Tools for Control System Design — Stratification of Matrix Pairs and Periodic Riccati Differential Equation Solvers

Stefan Johansson

PhD Thesis, February 2009

Department of Computing Science
SE-901 87 Umeå

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

stefanj@cs.umu.se

ISBN 978-91-7264-733-6
ISSN 0348-0542
UMINF 09.04

Printed by Print & Media, Umeå University, 2009

To Annica

Abstract

Modern control theory is today an interdisciplinary area of research. Just as much as this can be problematic, it also provides a rich research environment where practice and theory meet. This Thesis is conducted in the borderline between computing science (numerical analysis) and applied control theory. The design and analysis of a modern control system is a complex problem that requires high qualitative software to accomplish. Ideally, such software should be based on robust methods and numerical stable algorithms that provide quantitative as well as qualitative information.

The introduction of the Thesis is dedicated to the underlying control theory and to introduce the reader to the main subjects. Throughout the Thesis, the theory is illustrated with several examples, and similarities and differences between the terminology from mathematics, systems and control theory, and numerical linear algebra are highlighted. The main contributions of the Thesis are structured in two parts, dealing with two mainly unrelated subjects.

Part I is devoted to the qualitative information which is provided by the stratification of orbits and bundles of matrices, matrix pencils and system pencils. Before the theory of stratification is established the reader is introduced to different canonical forms which reveal the system characteristics of the model under investigation. A stratification reveals which canonical structures of matrix (system) pencils are near each other in the sense of small perturbations of the data. Fundamental concepts in systems and control, like controllability and observability of linear continuous-time systems, are considered and it is shown how these system characteristics can be investigated using the stratification theory. New results are presented in the form of the cover relations (nearest neighbours) for controllability and observability pairs. Moreover, the permutation matrices which take a matrix pencil in the Kronecker canonical form to the corresponding system pencil in (generalized) Brunovsky canonical form are derived. Two novel algorithms for determining the permutation matrices are provided.

Part II deals with numerical methods for solving periodic Riccati differential equations (PRDE:s). The PRDE:s under investigation arise when solving the linear quadratic regulator (LQR) problem for periodic linear time-varying (LTV) systems. These types of (periodic) LQR problems turn up for example in motion planning of underactuated mechanical systems, like a humanoid robot, the Furuta pendulum, and pendulums on carts. The constructions of the nonlinear controllers are based on linear versions found by stabilizing transverse dynamics of the systems along cycles.

Three different methods explicitly designed for solving the PRDE are evaluated on both artificial systems and stabilizing problems originating from experimental control systems. The methods are the one-shot generator method and two recently proposed methods: the multi-shot method (two variants) and the SDP method. As these methods use different approaches to solve the PRDE, their numerical behavior and performance are dependent on the nature of the underlying control problem. Such method characteristics are investigated and summarized with respect to different user requirements (the need for accuracy and possible restrictions on the solution time).

Sammanfattning

Modern reglerteknik är idag i högsta grad ett interdisciplinärt forskningsområde. Lika mycket som detta kan vara problematiskt, resulterar det i en stimulerande forskningsmiljö där både praktik och teori knyts samman. Denna avhandling är utförd i gränsområdet mellan datavetenskap (numerisk analys) och tillämpad reglerteknik. Att designa och analysera ett modernt styrsystem är ett komplext problem som erfordrar högkvalitativ mjukvara. Det ideala är att mjukvaran består av robusta metoder och numeriskt stabila algoritmer som kan leverera både kvantitativ och kvalitativ information.

Introduktionen till avhandlingen beskriver grundläggande styr- och reglerteori samt ger en introduktion till de huvudsakliga problemställningarna. Genom hela avhandlingen illustreras teori med exempel. Vidare belyses likheter och skillnader i terminologin som används inom matematik, styr- och reglerteori samt numerisk linjär algebra. Avhandlingen är uppdelade i två delar som behandlar två i huvudsak orelaterade problemklasser.

Del I ägnas åt den kvalitativa informationen som ges av stratifiering av mångfalder (orbits och bundles) av matriser, matrisknippen och systemknippen. Innan teorin för stratifiering introduceras beskrivs olika kanoniska former, vilka var och en avslöjar olika systemegenskaper hos den undersökta modellen. En stratifiering ger information om bl.a. vilka kanoniska strukturer av matrisknippen (systemknippen) som är nära varandra med avseende på små störningar i datat. Fundamentala koncept i styr- och reglerteori behandlas, så som styrbarhet och observerbarhet av linjära tidskontinuerliga system, och hur dessa systemegenskaper kan undersökas med hjälp av stratifiering. Nya resultat presenteras i form av relationerna för täckande (närmsta grannar) styrbarhets- och observerbarhets-par. Dessutom härleds permutationsmatriserna som tar ett matrisknippe i Kroneckers kanoniska form till motsvarande systemknippe i (generaliserade) Brunovskys kanoniska form. Två algoritmer för att bestämma dessa permutationsmatriser presenteras.

Del II avhandlar numeriska metoder för att lösa periodiska Riccati differentialekvationer (PRDE:er). De undersökta PRDE:erna uppkommer när ett linjärt kvadratiskt regulatorproblem för periodiska linjära tidsvariabla (LTV) system löses. Dessa typer av (periodiska) regulatorproblem dyker upp till exempel när man planerar rörelser för understyrda (underactuated) mekaniska system, så som en humanoid (mänsklig) robot, Furuta-pendeln och en vagn med en inverterad (stående) pendel. Konstruktionen av det icke-linjära styrsystemet är baserat på en linjär variant som bestäms via stabilisering av systemets transversella dynamik längs med cirkulära banor.

Tre metoder explicit konstruerade för att lösa PRDE:er evalueras på både artificiella system och stabiliseringsproblem av experimentella styrsystem. Metoderna är sk. en- och flerskotts metoder (one-shot, multi-shot) och SDP-metoden. Då dessa metoder använder olika tillvägagångssätt för att lösa en PRDE, beror dess numeriska egenskaper och effektivitet på det underliggande styrproblemet. Sådana metodegenskaper undersöks och sammanfattas med avseende på olika användares behov, t.ex. önskad noggrannhet och tänkbar begränsning i hur lång tid det får ta att hitta en lösning.

Preface

This PhD Thesis consists of two parts, where each part consists of two papers. The first part is focused on canonical forms and stratification of matrix and system pencils. The second part is devoted to numerical methods for solving periodic Riccati differential equations. The Thesis also includes a brief introduction to linear control theory and control system design, and summaries of the two parts and the papers.

Part I – Canonical Structure Information and Stratification

Paper I S. Johansson. *Reviewing the Closure Hierarchy of Orbits and Bundles of System Pencils and Their Canonical Forms*. UMINF-09.02, Department of Computing Science, Umeå University, Sweden, 2009.

Paper II E. Elmroth, S. Johansson, and B. Kågström. Stratification of Controllability and Observability Pairs - Theory and Use in Applications[1]. *SIAM J. Matrix Analysis and Applications*, accepted for publication, revised October 2008. Also as Report UMINF-08.03, Department of Computing Science, Umeå University, Sweden, 2008.

Part II – Periodic Riccati Differential Equations

Paper III S. Johansson, B. Kågström, A. Shiriaev, and A. Varga. Comparing One-shot and Multi-shot Methods for Solving Periodic Riccati Differential Equations[2]. *In Proc. of the 3rd IFAC Workshop Periodic Control Systems* (PSYCO'07), St Petersburg, Russia, 2007.

Paper IV S. Gusev, S. Johansson, B. Kågström, A. Shiriaev, and A. Varga. *A Numerical Evaluation of Solvers for the Periodic Riccati Differential Equation*. UMINF-09.03, Department of Computing Science, Umeå University, Sweden, 2009.

Other papers produced within the PhD studies but not included in the Thesis are:

S. Johansson. *Canonical Forms and Stratification of Orbits and Bundles of System Pencils*. UMINF-05.16, Department of Computing Science, Umeå University, Sweden, 2005.

E. Elmroth, P. Johansson, S. Johansson, and B. Kågström. Orbit and Bundle Stratification for Controllability and Observability Matrix Pairs in StratiGraph. *In Proc. of the Sixteenth International Symposium on Mathematical Theory of Networks and Systems* (MTNS'04), Leuven, Belgium, 2004.

Acknowledgements

I am grateful to my supervisors Bo Kågström and Erik Elmroth who have been an invaluable support in the work of this PhD Thesis. Bo, thank you for always supporting me in my research and letting me take my time. Thank you also for your careful and critical reading of this Thesis and all your constructive and valuable comments (my favorite “gör om, gör rätt” ;-)). The Thesis would not be what it is without you! Bo is co-author of Papers II–IV and is also a major contributor to Paper I, especially formulating the theory and help with literature study.

For my master’s thesis and my first years as a PhD student, the one person I owe the most to is Erik. Thank you for introducing me to this fascinating subject and all the time you have taken to explain and once again explain the world of matrices and matrix pencils. Erik is co-author of Paper II and has also been of great help with proofs and theory in Paper I.

I am also grateful to Pedher Johansson, my good friend next door, who has in many ways been a great coworker in my research. Thank you for all our small chats and discussions (about everything and nothing). Pedher has made it possible for me to test and visualize the results in Part I of this Thesis.

As the second half of my PhD studies took another direction, new people become valuable for my work.

First I want to thank Anton Shiriaev who introduced me to my second topic, which is Part II of the Thesis. You have always new interesting ideas and have been of great importance for the progress of my research. You have also helped me a lot to get some insight into applied systems and control. I am looking forward to a continued collaboration. Anton is co-author of Papers III and IV.

A special thanks to Andras Varga and Sergei Gusev who have been of great help with the writing of Part II. You have always had time to answer my questions and our e-mail discussions have been invaluable for me. Andras is co-author to Papers III and IV, and Sergei is co-author to Paper IV. They are also originators of the methods evaluated in Part II, and Andras has contributed with the last example in Paper II.

Now it is time to thank all present and former colleagues at the Department of Computing Science for all enjoyable moments in the coffee room and challenging battles on the floorball field. Hopefully there are many more to come!

Privately, I first want to thank my parents Gulli and Sven-Olof who at all times have been supporting me and always encouraging me to continue to study. I also want to thank my sister Sofie with family, who through the last three years have been asking “When are you done?”. At last I can answer – now!

To all my friends, especially Henrik L., Marja, Anna, Henrik S., Linda, Lars, who have made the life outside work an enjoyable time, thank you!

Finally, and most, I want to thank Annica and Noah for bringing joy into my life. I am lucky to have you both, I cannot thank you enough!

Financial support has been provided jointly by the Faculty of Science and Technology, Umeå University, and the Swedish Foundation for Strategic Research (SSF) under the frame program grant A3 02:128.

Umeå, February 2009

Stefan Johansson

Contents

CHAPTER 1

Introduction

Today, control systems are common parts of products we are using in our everyday life, like automobiles, DVD players, home heating systems etc. In industrial environments, control systems are even more common, for example, to regulate the temperature in a chemical process or to control an autonomous robot. As these systems become more and more complex, new robust and advanced numerical methods are required that help us to analyze and understand their behavior. In order to develop these methods, the underlying mathematical theory as well as the practical implications have to be well understood.

Since the beginning of the 1970s great advancements have been done in systems and control theory, and the research is today conducted by scientists in a broad range of disciplines, like systems and control theory, computing science, and mathematics. A consequence of this situation is that we see different terminology and notation used for similar representations. This can lead to that new important results are missed and that old unreliable methods are used when there exist new robust methods. Throughout this Thesis an unified terminology is used where the difference between related terminology is highlighted. For this survey, an extensive literature study has been done in papers and books on control theory, mathematics and numerical linear algebra. Systems and control theory is a huge area, so we have chosen to limit our attention to a few related subjects.

For an introduction to systems and control theory there exist several introductory textbooks where the fundamental terminology and notation are explained. For more advanced textbooks that also consider numerical aspects we refer to [6, 35, 58, 97, 98, 99].

Since control systems often are very large and complex, it is desirable or even necessary to approximate the reality with a model. Large systems are often reduced using model reduction and complex systems are represented by a simplified model. In this Thesis, we restrict our discussion to continuous-time linear finite dimensional systems, both time-invariant (in Papers I and II) and periodic time-varying systems (in Papers III and IV). One advantage with this restriction is that these systems can be understood and analyzed using well known theory and existing robust algorithms. Furthermore, nonlinear systems are normally approximated with a sequence of linear systems, and methods for time varying systems are often based on recursive use of time-invariant methods. For infinite dimensional systems (for example arising in partial differential equations), discretization is typically done using a finite elements method to represent the underlying operator. This generates a (typically) large and sparse matrix which now is of finite dimension.

In systems and control theory, we usually consider a system $\mathcal{S}$ that from input signals produce output signals given the current states of the system. Such a system can either be analyzed using a polynomial model

$$D(\lambda)Y(\lambda) = N(\lambda)U(\lambda),$$

which is the classical approach, or by its associated state-space model

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t). \end{aligned}$$

This Thesis focus mainly on the state-space representation, which is more convenient and advisable to use for our purpose. In the next chapter, it will be clear why this is the case and we also explain the relation between those two models.

A state-space system can also be represented and analyzed in terms of a system pencil

$$S - \lambda T = \begin{bmatrix} A & B \\ C & D \end{bmatrix} - \lambda \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}.$$

The system pencil $S - \lambda T$ is a special form of a general matrix pencil $G - \lambda H$, where G and H are arbitrary and unstructured matrices. Both these pencil forms are of great interest and are thoroughly explained and discussed in the next chapter and in Papers I and II.

The Thesis is organized as follows. In Chapter 2, we introduce the reader to systems and control theory, including basic terminology and definitions. Chapter 3 gives a brief introduction to control system design with focus on methods for finding a stabilizing controller for linear control systems. The major contributions of the Thesis are divided into two parts. Part I consists of Papers I and II and is devoted to canonical forms and stratification of orbits and bundles. Part II consists of Papers III and IV and addresses the numerical solution of periodic Riccati differential equations. Brief introductions to the two parts are presented in Chapter 4 and Chapter 5, respectively. In Chapter 6, an overview of important existing software for computer-aided control system design (CACSD) is presented. The main contributions of the Thesis and the papers are summarized in Chapter 7.

