$, $\mathbf{w}^{\text{NN}}(p)$, $\mathbf{w}^{\text{NO}}(p)$, $\mathbf{w}^{\text{CNO}}(p)$, $\mathbf{w}^{\text{INO}}(p)$, and $\mathbf{w}^{\text{RNO}}(p)$ mean that the type of the weighting functions are SN, NN, NO, CNO, INO, or RNO, respectively. The convex weighting functions that are at least SN and NN types are indicated as $\mathbf{w}^{\text{CO}}(p)$.
- $\mathbf{x}$ = state vector
- Ω = transformation space of the TP model transformation

Author's publication

[P–1] P. Baranyi, Z. Petres, P. Korondi, Y. Yam, and H. Hashimoto. Complexity relaxation of the Tensor Product Model Transformation for Higher Dimensional Problems. *Asian Journal of Control*, 2007. (accepted, in press).

[P–2] Z. Petres and P. Baranyi. Reference Signal Tracking Control of the TORA System: a Case Study of TP Model Transformation Based Control. *Periodica Polytechnica Electrical Engineering*, 2007. (accepted, in press).

[P–3] Z. Petres, P. Baranyi, and H. Hashimoto. Trajectory tracking by TP model transformation: case study of a benchmark problem. *IEEE Transactions on Industrial Electronics*, 2006. (accepted, in press).

[P–4] P. Baranyi, Z. Petres, P. L. Várkonyi, P. Korondi, and Y. Yam. Determination of different polytopic models of the Prototypical Aeroelastic Wing Section by TP Model Transformation. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 10(4):486–493, July 2006.

[P–5] P. Baranyi, Z. Petres, P. Várlaki, and P. Michelberger. Observer and Control law Design to the TORA System via TPDC Framework. *WSEAS Transactions on Systems*, 1(5):156–163, January 2006.

[P–6] Z. Petres and P. Baranyi. Trade-Off Properties of Tensor Product Model Transformation: a Case Study of the TORA System. *Production Systems and Information Engineering*, 4:33–51, 2006.

[P–7] Z. Petres, B. Reskó, and P. Baranyi. TP Model Transformation Based Control of the TORA System. *Production Systems and Information Engineering*, 2:159–175, 2004.

[P–8] P. Korondi and Z. Petres. Sliding Mode Control Based on Tensor Product Model Transformation. In *Proceedings of IEEE 3rd International Conference on Mechatronics (ICM 2006)*, pages 672–677, Budapest, Hungary, July 3–5 2006.

[P–9] Z. Petres, P. Baranyi, and H. Hashimoto. Decrease of the Computational Load of TP Model Transformation. In *Proceedings of IEEE 3rd International Conference on Mechatronics (ICM 2006)*, pages 655–659, Budapest, Hungary, July 3–5 2006.

[P–10] Z. Petres, P. Baranyi, F. Kolonić, and A. Poljugan. Approximation Trade-off by TP Model Transformation. In *6th International Symposium of Hungarian Researchers*

on Computational Intelligence, pages 731–741, Budapest, Hungary, November 18–19 2005.

[P–11] Z. Petres and T. Kiss. Investigation of the approximation accuracy of the TP model transformation through the prototypical aeroelastic wing and the TORA system. In *Proceedings of the 2005 International Conference on Intelligent Engineering Systems (INES 2005)*, pages 289–294, Mediterranean Sea, September 16–19 2005.

[P–12] Z. Petres, P. L. Várkonyi, P. Baranyi, and P. Korondi. Different Affine Decomposition of the Model of the TORA System by TP model transformation. In *Proceedings of the 2005 International Conference on Intelligent Engineering Systems (INES 2005)*, pages 105–110, Mediterranean Sea, September 16–19 2005.

[P–13] Z. Petres, B. Reskó, and P. Baranyi. Nonlinear Reference Signal Control of the TORA System: a TP Model Transformation Based Approach. In *Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2004)*, volume 2, pages 1081–1086, Budapest, Hungary, July 25–29 2004.

[P–14] P. Baranyi, A. R. Várkonyi-Kóczy, and Z. Petres. Reference Signal Following Control Design of the TORA System: a TP Model Transformation Based Approach. In *Proceedings of Sixth Portuguese Conference on Automatic Control (CONTROLO 2004)*, pages 85–90, Faro, Portugal, June 7–9 2004.

[P–15] A. R. Várkonyi-Kóczy, P. Baranyi, Z. Petres, S. Győri, A. E. Ruano, and P. Légrády. SVD Based Modeling of Nonlinear Systems. In *Proceedings of the 19th IEEE Instrumentation and Measurement Technology Conference*, volume 2, pages 887–891, Anchorage, Alaska, USA, May 21–23 2002.

[P–16] Z. Petres, P. Korondi, and F. Kolonić. Practical application of tensor product model transformation based control design. *IFAC Control Engineering Practice*, 2006. (submitted).

[P–17] Z. Petres, P. Baranyi, and H. Hashimoto. Approximation and Complexity Trade-off by TP model transformation in Controller Design: a Case Study of the TORA system. *Asian Journal of Control*, 2005. (submitted).

[P–18] P. Baranyi, Z. Petres, P. Várlaki, and P. Michelberger. Tensor Product Model Transformation Based Control of Translational Oscillations with an Eccentric Rotational Proof Mass Actuator (TORA) System. *Journal of Guidance, Control, and Dynamics*, 2005. (submitted).

Bibliography

[1] P. Apkarian and P. Gahinet. A convex characterization of gain-scheduled H_∞ controllers. *IEEE Trans. Aut. Contr.*, 1995.

[2] P. Apkarian, P. Gahinet, and G. Becker. Self-scheduled H_∞ control of linear parameter-varying systems. *Proc. Amer. Contr. Conf.*, pages 856–860, 1994.

[3] V. I. Arnold. On functions of three variables. *Doklady Akademii Nauk USSR*, 114:679–681, 1957.

[4] R. Bambang, E. Shimemura, and K. Uchida. Mixed H_2/H_∞ control with pole placement, state-feedback case. In *Proceeding of American Control Conference*, pages 2777–2779, 1993.

[5] P. Baranyi. TP model transformation as a way to LMI based controller design. *IEEE Transaction on Industrial Electronics*, 51(2):387–400, April 2004.

[6] P. Baranyi. Output feedback control of two-dimensional aeroelastic system. *Journal of Guidance, Control, and Dynamics*, 29(3):762–767, May-June 2005.

[7] P. Baranyi. Tensor-product model-based control of two-dimensional aeroelastic system. *Journal of Guidance, Control, and Dynamics*, 29(2):391–400, May-June 2005.

[8] P. Baranyi, P. Korondi, R. J. Patton, and H. Hashimoto. Global asymptotic stabilisation of the prototypical aeroelastic wing section via TP model transfromation. *Asian Journal of Control*, 7(2):99–111, 2004.

[9] P. Baranyi, Z. Petres, P. Várlaki, and P. Michelberger. Observer and Control law Design to the TORA System via TPDC Framework. *WSEAS Transactions on Systems*, 1(5):156–163, January 2006.

[10] P. Baranyi, L. Szeidl, P. Várlaki, and Y. Yam. Definition of the HOSVD-based canonical form of polytopic dynamic models. In *3rd International Conference on Mechatronics (ICM 2006)*, pages 660–665, Budapest, Hungary, July 3-5 2006.

[11] P. Baranyi, L. Szeidl, P. Várlaki, and Y. Yam. Numerical reconstruction of the HOSVD-based canonical form of polytopic dynamic models. In *10th International Conference on Intelligent Engineering Systems*, pages 196–201, London, UK, June 26-28 2006.

[12] P. Baranyi, D. Tikk, Y. Yam, and R. J. Patton. From differential equations to PDC controller design via numerical transformation. *Computers in Industry, Elsevier Science*, 51:281–297, 2003.

[13] P. Baranyi, A. R. Varkonyi-Koczy, P. Varlaki, P. Michelberger, and R. J. Patton. *Singular Value Based Model Approximation*. World Scientific and Engineering Society Press, Danvers, section in n. mastorakis (ed.) problems in applied mathematics and computational intelligence, mathematics and computers in sci. and eng. edition, 2001.

[14] P. Baranyi, P. L. Várkonyi, and P. Korondi. Different affine decomposition of the model of the prototypical aeroelastic wing section by TP model transformation. In *IEEE 9th International Conference on Intelligent Engineering Systems*, pages 93–98, Mediterranean Sea, September 16–19 2005.

[15] P. Baranyi and Y. Yam. *Fuzzy rule base reduction*, chapter Chapter 7 of Fuzzy IF-THEN Rules in Computational Intelligence: Theory and Applications, pages 135–160. Kluwer, 2000.

[16] B. R. Barmish. Stabilization of uncertain systems via linear control. *IEEE Transaction on Automatic Control*, AC-28:848–850, 1983.

[17] E. K. Blum and L. K. Li. Approximation theory and feedforward networks. *Neural Netwroks*, 4(4):511–515, 1991.