CHAPTER 2

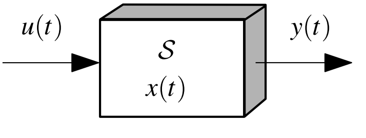
Basic Concepts in Systems and Control Theory

This chapter gives a brief introduction to systems and control theory, and demonstrates how important problems in control systems design and analysis can be expressed and solved in terms of linear algebra. Moreover, different forms of representation of a control system are introduced, and how mathematical tools can be used to manipulate and extract information from them is presented.

2.1 Control systems

The goal with a control system is to achieve a desired behavior of a physical system by use of an appropriate control signal. The physical system under investigation is known as the *plant*, and can for example be a robot-arm which should follow a specified trajectory, a chemical process where an exact temperature must be kept, or a helicopter which should hover (be stabilized) in an upright position. However, for practical reasons the plant can often not be used when developing the control system and they are even frequently developed simultaneously. Instead a *model* of the plant is created, i.e., a mathematical approximation describing the dynamical behavior of the true system. The fact that the controller is developed for the model instead of the plant gives rise to new problems and design issues, e.g., the plant is often very complex and can not be modeled exactly and it can be hard to see if poor results of the controller are caused by the simplified model or the controller itself (what do we actually need to improve?). The control problem and how a control system is designed are further discussed in Chapter 3.

In the following, we consider a system $\mathcal{S}$ that given an input signal $u(t)$ (also called control variable) and a state $x(t)$ produces an output signal $y(t)$:



The system $\mathcal{S}$ can be a model of a plant, a controller, etc. The state and the input and output signals can be composed of several components. In that case, they are given as vectors of length n, m and p, respectively. The system $\mathcal{S}$ is called a *single-input* (SI) system when $m = 1$ and a *multi-input* (MI) system when $m > 1$. Likewise, $\mathcal{S}$ is called a *single-output* (SO) system when $p = 1$ and a *multi-output* (MO) system when $p > 1$. Moreover, the dimension n of $x(t)$ gives the order of $\mathcal{S}$.

2.2 State-space systems

The classical approach in control theory is to analyze the system $\mathcal{S}$ in the frequency domain by using a polynomial model

$$Y(\lambda) = G(\lambda)U(\lambda), \tag{2.1}$$

where $G(\lambda)$[1] is the *transfer function*. As an alternative the system can be analyzed in the time domain by representing the system in *state-space form*. The methods for the polynomial model have the advantage of being faster than the methods for the state-space model, and typically a polynomial model has less free parameters than the corresponding state-space model. However, from a numerical point of view it is more convenient and advisable to use the state-space representation. Moreover, as the systems become larger the difference between the number of free parameters become smaller and the advantage of the polynomial model diminishes.

Let us first consider *linear time-invariant, finite dimensional systems* (LTI systems) which are also the main subject of Papers I and II. Such systems can be described by a linear time-invariant model. Let $\mathbb{K}$ denote either the field $\mathbb{C}$ of complex numbers or the field $\mathbb{R}$ of real numbers. In continuous time, the system is represented as a *state-space model* by a system of differential equations

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t), \end{aligned} \tag{2.2}$$

where $A \in \mathbb{K}^{n\times n}$, $B \in \mathbb{K}^{n\times m}$, $C \in \mathbb{K}^{p\times n}$ and $D \in \mathbb{K}^{p\times m}$ are time-invariant matrices, and $\dot{x}(t)$ is the derivative of x with respect to time t, i.e., $dx(t)/dt$.[2] The state $x(t)$ is in the n-dimensional *state space* represented by a vector whose evolution in time follows a state vector trajectory, see Figure 2.1. By examining this trajectory it is possible to see, e.g., if the system is stable or converges to a periodic oscillating behavior. In discrete time, the state-space model is given by the difference equations

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k + Du_k. \end{aligned} \tag{2.3}$$

[1] In literature on control theory, instead of the symbol λ the complex variable s is often used in continuous time and the variable z in discrete time. However, we use the λ notation which is more common in numerical linear algebra.

[2] In some literature, the operator λ is used to represent the differential operator $\frac{d}{dt}$. Consequently, $\dot{x}(t)$ is then expressed as $\lambda x(t)$.

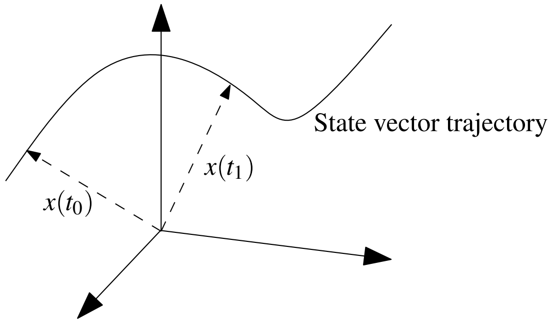


FIGURE 2.1: An example of a three dimensional state space, where $x(t_0)$ and $x(t_1)$ are the state-vectors at time t_0 and t_1, respectively.

In the following, we only discuss the continuous-time case. The corresponding block diagram of the state-space system (2.2) is shown in Figure 2.2, where A is the *system* (*state*) *matrix*, B the *input* (*control*) *matrix*, C the *output matrix*, and D is the *feedforward matrix*.

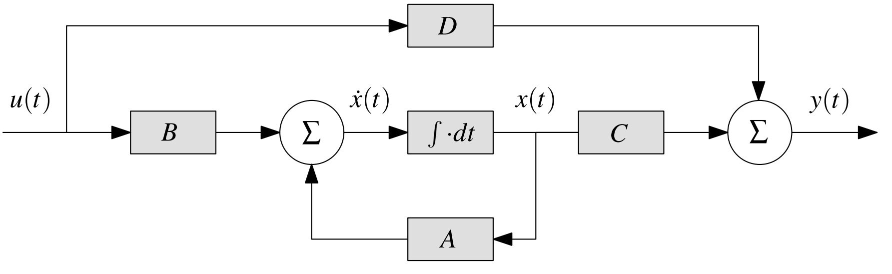


FIGURE 2.2: Block diagram of a linear time-invariant system in continuous time.

We also consider the *generalized state-space system* (or *descriptor system*)

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t), \end{aligned} \tag{2.4}$$

where A and E not necessarily have to be square. In that case, $E, A \in \mathbb{K}^{q\times n}$ and $B \in \mathbb{K}^{q\times m}$. However, in most cases we only consider square systems where $q = n$. If E is singular, (2.4) is a descriptor system of *index* ν, where ν is the size of the largest Jordan block corresponding to the infinite eigenvalue of $\lambda E - A$ (see also Paper I). The system and the associated matrix pencil $\lambda E - A$ are said to be *regular* if $\det \lambda E - A \neq 0$. If E is nonsingular, the generalized state-space system can be transformed into the state-space form (2.2). By convention, the system is in this case said to be of index zero.

The state-space system (2.2) is in short form represented by a *quadruple of matrices* denoted (A,B,C,D) and the generalized state-space system (2.4) is represented by the 5-tuple (E,A,B,C,D). Often we are only considering parts of (2.2). These are pairs and triples of matrices, denoted (A,B), (A,C) and (A,B,C), associated with the following equations

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{2.5}$$

$$\begin{aligned} \dot{x}(t) &= Ax(t), \\ y(t) &= Cx(t), \end{aligned} \tag{2.6}$$

and

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t), \end{aligned} \tag{2.7}$$

respectively. These systems also appear in generalized versions with the matrix E as in (2.4).

Linear time-varying systems (LTV systems) are also considered. For a time-varying system the matrices A, B, C and D are varying over time, in opposite to a time-invariant system where the matrices have constant entries. An LTV system where the matrices also are periodic with periodicity T is called a T-periodic system. Such systems are the subject of Papers III and IV and have the state-space model

$$\begin{aligned} \dot{x}(t) &= A(t)x(t) + B(t)u(t), \\ y(t) &= C(t)x(t) + D(t)u(t), \end{aligned} \tag{2.8}$$

where $A(t) \in \mathbb{K}^{n\times n}$, $B(t) \in \mathbb{K}^{n\times m}$, $C(t) \in \mathbb{K}^{p\times n}$ and $D(t) \in \mathbb{K}^{p\times m}$ are T-periodic matrices, i.e., $A(t) = A(t+T)$, $B(t) = B(t+T)$, $C(t) = C(t+T)$ and $D(t) = D(t+T)$ for all $t \geq 0$.

2.3 Pencil representation

The set of matrices of the form $G - \lambda H$ with $\lambda \in \mathbb{C}$ corresponds to a general *matrix pencil* [54, 56], where the two complex matrices G and H are of size $m_{\mathrm{p}} \times n_{\mathrm{p}}$. If G and H are square, then $(G - \lambda H)v = 0$ defines the generalized eigenvalue problem. The scalars λ and nonzero vectors v which satisfy $Gv = \lambda Hv$ are the generalized eigenvalues and their corresponding generalized eigenvectors of the matrix pencil. All matrix pencils with rectangular G and H or $\det(G - \lambda H) \equiv 0$ (for all λ) are singular, which is the case in most control applications.

A system $\mathcal{S}$ can also be represented and analyzed in terms of a matrix pencil, which in this special form is called a *system pencil*, $\mathbf{S}(\lambda)$. In contrary to a general matrix pencil, a system pencil emphasizes the structure of the system. When we do not want to make any distinction between a matrix pencil or a system pencil, we simply denote it a pencil. The associated system pencil for the generalized state-space system (2.4) is

$$\mathbf{S}(\lambda) = G - \lambda H = \begin{bmatrix} A & B \\ C & D \end{bmatrix} - \lambda \begin{bmatrix} E & 0 \\ 0 & 0 \end{bmatrix}, \tag{2.9}$$

where G and H are of size $(n+p) \times (n+m)$. For the standard state-space system (2.2) the associated system pencil is

$$\mathbf{S}(\lambda) = \begin{bmatrix} A & B \\ C & D \end{bmatrix} - \lambda \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}. \tag{2.10}$$

2.4 Controllability and observability

In the case of designing a controller for a system, the concept of controllability plays an important role. The controllability of a continuous-time system is defined as follows[3].

> A linear control system is said to be *controllable* if there exists an input signal $u(t)$, $t_0 \leq t \leq t_f$, that takes every state variable from an initial state $x(t_0)$ to a desired final state $x(t_f)$ in finite time. Otherwise it is said to be *uncontrollable*.

The classical algebraic approach to determine if a system $\mathcal{S}$ is controllable is to form the controllability matrix and determine its rank. Given the matrix pair (A,B) of a state-space system with n states, the system is controllable if the *controllability matrix*

$$\mathbf{C}(A,B) = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}, \tag{2.11}$$

of size $n \times nm$ is of rank n. For a single-input system it is analogous to check if $\mathbf{C}(A,B)$ is nonsingular. This method, however, is not recommended because powers of A must be computed, which can result in a significant build-up of rounding errors [96]. Since the controllability properties of the system only depend on the matrix pair (A,B), it is referred to as the *controllability pair* with the corresponding system pencil

$$\mathbf{S}_\mathrm{C}(\lambda) = \begin{bmatrix} A & B \end{bmatrix} - \lambda \begin{bmatrix} I_n & 0 \end{bmatrix}. \tag{2.12}$$

Another approach to determine the controllability of a system is to check if $\mathbf{S}_\mathrm{C}(\lambda)$ has any eigenvalues. If so, the eigenvalues correspond to the *uncontrollable modes* (*eigenvalues*) of the system. Methods based on such an approach are however not always reliable, especially if the eigenvalues are sensitive to small perturbations. A

[3] For continuous-time systems the concept of reachability coincides with that of controllability. That is however not the case for discrete-time systems. It would be more appropriate to always use the term reachability, because that is what normally is meant for both types of systems. However, we use the term controllability because that is more common. [106]

more robust approach is to perform a staircase reduction of the controllability pair (A,B) to the so called controllability staircase form [24, 37, 76, 96, 104, 105, 107], see also Paper I. This method has the advantage that neither the eigenvalues nor the powers of A have to be computed. Instead, the rank is revealed directly from the submatrices in the staircase form.

An even more robust method is to compute the distance from a controllable system to the nearest uncontrollable by converting the rank test to a distance problem. For a controllable pair (A,B), the distance to uncontrollability [96] is defined as the smallest perturbation of the system matrices that make the system uncontrollable:

$$\tau(A,B) = \min\left\{ \| \begin{bmatrix}\Delta A & \Delta B\end{bmatrix} \| : (A+\Delta A, B+\Delta B) \text{ is uncontrollable} \right\},$$

where $\|\cdot\|$ denotes the 2-norm or Frobenius norm. Equivalently,

$$\tau(A,B) = \inf_{\lambda\in\mathbb{C}} \sigma_{\min}\left(\begin{bmatrix}A-\lambda I & B\end{bmatrix}\right),$$

where $\sigma_{\min}(X)$ denotes the smallest singular value of $X \in \mathbb{C}^{n\times(n+m)}$ [45]. The problem to determine the distance to uncontrollability has been studied by several authors, see for example [25, 34, 36, 45, 59, 66, 96] and recent publications [30, 42, 67, 91, 78].

The dual concept of controllability is the observability of a system, which is defined as follows.

> A system is said to be *observable* if it is possible to find the initial state $x(t_0)$ from the input signal $u(t)$ and the output signal $y(t)$ measured over a finite interval of time $t_0 \le t \le t_f$. Otherwise it is said to be *unobservable*.

Given the matrix pair (A,C) the system is observable if the *observability matrix*

$$\mathbf{O}(A,C) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}, \tag{2.13}$$

of size $np \times n$ is of rank n. The matrix pair (A,C) is known as the *observability pair* with the corresponding system pencil

$$\mathbf{S}_{\mathrm{O}}(\lambda) = \begin{bmatrix} A \\ C \end{bmatrix} - \lambda \begin{bmatrix} I_n \\ 0 \end{bmatrix}. \tag{2.14}$$

The eigenvalues of $\mathbf{S}_{\mathrm{O}}(\lambda)$ correspond to the *unobservable modes* (*eigenvalues*) of the system. It follows, if $\mathbf{S}_{\mathrm{O}}(\lambda)$ has no eigenvalues the system is observable. As for controllability, a more robust approach is to perform a reduction of the observability pair (A,C) to the observability staircase form, which is the dual form of the controllability staircase form.