[18] J. Bokor, P. Baranyi, P. Michelbereger, and P. Varlaki. Tp model transformation in non-linear system control. In *3rd IEEE International Conference on Computational Cybernetics (ICCC)*, pages 111–119, Mauritius, Greece, 13-16 April 2005.

[19] S. Boyd, V. Balakrishnan, and P. Kabamba. A bisection method for computing the H_∞ norm of a transfer matrix related problems. *Math. Contr. Sign. Syst.*, 2:207–219, 1989.

[20] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in Systems and Control Theory*. SIAM books, Philadelphia, 1994.

[21] S. Boyd and Q. Yang. Structured and simultaneous Lyapunov functions for system stability problems. *International Journal on Control*, 49:2215–2240, 1989.

[22] R. Bupp, D. S. Bernstein, and V. T. Coppola. A benchmark problem for nonlinear control design. *International Journal of Robust and Nonlinear Control*, 8:307–310, 1998.

[23] R. Bupp, V. T. Coppola, and D. S. Bernstein. Vibration suppression of multi-modal translational motion using a rotational actuator. In *Proc. of the IEEE Int. Decision and Control*, pages 4030–4034, Orlando, FL, 1994.

[24] R. T. Bupp, D. S. Bernstein, and V. T. Coppola. A benchmark problem for nonlinear control design: Problem statement, experiment testbed and passive nonlinear compensation. *American Control Conference, Seattle*, pages 4363–4367, 1995.

[25] J. L. Castro. Fuzzy logic controllers are universal approximators. *IEEE Trans. on SMC*, 25:629–635, 1995.

[26] M. Chilali and P. Gahinet. H_∞ design with pole placement constraints: an LMI approach. In *Proceedings of Conference on Decision Control*, pages 553–558, 1994.

[27] G. Cybenko. Approximation by superposition of sigmoidal functions. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.

[28] E. F. Deprettere, editor. *SVD and Signal Processing*, volume Algorithms, Applications and Architectures. North-Holland, Amsterdam, 1988.

[29] J. C. Doyle, K. Glover, P. Khargonekar, and B. Francis. State-space solutions to standard H_2 and H_∞ control problems. *IEEE Trans. Aut. Contr.*, AC-34:831–847, 1989.

[30] A. Edelmayer and J. Bokor. Optimal H_2 and H_∞ scaling for sensitivity optimization detection filters. *International Journal of Robust and Nonlinear Control*, 12(8):749–760, 2002.

[31] G. Escobar, R. Ortega, and H. Sira-Ramirez. Output-feedback global stabilization of a nonlinear benchmark system using a saturated passivity-based controller. *IEEE Transaction on Control System Technology*, 7(2):289–293, 1999.

[32] E. Feron, P. Apkarian, and P. Gahinet. S-procedure for the analyisi of control systems with parametric uncertanities via parameter-dependent lyapunov functions. *Thrid SIAM Conf. on Contr. and its Applic.*, 1995.

[33] P. Gahinet. Explicit controller formulas for lmi-based H_∞ synthesis. *Automatica and also in Proc. Amer. Contr. Conf.*, pages 2396–2400, 1994.

[34] P. Gahinet and P. Apkarian. A linear matrix inequality approach to H_∞ control. *International Journal on Robust and Nonlinear Control*, 4:421–448, 1994.

[35] P. Gahinet, P. Apkarian, and M. Chilali. Affine parameter-dependent Lyapunov functions for real parametric uncertainty. In *Proceedings of Conference on Decision Control*, pages 2026–2031, 1994.

[36] P. Gahinet and A. J. Laub. Reliable computation of γ_{opt} in singular H_∞ control. *SIAM J. Contr. Opt., also in Proc. Conf. Dec. Contr.*, pages 1527–1532, 1994.

[37] P. Gahinet, A. Nemirovskii, A. J. Laub, and M. Chilali. *LMI Control Toolbox User's Guide*. The MathWorks, Inc., 1995.

[38] P. Gáspár and J. Bokor. *Progress in system and robot analysis and control design*. Springer, 1999.

[39] G. H. Golub and W. Kahan. Calculating the singular values and pseudoinverse of a matrix. *SIAM Journal on Numerical Analysis*, 2:205–224, 1965.

[40] I. Grattan-Guinness. A sideways look at hilbert's twenty-three problems of 1900. *Notices of the AMS*, 47, 2000.

[41] J. Gray. The hilbert problems 1900-2000. *Newsletter*, 36:10–13, 2000.

[42] G. Hancke and A. Szeghegyi. Application study of the TP model transformation in the control of an inverted pendulum. In *International Conference on Computational Cybernetics (ICCC)*, Siófok, Hungary, 2003.

[43] D. Hilbert. Methematische probleme. *2nd International Congress of Mathematican*, 1900. Paris, France.

[44] H. P. Horisberger and P. R. Belanger. Regulators for linear time-varying plants with uncertain parameters. *IEEE Transaction on Automatic Control*, AC-21:705–708, 1976.

[45] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[46] T. Iwasaki and R. E. Skelton. All controllers for the general H_∞ control problem: LMI existence conditions and state-space formulas. *Automatica*, 30:1307–1317, 1994.

[47] M. Jankovic, D. Fontanie, and P. V. Kokotovic. Tora example: cascade- and passivity-based control designs. *IEEE Transaction on Control System Technologies*, 4:292–297, 1996.

[48] N. Kamarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[49] I. Kaminer, P. P. Khargonekar, and M. A. Rotea. Mixed $\mathrm{H}_2/\mathrm{H}_\infty$ control fir discrete time systems via convex optimization. *Automatica*, 29:57–70, 1993.

[50] I. Kaplansky. Hilbert's problemsversity of chicago. 1977.

[51] P. P. Khargonekar and M. A. Rotea. Mixed H_2/H_∞ control: a convex optimization approach. *IEEE Transaction on Automatic Control*, 39:824–837, 1991.

[52] A. N. Kolmogorov. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Dokl. Akad. USSR*, 114:953–956, 1957.

[53] F. Kolonić, A. Poljugan, and A. Slutej. Modern laboratory concept for mechatronic education. In *XXVII International Convention MIPRO 2004*, pages 143–146, Opatija, Croatia, May 2004.

[54] F. Kolonić, A. Poljugan, and Željko Jakopović. Laboratory-based and industrial-oriented course in mehatronics. In *Proceedings of 13th International Conference on Electrical Drives and Power Electronics (EDPE'05)*, pages 1–8, Dubrovnik, 2005.

[55] B. Kosko. Fuzzy systems as universal approximators. *Proc. of the IEEE Int. Conf. On Fuzzy Systems*, pages 1153–1162, 1992. San Diego.

[56] P. M. Kroonenberg. *Three-Mode Principal Component Analysis*. DSWO Press, Leiden, 1983.

[57] L. D. Lathauwer. *Signal Processing Based on Multilinear Algebra*. PhD thesis, K.U. Leuven, E.E. Dept.-ESAT, Belgium, 1997.

[58] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. Blind source separation by higher-order singular value decomposition. In *Signal Processing VII: Theories and Applications, Proc. EUSIPCO-94*, pages 175–178, Edinburgh, UK, 1994.

[59] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. Dimensionality reduction in higher-order-only ica. In *IEEE Signal Processing workshop on HOS*, pages 316–320, Banff, Alberta, Canada, July 21-23 1997.

[60] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[61] G. G. Lorentz. *Approximation of functions*. Holt, Reinhard and Winston, 1966. New York.

[62] I. Masubuchi, A. Ohara, and N. Suda. LMI-based controller synthesis: A unified formulation and solution. *International Journal on Robust and Nonlinear Control*, 8(8):669–686, 1998.

[63] M. Moonen and B. D. Moor, editors. *SVD and Signal Processing*, volume III. Algorithms, Applications and Architectures. Elsevier, Amsterdam, 1995.

[64] A. Nemirovski and P. Gahinet. The projective method for solving linear matrix inequalities. *Proc. Amer. Contr. Conf.*, pages 840–844, 1994.

[65] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, 1994.

[66] H. T. Nguyen and V. Kreinovich. On approximations of controls by fuzzy systems. LIFE Chair of Fuzzy Theory TR 92-93/302, Tokyo Institute of Technology, 1992.

[67] A. Packard and J. C. Doyle. The complex structured singular value. *Automatica*, 29:71–109, 1994.

[68] Z. Petres, B. Reskó, and P. Baranyi. Nonlinear Reference Signal Control of the TORA System: a TP Model Transformation Based Approach. In *Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2004)*, volume 2, pages 1081–1086, Budapest, Hungary, July 25–29 2004.

[69] Z. Petres, P. L. Várkonyi, P. Baranyi, and P. Korondi. Different Affine Decomposition of the Model of the TORA System by TP model transformation. In *Proceedings of the 2005 International Conference on Intelligent Engineering Systems (INES 2005)*, pages 105–110, Mediterranean Sea, September 16–19 2005.

[70] R. H. Rand, R. J. Kinesey, and D. L. Mingori. Dynamics of spinup through resonance. *International Journal Non-linear Mechanics*, 27:489–502, 1992.