If the pair (A,B) is controllable and the pair (A,C) is observable, then the system is said to be *minimal* (or *irreducible*). A state-space model that is reduced to be both controllable and observable is called a *minimal realization*, i.e., it has the minimal number of states necessary for representing its complete behavior. Notably, a system is generically minimal.

EXAMPLE 1

Consider a SISO system with the state-space model

$$\dot{x}(t) = \begin{bmatrix} 1 & -2 \\ 0 & -3 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ -8 \end{bmatrix} u(t),$$
$$y(t) = \begin{bmatrix} -1 & 0.5 \end{bmatrix} x(t) + u(t).$$

By computing the ranks of the controllability and observability matrices of the system we get

$$\operatorname{rank}\left(\begin{bmatrix} B & AB \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} 0 & 16 \\ -8 & 24 \end{bmatrix}\right) = 2,$$

and

$$\operatorname{rank}\left(\begin{bmatrix} C \\ CA \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} -1 & 0.5 \\ -1 & 0.5 \end{bmatrix}\right) = 1,$$

i.e., the system is controllable but not observable (it has one unobservable mode).

Since this SISO system has distinct eigenvalues it can be transformed such that the system matrix A becomes diagonal, where each diagonal element corresponds to a mode of the system:

$$\begin{aligned} \frac{d\widetilde{x}(t)}{dt} &= \begin{bmatrix} 1 & 0 \\ 0 & -3 \end{bmatrix} \widetilde{x}(t) + \begin{bmatrix} 2 \\ -4 \end{bmatrix} u(t), \\ y(t) &= \begin{bmatrix} -2 & 0 \end{bmatrix} \widetilde{x}(t) + u(t). \end{aligned} \tag{2.15}$$

The system can now be represented by the block diagram in Figure 2.3. It is easy to see that state $\widetilde{x}_1$ is both controllable and observable but $\widetilde{x}_2$ is unobservable (we cannot observe the state $\widetilde{x}_2$ from the output). From the diagonal form of the system matrix we get the unobservable mode as the one corresponding to the zero element in the output matrix $\widetilde{C} = \begin{bmatrix} -2 & 0 \end{bmatrix}$, i.e., the unobservable mode is -3. If the system has had any uncontrollable mode, the transformed input matrix $\widetilde{B} = \begin{bmatrix} 2 & -4 \end{bmatrix}^T$ would have had at least one zero element.

By eliminating the unobservable mode we get the minimal realization of the state-space system given by the following SISO system of order one (computed with the MATLAB function minreal):

$$\dot{x}(t) = x(t) - 3.578u(t),$$
$$y(t) = 1.118x(t) + u(t).$$

The minimal realization can also be seen directly from the diagonal form (2.15):

$$\dot{x}(t) = x(t) + 2u(t),$$
$$y(t) = -2x(t) + u(t).$$

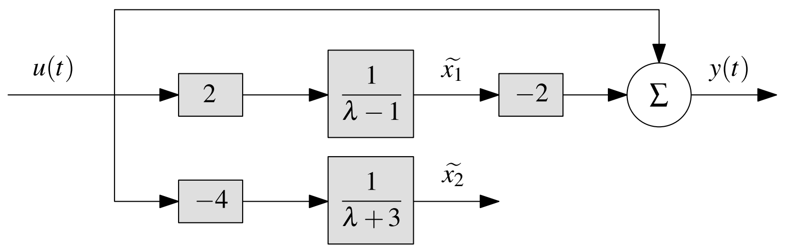


FIGURE 2.3: Block diagram of a SISO system of order two with one unobservable mode.

2.5 Decompositions of state-space systems

State-space systems can be transformed into different canonical forms. Here we give examples of some that are of great interest for understanding and establishing theoretical results for state-space systems. For more details and numerical aspects see Paper I.

For linear systems, the controllability and observability properties remain invariant under certain transformations. One of the most important is the linear transformation on the state vector, defined as

$$\widetilde{x}(t) = Px(t),$$

where P is a nonsingular matrix. The transformed state-space model is

$$\frac{d\widetilde{x}(t)}{dt} = \widetilde{A}\widetilde{x}(t) + \widetilde{B}u(t),$$
$$y(t) = \widetilde{C}\widetilde{x}(t) + Du(t),$$

where

$$\widetilde{A} = PAP^{-1}, \quad \widetilde{B} = PB, \text{ and } \quad \widetilde{C} = CP^{-1}.$$

The transformation on the state matrix A is known as a *similarity transformation* and the matrices A and $\widetilde{A}$ are *similar*. We can now see that the state of a system is not uniquely defined, i.e., there exist infinite number of equivalent state-space models of a systems. This can be utilized by choosing a P such that the transformed state-space model gets a special structure, e.g., where the controllable part is separated from the uncontrollable part. In the following, we use unitary transformation matrices for complex-valued state-space systems and orthogonal matrices for real-valued state-space systems.

Suppose the controllability pair (A,B) is not controllable, then the system can be decomposed into controllable and uncontrollable parts.

Theorem 2.5.1 [35] *Consider a controllability pair (A,B) with $\operatorname{rank}(\mathbf{C}(A,B)) = k < n$. Then there exists a unitary matrix U (or an orthogonal matrix if a real-valued state-space system) such that*

$$\widetilde{A} = UAU^H = \begin{bmatrix} \widetilde{A}_c & \widetilde{A}_{12} \\ 0 & \widetilde{A}_{\bar{c}} \end{bmatrix}, \qquad \widetilde{B} = UB = \begin{bmatrix} \widetilde{B}_c \\ 0 \end{bmatrix},$$

where $\widetilde{A}_c \in \mathbb{C}^{k\times k}$, $\widetilde{A}_{\bar{c}} \in \mathbb{C}^{n-k\times n-k}$, $\widetilde{B}_c \in \mathbb{C}^{k\times m}$, and $(\widetilde{A}_c, \widetilde{B}_c)$ is controllable.

The transformed state-space model is

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} \widetilde{x}_c(t) \\ \widetilde{x}_{\bar{c}}(t) \end{bmatrix} &= \begin{bmatrix} \widetilde{A}_c & \widetilde{A}_{12} \\ 0 & \widetilde{A}_{\bar{c}} \end{bmatrix} \begin{bmatrix} \widetilde{x}_c(t) \\ \widetilde{x}_{\bar{c}}(t) \end{bmatrix} + \begin{bmatrix} \widetilde{B}_c \\ 0 \end{bmatrix} u(t), \\ y(t) &= \begin{bmatrix} \widetilde{C}_c & \widetilde{C}_{\bar{c}} \end{bmatrix} \begin{bmatrix} \widetilde{x}_c(t) \\ \widetilde{x}_{\bar{c}}(t) \end{bmatrix} + Du(t), \end{aligned} \tag{2.16}$$

which is represented by the block diagram in Figure 2.4.

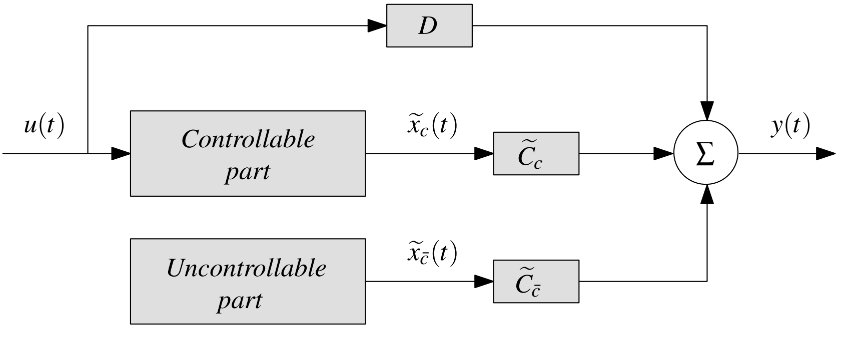


FIGURE 2.4: Decomposition of controllable-uncontrollable parts.

Dually, an unobservable system can be decomposed into observable and unobservable parts.

Theorem 2.5.2 [35] *Consider an observability pair* (A,C) *with* $\text{rank}(\mathbf{O}(A,C)) = k < n$. *Then there exists a unitary matrix U (or an orthogonal matrix if a real-valued state-space system) such that*

$$\widetilde{A} = UAU^H = \begin{bmatrix} \widetilde{A}_o & 0 \\ \widetilde{A}_{21} & \widetilde{A}_{\bar{o}} \end{bmatrix}, \qquad \widetilde{C} = CU^H = \begin{bmatrix} \widetilde{C}_o & 0 \end{bmatrix},$$

where $\widetilde{A}_o \in \mathbb{C}^{k\times k}$, $\widetilde{A}_{\bar{o}} \in \mathbb{C}^{n-k\times n-k}$, $\widetilde{C_o} \in \mathbb{C}^{p\times k}$, *and* $(\widetilde{A}_o, \widetilde{C}_o)$ *is observable.*

Combining Theorems 2.5.1 and 2.5.2 results in what is known as the *Kalman canonical decomposition*, e.g., see [35]. However, in general this decomposition can not be achieved by unitary (or orthogonal) transformation matrices. The transformed state-space model is represented by the block diagram in Figure 2.5.

Theorem 2.5.3 *Given the state-space system (2.2) there always exists a transformation* $\widetilde{x}(t) = Tx(t)$ *with a nonsingular matrix T such that*

$$\frac{d}{dt}\begin{bmatrix} \widetilde{x}_{c\bar{o}}(t) \\ \widetilde{x}_{co}(t) \\ \widetilde{x}_{\bar{c}\bar{o}}(t) \\ \widetilde{x}_{\bar{c}o}(t) \end{bmatrix} = \begin{bmatrix} \widetilde{A}_{c\bar{o}} & \widetilde{A}_{12} & \widetilde{A}_{13} & \widetilde{A}_{14} \\ 0 & \widetilde{A}_{co} & 0 & \widetilde{A}_{24} \\ 0 & 0 & \widetilde{A}_{\bar{c}\bar{o}} & \widetilde{A}_{34} \\ 0 & 0 & 0 & \widetilde{A}_{\bar{c}o} \end{bmatrix} \begin{bmatrix} \widetilde{x}_{c\bar{o}} \\ \widetilde{x}_{co} \\ \widetilde{x}_{\bar{c}\bar{o}} \\ \widetilde{x}_{\bar{c}o} \end{bmatrix} + \begin{bmatrix} \widetilde{B}_{c\bar{o}} \\ \widetilde{B}_{co} \\ 0 \\ 0 \end{bmatrix} u(t),$$

$$y(t) = \begin{bmatrix} 0 & \widetilde{C}_{co} & 0 & \widetilde{C}_{\bar{c}o} \end{bmatrix} \begin{bmatrix} \widetilde{x}_{c\bar{o}} \\ \widetilde{x}_{co} \\ \widetilde{x}_{\bar{c}\bar{o}} \\ \widetilde{x}_{\bar{c}o} \end{bmatrix} + Du(t),$$

where

$\widetilde{x}_{c\bar{o}} \equiv$ *controllable but not observable states*,
$\widetilde{x}_{co} \equiv$ *controllable and observable states*,
$\widetilde{x}_{\bar{c}\bar{o}} \equiv$ *neither controllable nor observable states*,
$\widetilde{x}_{\bar{c}o} \equiv$ *observable but not controllable states*.

Moreover, the minimal realization of the state-space system (2.2) is given by the subsystem $(\widetilde{A}_{co}, \widetilde{B}_{co}, \widetilde{C}_{co}, D)$ *which is both controllable and observable.*

2.6 Poles, zeros and stability

As mentioned in Section 2.2, a state-space system can be expressed by the polynomial model (2.1) and analyzed using its transfer function $G(\lambda)$. Given the state-space model

$$\dot{x}(t) = Ax(t) + Bu(t),$$
$$y(t) = Cx(t) + Du(t),$$

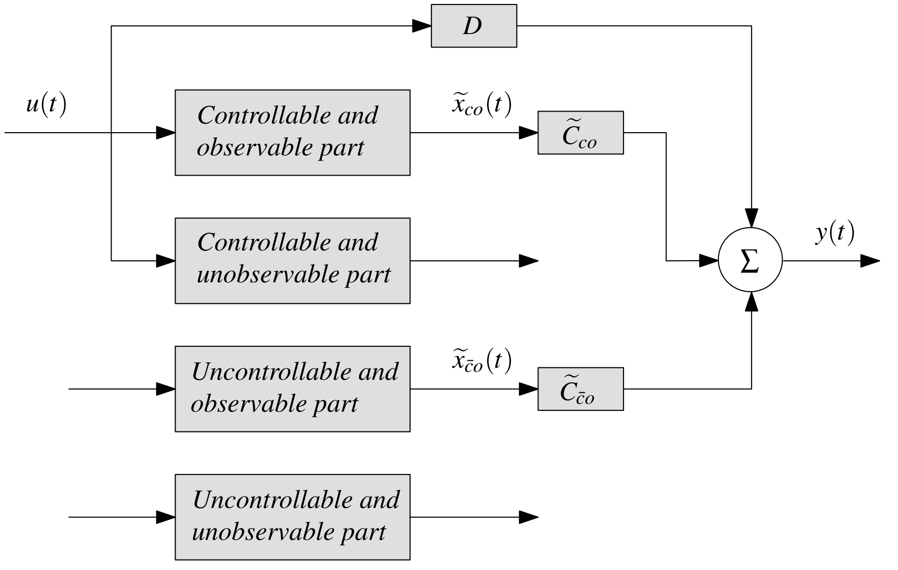


FIGURE 2.5: Block diagram of the Kalman canonical decomposition.

we take the Laplace transform (with initial condition $x(0)$):

$$\lambda X(\lambda) - x(0) = AX(\lambda) + BU(\lambda),$$
$$Y(\lambda) = CX(\lambda) + DU(\lambda).$$

This results in the polynomial models

$$\begin{aligned} X(\lambda) &= R(\lambda)x(0) + R(\lambda)BU(\lambda), \\ Y(\lambda) &= CR(\lambda)x(0) + \big(CR(\lambda)B + D\big)U(\lambda), \end{aligned} \tag{2.17}$$

where $R(\lambda)$ is the resolvent:

$$R(\lambda) = (\lambda I - A)^{-1}.$$

With zero initial condition, $x(0) = 0$, the output $Y(\lambda)$ is related to the transfer function $G(\lambda)$ as:

$$\begin{aligned} Y(\lambda) &= G(\lambda)U(\lambda), \quad \text{where} \\ G(\lambda) &= C(\lambda I - A)^{-1}B + D. \end{aligned} \tag{2.18}$$

The transfer function $G(\lambda)$ is a rational matrix of dimension $p \times m$, where the (i, j)th entry denote the transfer function from the jth input to the ith output. Moreover, the transfer function is always both controllable and observable and from Theorem 2.5.3 we have that

$$G(\lambda) = \widetilde{C}_{co}(\lambda I - \widetilde{A}_{co})^{-1}\widetilde{B}_{co} + D.$$

The opposite, to go from a transfer function $G(\lambda)$ to a state-space system (A,B,C,D), is called a *state-space realization*. A state-space realization is not unique, there can exist several state-space models for the same transfer function. The most important one is the minimal realization from Section 2.4, which is the state-space realization (A,B,C,D) of $G(\lambda)$ with the smallest possible dimension of the matrix A. It follows that a state-space realization (A,B,C,D) of $G(\lambda)$ is minimal if and only if (A,B) is controllable and (A,C) is observable.

The rational matrix $G(\lambda)$ can also be represented in polynomial matrix fraction form as

$$G(\lambda) = D(\lambda)^{-1}N(\lambda), \tag{2.19}$$

where the nominator $N(\lambda)$ is a polynomial matrix and the denominator $D(\lambda)$ is a non-singular polynomial matrix. This is an extension of the SISO systems where $G(\lambda)$ is a rational function, with scalar polynomials $D(\lambda)$ and $N(\lambda)$ of degrees n and m, respectively. From this representation some important terms are defined.

- The *poles* are the roots of $D(\lambda)$.
- The *zeros* are the roots of $N(\lambda)$.
- The model is *strictly proper* if $m < n$, or equivalently $G(\infty) = 0$.
- The model is *proper* if $m \le n$, or equivalently $G(\infty)$ is zero or nonzero constant.
- The model is *biproper* if $m = n$, or equivalently $G(\infty)$ is nonzero constant.
- The model is *improper* if $m > n$, or equivalently $|G(\infty)| = \infty$.

For MIMO systems the above definition of poles and zeros fails when there are coalescing poles and zeros, e.g., when the state-space model is not a minimal realization. Moreover, the definition does not give any detailed information about the multiplicity of the poles and zeros. A more appropriate method is based on computing the eigenvalues and the generalized eigenvalues.

Theorem 2.6.1 [48, 99, 118] *Given a transfer function $G(\lambda)$ and its corresponding matrix quadruple (A,B,C,D), the* poles *of $G(\lambda)$ are the eigenvalues of the system matrix A and the* zeros *of $G(\lambda)$ are the generalized eigenvalues of the system pencil*

$$\mathbf{S}(\lambda) = \begin{bmatrix} \lambda I - A & B \\ -C & D \end{bmatrix}.$$

The above method to compute the zeros can only be applied straightforwardly if (A,B,C,D) is a minimal realization of $G(\lambda)$. Therefore, the system pencil $\mathbf{S}(\lambda)$ is transformed into a staircase-type form by unitary (orthogonal) matrices [48, 104], from which also the left and right minimal indices and the infinite zeros are revealed. This method is further discussed in Paper I.

Let the system pencil $\mathbf{S}(\lambda)$ be associated with a system $\mathcal{S}$ with the controllability system pencil $\mathbf{S}_\mathrm{C}(\lambda)$ and the observability system pencil $\mathbf{S}_\mathrm{O}(\lambda)$. Then the following types of zeros are defined for $\mathcal{S}$.

Definition 2.6.1 [99] *The zeros of* $\mathbf{S}_C(\lambda)$ *are called the* input decoupling zeros *of the system* $\mathcal{S}$*, and the zeros of* $\mathbf{S}_O(\lambda)$ *are called the* output decoupling zeros *of the system* $\mathcal{S}$*. If the system* $\mathcal{S}$ *has no input and no output decoupling zeros, then the zeros of the system pencil* $\mathbf{S}(\lambda)$ *are called the* transmission zeros *of the system.*

We remark that the input decoupling zeros are the uncontrollable modes of (A,B), and the output decoupling zeros are the unobservable modes of (A,C). It follows that, if the system is minimal the zeros of $\mathbf{S}(\lambda)$ are transmission zeros.

Knowing the poles we can also analyze the stability of the system $\mathcal{S}$. In the literature, there exist more than one definition of stability. We have chosen to use the following definition which is also called *asymptotic stability*. The system

$$\dot{x}(t) = Ax(t), \quad x(0) = x_0,$$

is said to be *stable* if $x(t) \to 0$ as $t \to \infty$ for every x_0. An important property is that the stability of $\mathcal{S}$ can be determined from the eigenvalues of the system matrix.

Theorem 2.6.2 [98] *A continuous-time system* $\mathcal{S}$ *is* stable if and only if *all eigenvalues* λ_k *of the system matrix A are in the open left-half of the complex plane:*

$$\mathrm{Re}(\lambda_k) < 0, \quad k = 1, 2, \ldots, n.$$

The eigenvalues of A with negative real parts are called *stable eigenvalues* and a matrix A with only stable eigenvalues is called a *stable matrix* (also known as a *Hurwitz* matrix). The distance $\min\{-\mathrm{Re}(\lambda_k) : k = 1, \ldots, n\}$ to the imaginary axis is called the *stability margin* of A. However, it is not always sufficient to compute the eigenvalues to determine if the system i stable. A more robust approach is to compute the *distance to instability* [31, 109, 91]. Let A be stable matrix, then the distance $\beta(A)$ from the set of unstable matrices is defined as

$$\beta(A) = \min\{\|\Delta A\| : A + \Delta A \in \mathcal{U}\},$$

where $\mathcal{U}$ is the set of $n \times n$ matrices with at least one eigenvalue on the imaginary axis, and $\|\cdot\|$ denotes the 2-norm or the Frobenius norm.

EXAMPLE 2

The corresponding transfer function for the state-space system in Example 1 is computed from Equation (2.18) as

$$G(\lambda) = \begin{bmatrix} -1 & 0.5 \end{bmatrix} \left(\lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & -2 \\ 0 & -3 \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 \\ -8 \end{bmatrix} + 1 = \frac{\lambda - 5}{\lambda - 1}.$$

Since the system is not completely observable the system has a common pole and zero and the transfer function is of order one less than the state-space model, i.e., the state-space system is not a minimal realization. This can also be seen from computing the poles and zeros directly from the eigenvalues of the state-space model, which give the poles -3 and 1 and the zeros 5 and -3. From these we get the transfer function

$$G(\lambda) = D(\lambda)^{-1}N(\lambda) = \frac{\lambda^2 - 2\lambda - 15}{\lambda^2 + 2\lambda - 3} = \frac{(\lambda - 5)(\lambda + 3)}{(\lambda - 1)(\lambda + 3)}.$$

Moreover, the system is not stable because the real part of one of the poles (eigenvalue 1) is greater than zero.

2.7 Robustness

The plant (the physical system) is often very complex and can only be modeled by an approximation of its true parameters, e.g., due to uncertainty in the plant, and round-off and measurement errors. If it is a large scale system the order of the system also need to be reduced by model reduction [5, 13, 18, 95], where the number of states is reduced such that the new system is close, in some sence, to the original system. Therefore, a state-space model is usually only an approximation of the true system. On top of that, computing the properties described in the previous sections (controllability, observability and stability) is an ill-posed problem. Nevertheless, it is desired that the performance of the controller developed for the model of the plant should be the same when applied to the true plant. Consequently, it is required that the properties of the model are similar to properties of the plant and that they do not change by small perturbations. This is examined by the so called *robustness* of the system.

Two robustness issues are of special interest. These are *robust stability* and *robust controllability*:

- A system is said to be *robustly stable* when both the model and plant are controlled by the same controller and the stability is retained. Equivalently, if the system considered and all nearby systems in a given neighborhood are stable, the system is called robustly stable.

- A system is said to be *robustly controllable* when both the model and plant are controlled by the same controller and the controllability is retained. Equivalently, if the system considered and all nearby systems in a given neighborhood are controllable, the system is called robustly controllable.

As indicated in previous sections, robust stability and robust controllability can be measured by computing the distance to instability and uncontrollability, respectively. Moreover, robust stability can further be examined by studying the pseudospectra of the system matrix A. For a detailed description of robustness issues see for example [91, 58].

CHAPTER 3
Control System Design

The aim of this chapter is to introduce the reader to basic control system design for continuous-time linear systems. We begin in Section 3.1 to define the control problem and look at different design approaches and solutions to the problem.

In Sections 3.2–3.4, we consider different methods to find a stabilizing controller for a linear control system. The basic problem is to find a feedback matrix K such that $A - BK$ is stable. This is known as *state-feedback stabilization*. Often in practise, the eigenvalues of $A - BK$ must meet some criteria, i.e., we want to find a matrix K such that the spectrum of $A - BK$ is in a specified region. This is known as the *pole placement problem* or the *eigenvalue assignment problem* (EVA problem). However, there exist no rules or guidelines where to place the poles (eigenvalues); *"A designer has to use his or her own intuition of how to use the freedom of choosing the eigenvalues to achieve the design objective"* [35]. Instead the problem is often solved via linear optimal control, where the problem is turned into an optimization problem. The solution is found by solving the *linear quadratic optimization problem*, or more commonly known as the *linear quadratic regulator* (LQR) problem.

Other related methods which we do not consider here are H_∞- and H_2-control, and the *linear quadratic Gaussian* (LQG) problem which deals with stochastic systems (systems with noise). Moreover, the majority of the above methods require that the states $x(t)$ are known. However, frequently only the input $u(t)$ and output $y(t)$ of the plant are available for input to the controller. In this case, a state estimator can be constructed which given $u(t)$ and $y(t)$ produce an estimate $\hat{x}(t)$ of the true states.

3.1 The control problem

As we mentioned in Section 2.1, the goal of a control system is to achieve a desired behavior of a plant by an appropriate control signal. The desired behavior is realized by minimizing the difference $e(t)$ between the desired behavior in form of a *reference signal* $r(t)$ and the true output signal $y(t)$ from the plant, see Figure 3.1. A more formal definition of *the fundamental control problem* is given in [58].

Definition 3.1.1 [58] *The central problem in control is to find a technically feasible way to act on a given process so that the process adheres, as closely as possible to some desired behavior. Furthermore, this approximate behavior should be achieved* *in the face of uncertainty of the process and in the presence of uncontrollable external disturbances acting on the process.*

The control problem can further be categorized into two classes:

The *regulator problem*. The task of the controller is to provide an input $u(t)$ such that the output $y(t)$ of the plant is kept constant despite external disturbances. The reference signal $r(t)$ is considered constant. Examples of such control problems are temperature regulators, speed control for cars, and motor speed control for CD players.

The *tracking* (or *servo*) *problem*. The task of the controller is to provide an input $u(t)$ such that the output $y(t)$ of the plant tracks a variable reference signal $r(t)$. The reference signal can be a polynomial, a function of time or the response of another plant. Examples of such control problems are industrial robots, radar antenna tracking an aircraft, and tracking control of the optical head in a CD player.

When investigating the dynamic behavior of a control system, the so called *step response* is of particular interest. The step response is the output from the transfer function

$$Y(\lambda) = G(\lambda)U(\lambda) = G(\lambda)\frac{1}{\lambda},$$

where $U(\lambda)$ is the Laplace transform of the unit step

$$\sigma(t) = \begin{cases} 1, & \text{if } t \geq 0, \\ 0, & \text{if } t < 0. \end{cases} \tag{3.1}$$

This is the most common test signal for control systems, since it has a lot in common with practical problems.

The steady-state response (if it exists) for a unit step is given by

$$\lim_{t\to\infty} y(t) = y_\infty = \lim_{\lambda\to 0} \lambda G(\lambda)\frac{1}{\lambda} = G(0).$$

If the system is stable, the transient part of the step response will decay exponentially to zero, and the steady-state y_∞ exists. How fast the step response converges to the steady-state is determined by the real part of the largest pole (eigenvalue of A). A stable pole close to the stability boundary causes a slow convergence and therefore is referred to as a *dominant* or *slow pole*. A stable pole far from the stability region is called a *fast pole*. Consider the step response for a stable system with output $y(t)$ in Figure 3.1, where the reference signal $r(t)$ is the response of the unit step (3.1). The following parameters define the dynamics of the system.

Output signal, $y(t)$**:** In this case, the step response of a stable system.

Reference signal, $r(t)$**:** In this case, the step response of the unit step $\sigma(t)$ in (3.1).

Error, $e(t)$: The difference between the output $y(t)$ and the reference $r(t)$.

Steady-state, y_∞: The final value of the step response; $\lim_{t\to\infty} y(t) = y_\infty$.

Rise time, t_r: The time it takes for the step response to reach (for the first time) 90%[1] of its final value.

Settling time, t_s: The time it takes for the step response to reach the point when it does not go outside the specified boundary $y_\infty - \delta < y(t) < y_\infty + \delta$.

Overshoot, M_p: The maximum instantaneous amount the step response exceeds y_∞ (usually expressed in percentage of y_∞). M_p occurs at time t_p.

Undershoot, M_u: The maximum instantaneous amount the step response falls below zero. M_u occurs at time t_u.