[71] C. Scherer. H_∞ optimization without assumptions on finite or infinite zeros. *SIAM J. Contr. Opt.*, 30:143–166, 1992.

[72] Special. This special issue presents 9 papers dealing with the control of the tora system. *International Journal of Robust and Nonlinear Control*, 8:305–457, 1998.

[73] D. A. Sprecher. On the structure of continuous functions of several variables. *Trans. Amer. Math. Soc.*, 115:340–355, 1965.

[74] G. Stein and J. C. Doyle. Beyond singular values and loop shapes. *Journal of Guidance*, 14:5–16, 1991.

[75] G. W. Stewart. On the early history of singular value decomposition. Technical Report TR–92–31, Institute for Advanced Computer Studies, University of Mariland, March 1992.

[76] Z. Szabó, J. Bokor, and F. Schipp. Identification of rational approximate models in H_∞ using generalized orthonormal basis. *IEEE Transactions on Automatic Control*, 44(1):153–158, 1999.

[77] G. Tadmor. Dissipative design, lossless dynamics, and the nonlinear tora benchmark example. *IEEE Transaction on Control System Technology*, 9(2):391–398, 2001.

[78] K. Tanaka and M. Sugeno. Stability analysis and design of fuzzy control systems. *Fuzzy Sets and Systems*, 45(2):135–156, 1992.

[79] K. Tanaka, T. Taniguchi, and H. O. Wang. Model-based fuzzy control of TORA system: Fuzzy regulator and fuzzy observer design via LMIs that represent decay rate, disturbance rejection, robustness, optimality. In *Proceedings of Seventh IEEE International Conference on Fuzzy Systems*, pages 313–318, Alaska, 1998.

[80] K. Tanaka and H. O. Wang. *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. John Wiley & Sons, Inc., 2001.

[81] D. Tikk, P. Baranyi, R. J. Patton, and J. K. Tar. Approximation capability of TP model forms. *Australian Journal of Intelligent Information Processing Systems*, 8(3):155–163, 2004.

[82] D. Tikk, L. T. Kóczy, and T. D. Gedeon. A survey on the universal approximation and its limits in soft computing techniques. *Int. J. of Approx. Reasoning*, 33(2):185–202, June 2003.

[83] R. Vaccaro, editor. *SVD and Signal Processing*, volume II. Algorithms, Applications and Architectures. Elsevier, Amsterdam, 1991.

[84] P. Várkonyi, D. Tikk, P. Korondi, and P. Baranyi. A new algorithm for RNO-INO type tensor product model representation. In *IEEE 9th International Conference on Intelligent Engineering Systems*, pages 263–266, Athens, Greece, 16-19 September 2005. ISBN: 0-7803-9474-7.

[85] C. J. Wan, D. S. Bernstein, and V. T. Coppola. Global stabilisation of the oscillating eccentric rotor. *Nonlinear Dynamics*, 10:49–62, 1996.

[86] H. O. Wang, K. Tanaka, and M. F. Griffin. An analytical framework of fuzzy modeling and control of nonlinear systems: Stability and design issues. In *Proceedings of 1995 American Control Conference*, pages 2272–2276, Seattle, USA, 1995.

[87] H. O. Wang, K. Tanaka, and M. F. Griffin. Parallel distributed compensation of nonlinear systems by takagi–sugeno fuzzy model. In *Proceedings of FUZZ-IEEE/IFES'95*, pages 531–538, 1995.

[88] L. X. Wang. Fuzzy systems are universal approximators. *Proc. of the IEEE Int. Conf. On Fuzzy Systems*, pages 1163–1169, 1992. San Diego.

[89] B. Wie and D. S. Bernstein. Benchmark problems in robust control design. *Journal of Guidance, Control and Dynamics*, 15:1057–1059, 1992.

[90] Y. Yam, P. Baranyi, and C. T. Yang. Reduction of fuzzy rule base via singular value decomposition. *IEEE Transactions on Fuzzy Systems*, 7(2):120–132, 1999.

[91] Y. Yam, C. T. Yang, and P. Baranyi. Singular value-based fuzzy reduction with relaxed normalization condition. In J. Casillas, O. Cordón, F. Herrera, and L. Magdalena, editors, *Interpretability Issues in Fuzzy Modeling*, volume 128 of *Studies in Fuzziness and Soft Computing*, pages 325–354. Springer-Verlag, 2003.

[92] P. M. Young, M. P. Newlin, and J. C. Doyle. *Robust Control Theory*, chapter Let's Get Real, pages 143–174. Springer Verlag, 1994.