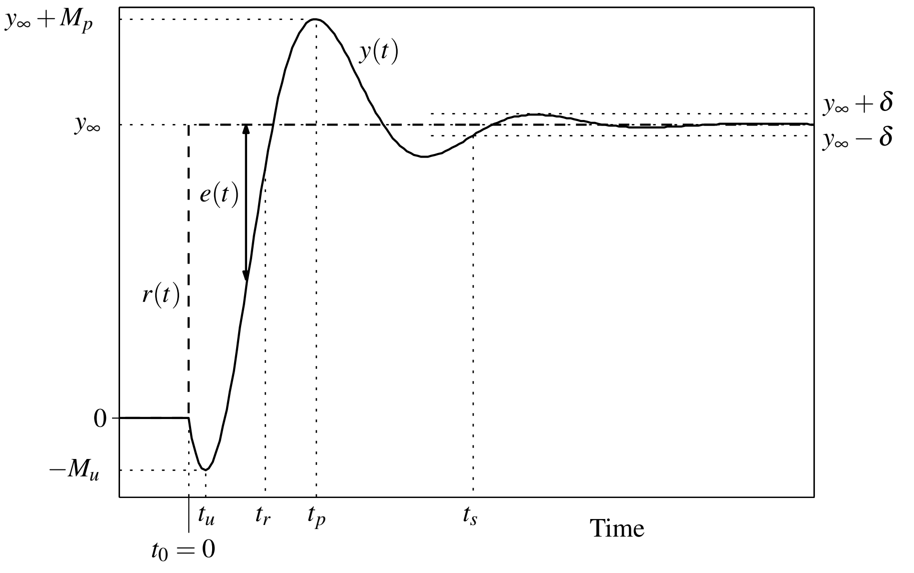


FIGURE 3.1: Step response of a stable system.

When designing a controller for a plant there are mainly two models that are used: the *open-loop controller* and the *closed-loop controller*. Both designs have their own pros and cons. Which one to use depends on the prerequisites and what the designer wants to achieve.

[1] This limit can vary. It is usually chosen to a value between 90% and 100%.

3.1.1 Open-loop control

The controller of an open-loop system is composed of a feedback gain and the model of the plant, see Figure 3.2. As we can see, for this design the control signal $u(t)$ is independent of the output from the plant and any disturbances in the system. This leads to some important properties that must be fulfilled to make the solution satisfactory [58].

- The model must be a very good approximation of the plant.
- The model and its inverse must be stable.
- Disturbances are negligible.
- Initial conditions are zero or known.

One advantage of this design is that it can be built without knowing the output from the plant.

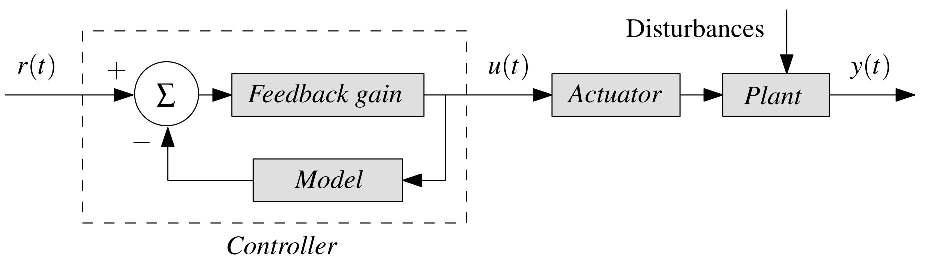


FIGURE 3.2: Open-loop system.

3.1.2 Closed-loop control

The drawbacks for the open-loop controller motivate a redesign of the control system. Instead of having the controller around the model, it is placed around the plant itself. This is called a closed-loop controller and is illustrated in Figure 3.3. One benefit of this design is that the controller has the ability to adapt to the output from the plant. The drawback is that the controller is relying on the sensors and the noise applied to it, since it is the measurement from the sensors that are fed back to the controller rather than the true output.

3.2 State-feedback stabilization

Consider the state-space model (2.2)

$$\dot{x}(t) = Ax(t) + Bu(t),$$
$$y(t) = Cx(t) + Du(t),$$

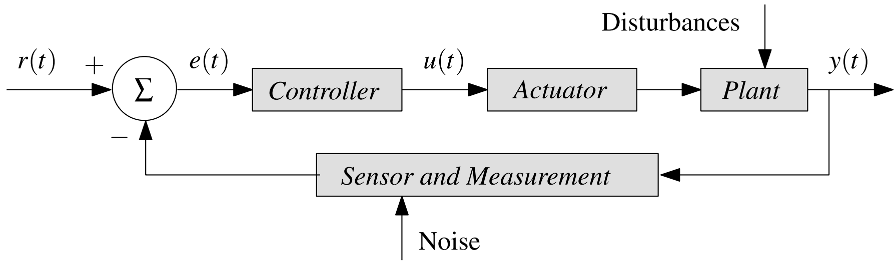


FIGURE 3.3: Closed-loop system.

where the state-vector $x(t)$ is known. Let us now replace the input signal $u(t)$ with the state-variable feedback

$$u(t) = r(t) - Kx(t),$$

where $r(t)$ is a reference signal and K a constant matrix. The new system

$$\begin{aligned} \dot{x}(t) &= (A - BK)x(t) + Br(t), \\ y(t) &= (C - DK)x(t) + Dr(t), \end{aligned} \tag{3.2}$$

is a closed-loop system and is illustrated in Figure 3.4. We can now formulate the problem of finding a stabilizing controller for (2.2) as: Given the controllability pair (A, B) of (2.2), find a matrix K (if it exists) such that $A - BK$ is stable (and hence, the system (3.2) becomes stable).

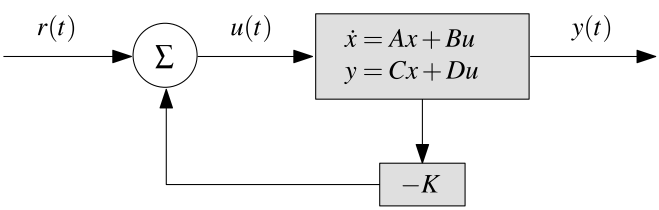


FIGURE 3.4: State-feedback controller.

3.2.1 Stabilizability and detectability

When is it possible to find a matrix K such that $A - BK$ becomes stable? First we take a look at the transformed state-space system (2.16) and the corresponding Figure 2.4. Consider the case when the uncontrollable part is not stable (i.e., one or more eigenvalues of $\widetilde{A}_{\bar{c}}$ are not in the open left-half complex plane), then the output $\widetilde{C}_{\bar{c}}\widetilde{x}_{\bar{c}}(t)$ can grow unbounded to infinity (or, minus infinity). Such a system cannot be stabilized by feedback by using the input $u(t)$, and consequently, precautions have to be taken when designing a controller for an uncontrollable system. This leads to the criteria of stabilizability.

Definition 3.2.1 [35] *The pair (A,B) is said to be* stabilizable *if there exists a matrix K such that $A-BK$ is stable, or, equivalently, the unstable modes of A are controllable.*

From the definition it follows immediately that, if the matrix pair (A,B) is controllable then it is stabilizable. Remark, the opposite is not true; Stabilizability does not imply controllability.

The dual concept of stabilizability is *detectability*.

Definition 3.2.2 [35] *The pair (A,C) is* detectable *if there exists a matrix L such that $A-LC$ is stable, or, equivalently, the unstable modes of A are observable.*

From the duality it follows that (A,C) is detectable if and only if (A^T,C^T) is stabilizable.

EXAMPLE 3

Consider the state-space model

$$\dot{x}(t) = \begin{bmatrix} 1 & 1 & -3 \\ 0 & 3 & 2 \\ 0 & 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} u(t).$$

The system is uncontrollable with $\text{rank}(\mathbf{C}(A,B)) = 2$. From Theorem 2.5.1 the system can be decomposed into its controllable and uncontrollable parts (rounded to four decimals):

$$UAU^{-1} = \begin{bmatrix} \widetilde{A}_c & \widetilde{A}_{12} \\ 0 & \widetilde{A}_{\bar{c}} \end{bmatrix} = \left[\begin{array}{cc|c} 1.0 & 0 & 2.0 \\ -2.8284 & 1.0 & 1.4142 \\ \hline 0 & 0 & 3.0 \end{array}\right],$$

$$UB = \begin{bmatrix} \widetilde{B}_c \\ 0 \end{bmatrix} = \left[\begin{array}{c} -1.41 \\ 0 \\ \hline 0 \end{array}\right].$$

As we can see, the uncontrollable part $A_{\bar{c}} = [3.0]$ is unstable and therefore the system is not stabilizable.

If we instead look at the following system:

$$\dot{x}(t) = \begin{bmatrix} 1 & 1 & -3 \\ 0 & 3 & 2 \\ 0 & 0 & -1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} u(t),$$

where the element A_{33} has been changed from 1 to -1. This system is controllable ($\text{rank}(\mathbf{C}(A,B)) = 3$) but still unstable, the eigenvalues of A are $\{-1,3,1\}$. However, this system is stabilizable since the unstable modes are controllable. For example, choose $K = \begin{bmatrix} 80 & -250 & -235 \end{bmatrix}$ then the eigenvalues of $A-BK$ are $\{-4,-4 \pm i8.4261\}$.

3.3 Pole placement

In practice, *just* stabilizing the system is often not enough. We also need to be able to alter the system's behavior in case of fast/slow response and over-/undershoot. These dynamics of the system are defined by the eigenvalues of $A - BK$, and can therefore be controlled by choosing a matrix K such that the poles (and zeros) of the system are located at predefined locations.

This leads to the *pole placement problem* (or *EVA problem by state feedback*). The aim is to find a matrix K such that the eigenvalues of $A - BK$ are assigned a specific set of scalars $\Lambda = \{\mu_1, \ldots, \mu_n\}$, i.e., $\sigma(A-BK) = \Lambda$ where $\sigma(A-BK)$ is the spectrum of $A - BK$. Furthermore, the set Λ of eigenvalues must be closed under complex conjugation, i.e., if a complex eigenvalue $\alpha \in \Lambda$ then $\overline{\alpha} \in \Lambda$.

Theorem 3.3.1 [35] *The pole placement problem is solvable for all Λ if and only if (A, B) is controllable. The solution is unique if and only if the system is a single-input system (i.e., if B is a vector). In the multi-input case, if the problem is solvable, there are infinitely many solutions.*

For further reading and numerical methods to solve the pole placement problem, see [7, 35, 89, 90] and references therein.

3.4 Linear quadratic regulator problem

Which eigenvalues to choose for the pole placement problem is a non-trivial problem. Therefore, we will turn our attention towards a method based on linear optimal control: the *linear quadratic regulator problem* (LQR problem) [3, 35, 88, 101]. As the name indicates, the LQR problem only considers the regulator problem as defined in Section 3.1. However, corresponding methods for the tracking problem exist, e.g., see [3]. The objective is to find an optimal controller for a linear system that takes an arbitrary nonzero initial state x_0 to the zero state at some time t_f, preferably as fast as possible. If $t_f < \infty$, the regulator problem is said to be of finite-time else, if $t_f = \infty$, of infinite-time. The LQR problem for continuous-time linear systems results in solving a *Riccati differential equation* (RDE) or a *continuous-time algebraic Riccati equation* (CARE). The discrete-time case is out of the scope of this Thesis and therefore will not be considered. In the following, we give a brief introduction to the LQR problem, the RDE, and the CARE. For details on the LQR problem see for example the text book by Anderson and Moore [3]. Theoretical background on the Riccati equation and methods for solving CARE and RDE can be found in [1, 22, 35, 82, 88, 101] and references therein. Large-scale systems are considered in [16, 20, 65]. A historical overview of the Riccati equation is presented in [22].

3.4.1 The quadratic cost functional

When constructing a controller for the regulator problem there are some limitations of the plant that have to be taken into account. In the following, we present an informal explanation of these performance criteria.

Recall from above, that the goal of the LQR problem is to take a nonzero initial state to the zero state as fast as possible. One natural solution would be to take the time t_f as close to t_0 as possible. However, as t_f approaches t_0 the required control energy to affect the states increases and will eventually be out of limit for the plant. Consequently, we want the magnitude of the control action to be bounded or kept to a minimum. One measurement that achieves this is to minimize

$$\int_{t_0}^{t_f} u(t)^T R(t) u(t) dt,$$

where $R(t)$ is a symmetric positive definite matrix for all t. The second criterion that has to be considered is that $\|x(t)\|$ should be kept small in the interval $[t_0, t_f]$. This is achieved by minimizing

$$\int_{t_0}^{t_f} x(t)^T Q(t) x(t) dt,$$

where $Q(t)$ is symmetric positive semidefinite. Moreover, from control point of view it is often not significant that the states actually reaches zero. Instead it is only necessary that some norm of the states is minimized at time t_f, for example, the quadratic form $x(t_f)^T F x(t_f)$, where the constant matrix F is symmetric positive semidefinite. This is the third and last criterion.

In summary, the above suggests that an optimal control action $u^*(t)$ can be achieved by minimizing the *quadratic cost functional* (the *quadratic performance index*)

$$J(x(t_0), u(\cdot), t_0) = \int_{t_0}^{t_f} \left[x(t)^T Q(t) x(t) + u(t)^T R(t) u(t) \right] dt + x(t_f)^T F x(t_f), \quad (3.3)$$

where $Q(t) = Q(t)^T$ and $F = F^T$ are symmetric positive semidefinite matrices, and $R(t) = R(t)^T$ is a symmetric positive definite matrix. Often, satisfactory results can even be achieved by letting $F = 0$.

The matrices $Q(t)$ and $R(t)$ are called *weighting matrices* for the states and the control vector, respectively. How to choose these matrices is not a simple task and no specific guidelines exist, however, good insight of the underlying system helps.

3.4.2 The time-varying regulator problem

Consider the LTV system

$$\dot{x}(t) = A(t)x(t) + B(t)u(t), \qquad x(0) = x_0, \quad (3.4)$$

where the matrices $A(t) \in \mathbb{R}^{n \times n}$ and $B(t) \in \mathbb{R}^{n \times m}$ are assumed to be time-varying matrices with continuous entries.

First, we look at the finite-time regulator problem of the LTV system (3.4) where $t \in [t_0, t_f]$, with the quadratic cost functional (3.3). Assume that $Q(t)$ and $R(t)$ are time-varying matrices with continuous entries. The core of the LQR problem is to find the optimal control $u^*(t)$ that minimizes (3.3) subject to the dynamics of (3.4).

The optimal cost functional $J^*(x(t), t)$ that minimizes (3.3) with respect to $u(t)$, $t_0 \leq t \leq t_f$, has the quadratic form

$$J^*(x(t), t) = x(t)^T X(t) x(t),$$

for some symmetric matrix $X(t)$. It can be shown that the matrix $X(t)$ is the solution of the *Riccati differential equation* (RDE) (e.g., see [3])

$$-\dot{X}(t) = A(t)^T X(t) + X(t)A(t) - X(t)B(t)R(t)^{-1}B(t)^T X(t) + Q(t), \tag{3.5}$$

with the boundary condition $X(t_f) = F$, where F is the same matrix as in (3.3). It follows that the optimal control that minimizes (3.3) and stabilizes (3.4) is the linear state feedback

$$u^*(t) = -K(t)x(t), \quad \text{where} \quad K(t) = R(t)^{-1}B(t)^T X(t).$$

Now we consider the infinite-time regulator problem, i.e., where $t_f = \infty$. In this case, the matrix F in (3.3) is usually assumed to be zero. To ensure that the optimal cost functional is finite, the assumption that the pair $(A(t), B(t))$ is stabilizable has to be added. To guarantee stability of the closed-loop system $\dot{x}(t) = (A(t) - B(t)K(t))x(t)$, the pair $(A(t), Q(t)^{1/2})$ also need to be detectable, where $(Q(t)^{1/2})^T Q(t)^{1/2} = Q(t)$.

Let $X(t,t_f)$ be the solution of the RDE (3.5) with the boundary condition $X(t_f,t_f) = 0$. Then $\overline{X}(t) = \lim_{t_f\to\infty} X(t,t_f)$ exists and is a solution of (3.5). It follows that the optimal cost functional $J^*(x(t),t)$ is $x(t)^T \overline{X}(t)x(t)$, and the optimal control at time t is uniquely given by

$$u^*(t) = -R(t)^{-1}B(t)^T \overline{X}(t)x(t).$$

In Chapter 5, we will revisit the problem of solving the RDE (3.5) and particularly the *periodic RDE* (PRDE), where the time-varying matrices $A(t)$, $B(t)$, $Q(t)$, and $R(t)$ are periodic matrices. In Papers III and IV, numerical methods for solving the PRDE are evaluated.

3.4.3 The time-invariant regulator problem

Consider the LTI system

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) = x_0, \tag{3.6}$$

where the matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are constant matrices.

The finite-time regulator problem has a quadratic cost functional (3.3) with constant weighting matrices:

$$J(x(t_0), u(\cdot), t_0) = \int_{t_0}^{t_f} \left[x(t)^T Qx(t) + u(t)^T Ru(t)\right] dt + x(t_f)^T F x(t_f), \tag{3.7}$$

where Q is positive semidefinite and R is positive definite. The optimal control that minimizes (3.7) and stabilizes (3.6) is $u^*(t) = -R^{-1}B^T X(t)x(t)$, where $X(t)$ is the solution of the RDE (3.5)

$$-\dot{X}(t) = A^T X(t) + X(t)A - X(t)BR^{-1}B^T X(t) + Q, \tag{3.8}$$

with the boundary condition $X(t_f) = F$.

Now consider the infinite-time regulator problem with $t_f = \infty$. Assume (A, B) is stabilizable and the quadratic cost functional is of the form (3.7) with $F = 0$. Also, let $X(t, t_f)$ be the solution of (3.8). Then $X = \lim_{t_f \to \infty} X(t, t_f) = \lim_{t \to \infty} X(t, t_f)$ is constant and satisfies the *continuous-time algebraic Riccati equation* (CARE)

$$A^T X + XA - XBR^{-1}B^T X + Q = 0. \tag{3.9}$$

The optimal control is $u^*(t) = -R^{-1}B^T Xx(t)$. In addition, suppose that $(A, Q^{1/2})$ is detectable, then X is the unique positive semidefinite solution of (3.9) and the optimal cost functional $J^*(x(t), t)$ is $x(0)^T Xx(0)$.

CHAPTER 4

Canonical Structure Information and Stratification

Determining the system characteristics (like poles, zeros, controllability, and observability) of an LTI system (2.2) involves computing the canonical structure information of the associated system pencil (2.10). This information is revealed by the canonical form of the system pencil.

This chapter introduces some of the most common canonical forms for matrices, matrix pencils, and system pencils. We also give a brief introduction and motivation to stratification of orbits and bundles. For a more detailed explanation we refer to Papers I and II, where in Paper I a comprehensive introduction to canonical forms and stratification of orbits and bundles of matrices, matrix pencils and system pencils is presented. In both papers, the mathematical background theory is presented as well as the relation to linear systems and control theory. Paper II is devoted to the stratification of controllability and observability pairs, and also illustrates the stratification theory on two examples from system and control applications.

4.1 Canonical forms

Any matrix (or matrix pencil) can be transformed into a *canonical form* in terms of similarity (or equivalence) transformations. A canonical form is the simplest or most symmetrical form a matrix or matrix pencil can be reduced to. The canonical forms reveal the *canonical structure information* from which the system characteristics are deduced.

For any matrix $A \in \mathbb{C}^{n\times n}$ there exists a *similarity transformation* with a nonsingular matrix P such that $\widetilde{A} = PAP^{-1}$, where $\widetilde{A}$ is in the *Jordan canonical form* (JCF) [54], also called *Jordan normal form* (JNF). Notice, throughout the Thesis the transformation matrix applied to the right side of a matrix in similarity and equivalence transformations is expressed as a matrix inverse. The JCF reveals the simplest standard shape a matrix can be reduced to. The matrix $\widetilde{A}$ in JCF is a block diagonal matrix of so called *Jordan blocks* associated with an (possibly multiple) eigenvalue $\mu_i \in \mathbb{C}$, where each Jordan block is defined as

$$J(\mu_i) = \begin{bmatrix} \mu_i & 1 & & \\ & \mu_i & \ddots & \\ & & \ddots & 1 \\ & & & \mu_i \end{bmatrix}. \tag{4.1}$$

The dimensions and number of the Jordan blocks associated with each distinct eigenvalue is unique, but not the ordering of the blocks along the diagonal. Two similar matrices $B = PAP^{-1}$ have always the same set of eigenvalues and the same JCF.

EXAMPLE 4

Consider the following three different matrices in JCF

$$\widetilde{A}_1 = \left[\begin{array}{c|c|c} 2 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 2 \end{array}\right], \quad \widetilde{A}_2 = \left[\begin{array}{cc|c} 2 & 1 & 0 \\ 0 & 2 & 0 \\ \hline 0 & 0 & 2 \end{array}\right], \text{ and } \quad \widetilde{A}_3 = \left[\begin{array}{ccc} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{array}\right],$$

each matrix associated with a 3×3 matrix A_j $(j = 1,2,3)$ with three multiple (identical) eigenvalues ($\mu = 2$), where $\widetilde{A}_j = PA_jP^{-1}$. The first matrix $\widetilde{A}_1$ has three 1×1 Jordan blocks associated with the multiple eigenvalue 2 on the diagonal. In this case, the matrix has three linearly independent eigenvectors and is diagonalizable.

The other two matrices correspond to 3×3 matrices which do not have a full set of eigenvectors, known as *defective matrices*. A defective matrix is not diagonalizable and it has not enough eigenvectors to construct the matrix P. To get a complete base, so called *principal vectors* are used. The JCF of these matrices has at least one Jordan block of size 2×2 or larger. In general, a Jordan block $J_k(\mu)$ of size $k \times k$ has one eigenvector and $k-1$ principal vectors.

The matrix $\widetilde{A}_2$ has one Jordan block of size 1×1 and one Jordan block of size 2×2 associated with one eigenvector and one principal vector. The last matrix $\widetilde{A}_3$ has one eigenvector and two principal vectors, and consequently only one Jordan block of size 3×3. In addition, $\widetilde{A}_1$ and $\widetilde{A}_2$ (but not $\widetilde{A}_3$) are also *derogative matrices*, since they have more than one eigenvector associated with the multiple eigenvalue $\mu = 2$.

The generalization of the JCF to general $m_\text{p} \times n_\text{p}$ matrix pencils $G - \lambda H$ is the *Kronecker canonical form* (KCF) [54]. Two matrix pencils $G - \lambda H$ and $\widetilde{G} - \lambda\widetilde{H}$ are *strictly equivalent* if there exist two nonsingular matrices U and V such that $\widetilde{G} - \lambda\widetilde{H} = U(G - \lambda H)V^{-1}$. Any matrix pencil can be transformed into KCF in terms of an equivalence transformation such that $\widetilde{G} = \text{diag}(G_1, \dots, G_b)$ and $\widetilde{H} = \text{diag}(H_1, \dots, H_b)$ are block diagonal. Each block $G_i - \lambda H_i$ must be of one of the following forms: L_k, L_k^T, $J_k(\mu)$, or N_k. If $m_{\rm p} = n_{\rm p}$ and $\det(G - \lambda H) \equiv 0$ only if $\lambda \in \mathbb{C}$ is an eigenvalue, then $G - \lambda H$ is called a *regular pencil* and only consists of Jordan blocks of finite and (if B is singular) infinite eigenvalues. These two types of blocks, $J_k(\mu_i)$ of size $k \times k$ associated with the finite eigenvalue μ_i and N_k of size $k \times k$ associated with the infinite eigenvalue, are defined as

$$J_k(\mu_i) = \begin{bmatrix} \mu_i - \lambda & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \mu_i - \lambda \end{bmatrix}, \quad \text{and} \quad N_k = \begin{bmatrix} 1 & -\lambda & & \\ & \ddots & \ddots & \\ & & \ddots & -\lambda \\ & & & 1 \end{bmatrix}.$$

If $m_{\rm p} \neq n_{\rm p}$ or $\det(G - \lambda H) \equiv 0$ for all $\lambda \in \mathbb{C}$, then the matrix pencil is a *singular pencil* and also includes a singular part. The singular part consists of $k \times (k+1)$ *right singular blocks* L_k and $(k+1) \times k$ *left singular blocks* L_k^T, which are defined as

$$L_k = \begin{bmatrix} -\lambda & 1 & & \\ & \ddots & \ddots & \\ & & -\lambda & 1 \end{bmatrix}, \quad \text{and} \quad L_k^T = \begin{bmatrix} -\lambda & & \\ 1 & \ddots & \\ & \ddots & -\lambda \\ & & 1 \end{bmatrix}.$$

EXAMPLE 5

The KCF of a regular 5×5 matrix pencil $G - \lambda H$ with two $J_2(\alpha)$ blocks and one $J_1(\beta)$ block has the following block diagonal form:

$$G^{(1)} - \lambda H^{(1)} = \operatorname{diag}(J_2(\alpha), J_2(\alpha), J_1(\beta)) =$$

$$= \begin{bmatrix} \alpha - \lambda & 1 & & & \\ 0 & \alpha - \lambda & & & \\ & & \alpha - \lambda & 1 & \\ & & 0 & \alpha - \lambda & \\ & & & & \beta - \lambda \end{bmatrix}.$$

The KCF of a singular 5×5 matrix pencil $G - \lambda H$ with one L_0 block, one

L_2^T block, and one $J_2(\alpha)$ block has the following block diagonal form:

$$G^{(2)} - \lambda H^{(2)} = \operatorname{diag}(L_0, L_2^T, J_2(\alpha)) =$$

$$= \begin{bmatrix} \, & & & \\ & -\lambda & 0 & & \\ & 1 & -\lambda & & \\ & 0 & 1 & & \\ & & & \alpha-\lambda & 1 \\ & & & 0 & \alpha-\lambda \end{bmatrix}.$$

The ordering of the regular blocks in $G^{(1)} - \lambda H^{(1)}$ is arbitrary. In $G^{(2)} - \lambda H^{(2)}$, L_0 must appear before L_2^T but $J_2(\alpha)$ can be anywhere in the block diagonal KCF.

When considering system pencils $\mathbf{S}(\lambda)$ associated with the state-space system (2.2), it is more appropriate to use a canonical form that preserves the special block structure of $\mathbf{S}(\lambda)$. One such canonical form is the *generalized Brunovsky canonical form* (GBCF) [93, 103], which is a generalization of the *Brunovsky canonical form* (BCF) [28]. Let $P \in \mathbb{C}^{n\times n}$, $T \in \mathbb{C}^{p\times p}$, and $Q \in \mathbb{C}^{m\times m}$ be nonsingular matrices, and $S \in \mathbb{C}^{n\times p}$ and $R \in \mathbb{C}^{m\times n}$. Then, for any system pencil $\mathbf{S}(\lambda)$ there exists a *feedback equivalence transformation*

$$U\mathbf{S}(\lambda)V^{-1} = \begin{bmatrix} P & S \\ 0 & T \end{bmatrix} \begin{bmatrix} A-\lambda I_n & B \\ C & D \end{bmatrix} \begin{bmatrix} P^{-1} & 0 \\ R & Q^{-1} \end{bmatrix} = \begin{bmatrix} A_B-\lambda I_n & B_B \\ C_B & D_B \end{bmatrix},$$

such that the matrix quadruple (A_B, B_B, C_B, D_B) is in GBCF, defined by

$$\left[\begin{array}{c|c} A_B-\lambda I_n & B_B \\ \hline C_B & D_B \end{array}\right] = \left[\begin{array}{cccc|cccc} A_\varepsilon & 0 & 0 & 0 & B_\varepsilon & 0 & 0 & 0 \\ 0 & A_\eta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & A_\infty & 0 & 0 & B_\infty & 0 & 0 \\ 0 & 0 & 0 & A_\mu & 0 & 0 & 0 & 0 \\ \hline 0 & C_\eta & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_\infty & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & D_\infty & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & D_0 \end{array}\right].$$

The subsystem $(A_\varepsilon, B_\varepsilon)$ corresponds the L_k blocks, $k \geq 1$, in the KCF of $\mathbf{S}(\lambda)$. It is controllable but not observable and consists of decoupled chains of integrators with inputs, but no outputs. The subsystem (A_η, C_η) corresponds to the L_k^T blocks, $k \geq 1$. It is observable but not controllable and consists of decoupled chains of integrators with outputs, but no inputs. The subsystem $(A_\infty, B_\infty, C_\infty)$ corresponds to the N_k blocks,

$k \geq 2$, which form the infinite zero structure. It is controllable and observable and consists of decoupled chains of integrators with inputs and outputs. The subsystem A_μ corresponds to all $J_k(\mu_i)$ blocks of the finite eigenvalues. It is uncontrollable and unobservable and consists of the finite zeros of the original system. The subsystem D_∞ corresponds to the N_1 blocks, and passes inputs unchanged to outputs. The subsystem D_0 corresponds to the L_0 and L_0^T blocks, and annihilates inputs and generates identically zero outputs.

4.2 Motivation for stratification of orbits and bundles

It is a well known fact that computing canonical forms (JCF, KCF, GBCF, etc.) in finite precision arithmetic are ill-posed problems (e.g., see [55, 77]). The transformation matrices that reduce, for example, a matrix to JCF can be arbitrary ill-conditioned and therefore the computed canonical structure can be completely changed by a small perturbation of the matrix. Instead so called *staircase-type forms* are used [10, 29, 76, 80, 107, 111], from which we can retrieve the same canonical structure information as from the simplest canonical forms. See also Paper I where a summary of a few staircase-type forms is presented. These staircase-type forms are computed by only using orthogonal (or unitary) transformation matrices and backward stable algorithms, called *staircase algorithms*. However, the staircase algorithms perform rank and nullity decisions based on certain tolerance parameters. As changing these parameters may change the computed staircase form, this still leads to uncertainty in the computed canonical structure information. Together with the uncertainty in the state-space system itself, e.g., due to round-off and measurement errors, it motivates the research and development of algorithms for studying qualitative as well and quantitative information of nearby canonical structures.

The qualitative information about nearby systems is revealed by the theory of *stratification* [43, 44]. A stratification shows which canonical structures are near to each other in the sense of small perturbations and their relations to other canonical structures, i.e., it reveals the closure hierarchy of orbits and bundles of canonical structures. An *orbit* is for matrices the manifold of similar matrices and for matrix pencils the manifold of equivalent matrix pencils, i.e, all matrices or matrix pencils in an orbit have the same canonical form. A *bundle* is the union of all orbits with the same canonical form but with unspecified eigenvalues [8].

An $n \times n$ matrix A can be seen as a point in an n^2-dimensional (matrix) space, one dimension for each parameter of A. Consequently, the union of all $n \times n$ matrices constitutes the entire matrix space, and an orbit of a matrix is a manifold in the space. Similarly, an $m_\text{p} \times n_\text{p}$ matrix pencil belongs to a $2m_\text{p}n_\text{p}$-dimensional space, and an $(n+m) \times (n+p)$ system pencil to an $(n+m)(n+p)$-dimensional space.

Following [44, 47], the stratification can be presented as a graph where each node represents an orbit or bundle of a canonical structure. The graph is organized with the most generic structure(s) at the top and other structures further down, ordered by increasing degeneracy (codimension). The hierarchy is revealed by the closure and cover relations among these orbits or bundles, where a cover relation guarantees that two orbits or bundles are nearest neighbours in the closure hierarchy. In the stratification graph, a cover relation is represented by an edge between two nodes. We end this chapter by outlining the concept of stratification with an illustrative example.

Consider a three-dimensional space which includes a surface, two curves, and one intersection point. This example is only of dimension three and can therefore easily be visualized, see left illustration in Figure 4.1. For general matrices, matrix pencils, and system pencils it is hard (or even impossible) to do a similar graphical illustration. The three dimensional space in Figure 4.1 can hierarchically be organized as the graph on the right side. The most generic structure is the complete space (denoted S), in the space we have a surface (denoted s) on which two curves (denoted c_1 and c_2) intersect each other in one point (denoted p). Let each node of this graph represent an orbit, which in turn, loosely speaking, is illustrated by one of the five manifolds in the space. When we now talk about the space, we mean the whole space except the surface, the two curves and the point. Instead we say that the surface, the curves, and the point, lie in the closure of the space. Likewise, the surface is the complete surface except the curves and the point, and the curves and the point lie in the closure of the surface. Remember that all matrices or matrix pencils in the same orbit have the same canonical form, and therefore we can represent each orbit by its canonical form.

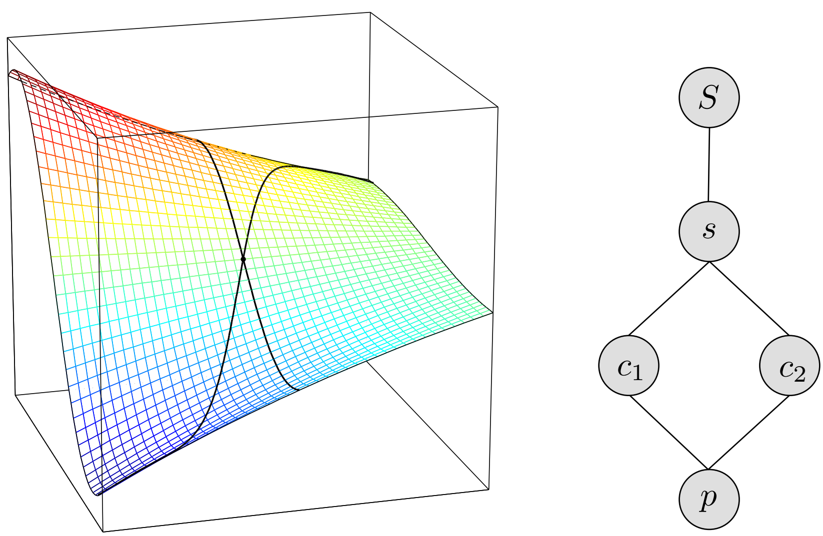


FIGURE 4.1: On the left, a three dimensional space with an intersection of two curves on a surface. On the right, the graph representing the hierarchy of the space, where S is the complete space, s the surface, c_1 and c_2 the curves, and p the intersection point.

Let us first consider the point p, which is the least generic (most degenerate) structure. Any small displacement from the point will lead to that we end up on one of the curves, the surface, or in the surrounding space. This corresponds to that by any small perturbation of the canonical structure p we will get one of the canonical structures

c_1, c_2, s, or S. Generally, it is always possible to go from any canonical structure to another higher up in the graph by a small perturbation if and only if they are connected by a path directed strictly upwards. The other way around is normally not possible, i.e., a canonical structure does not have to be near a structure below in the graph. Furthermore, two canonical structures on the same level in the graph are never near each other. In the example, this can be illustrated by considering one of the curves, e.g., c_1, for which it exists a small displacement that brings us to the surface s or the space S. However, we do not have to be close to the point p which is below in the closure hierarchy. Note, the line c_2 is never the closest structure to c_1, it is always possible to find a smaller displacement to s or S than to c_2 (except when we are in the intersection point p).

CHAPTER 5

Periodic Riccati Differential Equations

The main topic of this chapter is numerical methods for solving the *periodic Riccati differential equation* (PRDE). The PRDE arises, e.g., when solving the LQR problem for periodic LTV systems. In the following, we introduce numerical methods explicitly designed for solving the PRDE. Paper III evaluates two of these methods: the one-shot method and the multi-shot method. In addition, Paper IV presents known theoretical results for the PRDE and detailed descriptions of some recently proposed methods for solving the PRDE. These methods are the multi-shot method (two variants) and the SDP method. First, they are evaluated on a set of artificial systems. Then they are used for synthesis of a controller to achieve stable oscillations of two mechanical systems: the Furuta pendulum and a family of pendulum on cart systems.

5.1 Motivation and sample applications

Periodic behaviors are common in all types of biological and mechanical systems. By a periodic behavior we mean a behavior that repeats itself continuously (if not interrupted), i.e., the behavior repeatedly returns to its initial state after a finite time again and again. Examples of systems with a periodic behavior are:

- waves in ocean, circular motions of planets, prey-predator oscillations,
- heart-beats, flows in the cerebrospinal fluid system,
- human motions (and motions of a humanoid robot),
- helicopters, revolving satellites, electrical systems.

In the following, we will focus on controlled *mechanical systems*. These systems are of particular interest since they can often be accurately modeled, and problems of periodic motion planning and orbital stabilization are usually well defined. Other types of systems may not have all these criteria, for example, the motions of the planets cannot be controlled and the heart-beats cannot be described by an accurate model.

One of the above listed examples is human motions. Even if a human motion is usually performed in finite time, they can be seen (characterized) as a part of a repetitive sequence. To understand and imitate these motions are of especial interest for robotic researchers designing humanoid robots [81, 92] and for researches involved in sports medicine and rehabilitation. Typical motions of interest are sitting down or rising up from a chair, walking, running, etc. The control design of such a motion leads to an underactuated control system. By an underactuated system we mean a system with more degrees of freedom than number of actuators.

The main control problem we are interested in is to find a method for motion planning of underactuated mechanical systems, which can be (orbital) stabilized. One approach that has been proven successful is the idea of imposing *virtual holonomic constraints* on the system dynamics by feedback control. The theory of virtual holonomic constraints will not be discussed further in this Thesis, instead we refer to [32, 92].

Both for education and for benchmarking and testing advanced control designs, inverted pendulums are one of the most common used mechanical test systems. Examples of setups with inverted pendulums are the Furuta pendulum [53], the Pendubot [52], pendulums on carts [50], and the Acrobot [87]. The popularity of these mechanical systems are due to the fact that they have few (2–3) degrees of freedom and can be easily modeled, but are difficult to control and plan motions for.

Two models of mechanical systems used in Paper IV are a model of the Furuta pendulum and a model of pendulums on carts. The Furuta pendulum is a mechanical system with two degrees of freedom, see Figure 5.1. It consists of a horizontal arm that can be rotated around a vertical axis, and a pendulum attached to the end of the arm. The arm is actuated by a DC-motor, while the pendulum can freely rotate in the vertical plane perpendicular to the arm.

The second example is a family of pendulum-cart systems (say C copies), each system consisting of a cart with an inverted pendulum attached on top, see Figure 5.2. Each cart is controlled by a force applied along the horizontal axis, where the inverted pendulum can freely move in the vertical plane along the cart.

In Paper III, a model of another type of mechanical system is used. It is a juggling device, called the devil stick, which consists of a center stick and a hand stick. The center stick has a periodic propeller-like motion which is induced by a contact force from the hand stick, see Figure 5.3. For details, see Paper III and [51, 94].

Our main goal for each system is to find an interesting periodic solution and to consider tasks for its orbital stabilization. For the Furuta pendulum we are interested in motions corresponding to one particular swing-up strategy, for which we can vary and study the possibility to achieve orbital stability. By doing simulations with the model of the Furuta pendulum we can find a periodic motion trajectory of the pendulum with an arbitrary large period. Such a trajectory can be seen in Figure 5.4, where θ is the angle of the pendulum. For the pendulum-cart system we are interested to achieve a stable synchronization of oscillations of the pendulums and carts, where the initial states of the pendulums and carts are chosen randomly (in vicinity of the tangent orbit). For this system, we can choose any number of carts and therefore get a control problem with an arbitrary large number of degrees of freedom. Finally, the goal for the devil stick is an orbital stabilization of perpetual rotations of the center stick around the hand stick.

FIGURE 5.1: The Furuta pendulum built at Department of Applied Physics and Electronics, Umeå University.

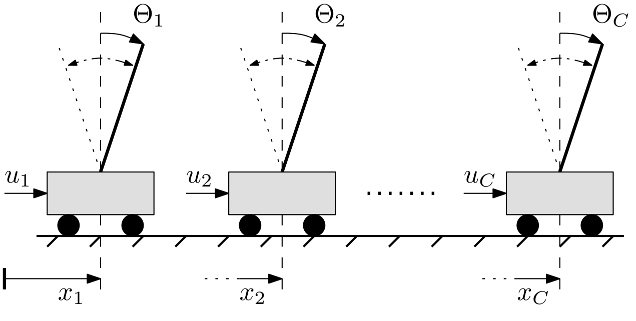


FIGURE 5.2: C identical Pendulum-cart systems.

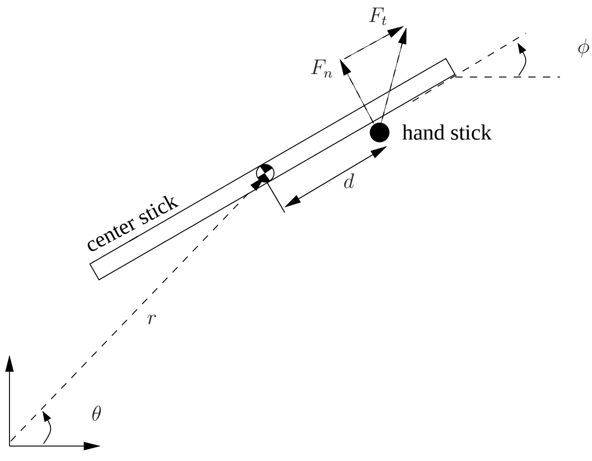


FIGURE 5.3: Model of the devil stick [51].

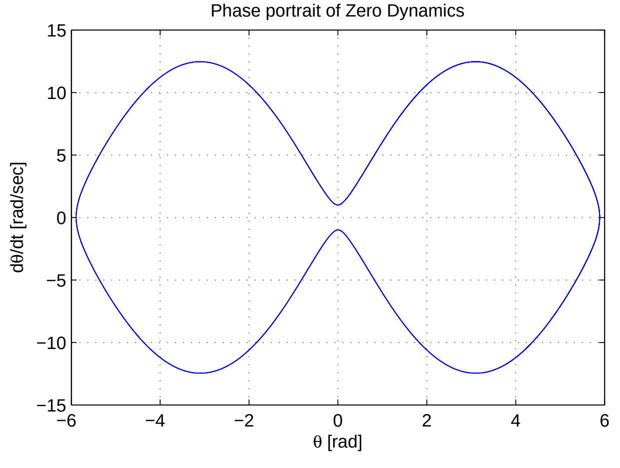


FIGURE 5.4: Phase portrait of a periodic trajectory of the Furuta pendulum, where θ is the angle of the pendulum.

For stabilization of the three mechanical systems a state-feedback design is used. The feedback matrix for a periodic system is computed by solving the LQR problem for a periodic linear system (see Section 5.2), where the main step is to find the solution to a PRDE. Simulink[1] models of the mechanical systems are used for simulating and evaluating the results in Papers III and IV. Orbital stabilization of the Furuta

[1] Simulink is a modeling and simulation software. Simulink is a registered trademark of The MathWork, Inc.

pendulum has also previously been successfully tested experimentally on a physical set-up [53].

5.2 The periodic LQR problem

As we described in Section 3.4.2, the main step in solving the infinite-time LQR problem for continuous-time LTV systems (3.4)

$$\dot{x}(t) = A(t)x(t) + B(t)u(t), \qquad x(0) = x_0,$$

is to find the solution of the RDE (3.5)

$$-\dot{X}(t) = A(t)^T X(t) + X(t)A(t) - X(t)B(t)R(t)^{-1}B(t)^T X(t) + Q(t).$$

We will now consider the *periodic LQR* problem for periodic LTV systems (3.4), where the matrices $A(t) \in \mathbb{R}^{n\times n}$ and $B(t) \in \mathbb{R}^{n\times m}$ are continuous T-periodic matrices, i.e., $A(t) = A(t+T)$ and $B(t) = B(t+T)$ for all $t \geq 0$. The optimal periodic controller is obtained by minimizing the quadratic cost function (3.3) (with $t_f = \infty$ and $F = 0$)

$$J(x(t_0), u(\cdot), t_0) = \int_{t_0}^{\infty} \left[x(t)^T Q(t)x(t) + u(t)^T R(t)u(t)\right] dt,$$

where $Q(t) = Q(t)^T$ is a continuous T-periodic symmetric positive semidefinite matrix and $R(t) = R(t)^T$ is a continuous T-periodic symmetric positive definite matrix.

Provided the pair $(A(t), B(t))$ is stabilizing and the pair $(A(t), Q(t)^{1/2})$ is detectable. The optimal control input $u^*(t)$ that stabilizes (3.4) and minimizes (3.3) is $u^*(t) = -R(t)^{-1}B(t)^T X(t)x(t)$, where $X(t)$ is the unique symmetric positive semidefinite T-periodic stabilizing solution of the PRDE of the form (3.5).

5.3 Numerical methods

Various numerical methods for solving the PRDE (3.5) exist, see Papers III and IV and references therein. However, several of them are unreliable or even unsuitable for PRDEs originating from systems of high order or with large period. We here outline two alternative approaches: the multi-shot method (two variants) and the SDP method. A third method has recently been presented in [2]. It is an iterative method that approximates the solution of the PRDE with a sequence of PRDEs with a negative semidefinite quadratic term. The method utilizes the fast method [115, 117] (see below). We have not done any further evaluation of this method and it is therefore not included in the comparison below. We remark that none of these solvers are suitable for large-scale systems (where $n \gtrsim 1000$). Moreover, we only consider the PRDE of the form (3.5). The methods can, however, be used to solve other types of PRDEs.

One traditional method to solve the PRDE is the *one-shot generator method* [69, 119], which computes an initial condition matrix X_0, called a periodic generator, to the PRDE such that the solution is periodic with boundary conditions $X(t_0) = X(T) =$

X_0. The one-shot method is in general not numerically reliable since two ODEs with possibly unstable dynamics have to be solved. Therefore, an alternative approach has been proposed in [115, 117], called the *multi-shot method* in the sequel. The method is of the type *multiple shooting* where the continuous-time problem is turned into an equivalent discrete-time problem.

Both the one-shot and the multi-shot methods belong to a family of methods which relies on that the solution of the PRDE is obtained by solving the *linear Hamiltonian system*

$$\frac{\partial}{\partial t}\Phi_H(t,t_0) = H(t)\Phi_H(t,t_0), \quad \Phi_H(t_0,t_0) = I_{2n}, \tag{5.1}$$

where $\Phi_H(t,t_0)$ is the symplectic *transition matrix* associated with the $2n \times 2n$ periodic time-varying *Hamiltonian matrix*

$$H(t) = \begin{bmatrix} A(t) & -B(t)R(t)^{-1}B(t)^T \\ -Q(t) & -A(t)^T \end{bmatrix}.$$

Since $H(t)$ is Hamiltonian it satisfies $H(t)^T J + JH(t) = 0$ for all t, where J is defined by

$$J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}.$$

The $2n$ eigenvalues of the transition matrix $\Phi_H(t,t_0)$ always appear in pairs of the form $(\mu, 1/\mu)$, i.e., they are symmetric with respect to the unit circle. For a T-periodic system, the transition matrix evaluated over one period is known as the *monodromy matrix* $\Psi_H(t_0) = \Phi_H(t_0+T,t_0)$. In the following, we assume $t_0 = 0$.

For both the one-shot and multi-shot methods, the linear Hamiltonian system (5.1) is preferably solved with a symplectic ODE solver, which preserves the symplectic flow of the problem. One example of such a solver is the symplectic and symmetric Gauss Runge-Kutta method [68].

The linear Hamiltonian system (5.1) is known to be ill-conditioned due to unstable dynamics, especially for systems with large period. This is the main motivation for the multi-shot method. To reduce the impact of the numerical inaccuracy from the unstable dynamics, the monodromy matrix is expressed in the product form

$$\Psi_H(0) \equiv \Phi_H(T,0) = \Phi_H(T,T-\Delta)\cdots\Phi_H(2\Delta,\Delta)\Phi_H(\Delta,0), \tag{5.2}$$

where $\Delta = T/N$ for a suitable integer N. The integration of the linear Hamiltonian system (5.1) can now be done over subintervals of the whole period. The first step in the multi-shot algorithm is to compute $\Phi_H(k\Delta,(k-1)\Delta)$, $k = 1,\ldots,N$, by solving (5.1) for each interval $[(k-1)\Delta, k\Delta]$. The monodromy matrix is now expressed as an N-cyclic matrix sequence. Generally, a matrix sequence $X_1, X_2, \ldots, X_N$ is called N-cyclic if $X_{k+N} = X_k$ for any positive integer k. In the next step, the *stable periodic invariant subspaces* $Q^{(1)}, Q^{(2)}, \ldots, Q^{(N)}$ associated with the n eigenvalues inside the unit circle are determined by computing the *ordered periodic real Schur form* [23, 70, 61] of the cyclic matrix sequence (5.2). Let

$$Q^{(k)} = \begin{bmatrix} Q_1^{(k)} \\ Q_2^{(k)} \end{bmatrix}, \quad k = 1,\ldots,N,$$

then the solution $X_k = X((k-1)\Delta)$ of the PRDE is $X_k = Q_2^{(k)}(Q_1^{(k)})^{-1}$.

The second variant of the multi-shot method is based on the same algorithm but uses the *fast method* [116, 117] to compute the stable invariant subspace. This method belongs to the family of "fast" methods [12, 33] for *discrete-time algebraic Riccati equations* (DARE), and is an extension of the *swapping and collapsing* approach [11, 12]. As shown in [117], the fast method for DARE can be adapted to solve the PRDE. The method implicitly constructs a stable invariant subspace from an associated lifted pencil (explained in Paper IV), from which the initial value X_1 is determined. The remaining solutions X_k, $k \geq 2$, of the PRDE are then computed iteratively using the well-known relation (e.g., see [21])

$$X_k = \left(X_{k+1}\Phi_{12}^{(k)} - \Phi_{22}^{(k)}\right)^{-1}\left(\Phi_{21}^{(k)} - X_{k+1}\Phi_{11}^{(k)}\right),$$

where

$$\Phi_H(k\Delta,(k-1)\Delta) = \begin{bmatrix} \Phi_{11}^{(k)} & \Phi_{12}^{(k)} \\ \Phi_{21}^{(k)} & \Phi_{22}^{(k)} \end{bmatrix}.$$

The second method we consider does not solve the PRDE using the linear Hamiltonian system (5.1). Instead the problem is reformulated as a *semidefinite programming* (SDP) problem with *linear matrix inequality* (LMI) constraints [9, 26, 27, 79]. This method is proposed in [49].

Define the inequality

$$\dot{X}(t) + A(t)^T X(t) + X(t)A(t) - X(t)B(t)R(t)^{-1}B(t)^T X(t) + Q(t) \geq 0, \tag{5.3}$$

and let the stabilizing solution of the PRDE be the maximum of a cost function $J(X(t))$ (we assume that this cost function exists, see Paper IV and [49] for details). The maximization problem is an infinite dimensional SDP problem, for which the linear functional $J(X(t))$ is maximized over a convex set $X(t)$ satisfying the inequality (5.3). By approximating the stabilizing solution $X(t)$ and its derivative with a finite dimensional trigonometric base function, here the finite dimensional Fourier expansion, we have

$$\widetilde{X}(t) = \sum_{k=-q}^{q} e^{\mathrm{i}k\omega t} X_k, \text{ and}$$

$$\frac{d\widetilde{X}(t)}{dt} = \sum_{k=-q}^{q} \mathrm{i}k\omega\, e^{\mathrm{i}k\omega t} X_k,$$

where X_{-k} is the complex conjugate of X_k. The problem can now be formulated as the finite dimensional SDP problem [49]

$$\begin{aligned} &\text{minimize } -J(\widetilde{X}(t)), \\ &\text{subject to } \mathcal{S}_j \geq 0, \quad j = 1,\ldots,N. \end{aligned}$$

The LMI constraints $\mathcal{S}_1,\ldots,\mathcal{S}_N$ determined from (5.3)[2] are

$$\mathcal{S}_j = \mathcal{S}(\widetilde{X}(t),t_j) = \begin{bmatrix} \frac{d\widetilde{X}(t)}{dt} + A(t)^T\widetilde{X}(t) + \widetilde{X}(t)A(t) + Q(t) & \widetilde{X}(t)B(t) \\ B(t)^T\widetilde{X}(t) & R(t) \end{bmatrix}, \ \forall t \geq 0,$$

[2] The reformulation is done using the Schur complement [120].

where $d\widetilde{X}(t)/dt + A(t)^T\widetilde{X}(t) + \widetilde{X}(t)A(t) + Q(t)$ is symmetric.

5.4 The MATLAB implementations

In line of the theory in [115, 117], MATLAB routines of the one-shot and multi-shot methods have been implemented by the author. The multi-shot method utilizes Fortran routines for computing the periodic real Schur from [86] and periodic eigenvalue reordering [61] (to be available in the upcoming *PEP toolbox* [64]). The SDP method [49] is implemented by S. Gusev[3] and uses *SeDuMi* [100, 102] (a MATLAB toolbox for optimization over symmetric cones) to solve the LMI problem and *YALMIP* [84, 85] for modeling the optimization problem. Presently, neither the multi-shot nor the SDP solver are publicly available. The fast multi-shot method [117] is available in the *Periodic system toolbox* for MATLAB [116] by A. Varga[4].

[3] Department of Mathematics and Mechanics, St Petersburg State University, St Petersburg, Russia.
[4] Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Oberpfaffenhofen, Germany.

CHAPTER 6

Software for Systems and Control

In the February 2004 issue of *IEEE Control Systems Magazine*, the topic of numerical awareness in the systems and control community is addressed. As the systems and control problems become larger and more complex, the need for efficient and numerically stable software is growing. As a result, a great effort has in the last 25 years been spent on developing new routines and algorithms. However, as pointed out in [113], papers dealing with problems for systems and control seldom make proper use of terms like numerical stability, problem conditions, accuracy and computational efficiency. Moreover, software and algorithms for *computer-aided control system design* (CACSD) are often developed by scientists and engineers in systems and control with limited numerical knowledge. To attack these problems, a joint work between control specialists, computing scientists and mathematicians is required.

One control group that actively address these problems is the European network called NICONET (*Numerics in Control Network*) Association. NICONET started in 1998 as an European project by the initiative of WGS (*Working Group of Software*), NAG (*Numerical Algorithms Group*, Oxford, U.K.), and DLR (*Deutsches Zentrum für Luft- und Raumfahrt e.V.*, Oberpfaffenhofen, Germany) and was a cooperation between 11 universities/research institutes and six companies. The NICONET project ended in 2002 and as a continuation the club NICONET e.V., based in Braunschweig, Germany, was founded in February 2007. The main goal in NICONET was to develop high-quality CACSD software. This work has resulted in the numerical library SLICOT (*Subroutine Library in Control Theory*) [17, 108], which is freely available for non-commercial usage. The SLICOT library is implemented in Fortran 77 and based on linear algebra routines in BLAS (*Basic Linear Algebra Subprograms*) [40, 41, 83] and LAPACK (*Linear Algebra Package*) [4], and consists of a broad range of routines (about 400) for design and analysis of control systems. MATLAB[1] and Scilab[2] [57] gateway routines to call the SLICOT Fortran library have also been developed and several MATLAB Toolboxes based on SLICOT exist. The SLICOT library and additional software can be found on the dedicated homepage http://www.slicot.org.

[1] MATLAB is a registered trademark of The MathWorks, Inc.
[2] Scilab is maintained and developed by the Scilab Consortium, and is distributed freely as open source. Binaries and source code can be downloaded at Scilab's web site http://www.scilab.org/.

Stand-alone CACSD platforms are for example provided by National Instruments Corporation (e.g., MATRIXx and LabVIEW), DLR (e.g., MOPS), MSC Software Corporation (SimEnterprise solutions), Modelica Association (Modelica), and Dynasim (Dymola). There also exist several software packages (toolboxes) for commercial products not explicitly designed for CACSD, like Maple[3], Mathematica[4], MATLAB, Scilab (non-commercial), and Octave[5] (non-commercial). In the following, we address some of them.

The Control System Professional Suite from Wolfram Research extends Mathematica with a wide range of applications and numerical routines for CACSD. For Maple there exists MapleSim, DynaFlexPro, BlockBuilder for Simulink, etc.

MathWorks provides a number of CACSD toolboxes for MATLAB, for example the Control System Toolbox, Robust Control Toolbox, System Identification Toolbox and the complementary modeling and simulation software Simulink. In line of enhancing the numerical quality of CACSD software, a cooperate agreement between NICONET and Mathworks has been arranged which has replaced some of the previous routines in the Control System Toolbox with the more robust routines in SLICOT.

Another toolbox for MATLAB is the *Descriptor Systems Toolbox* [112] from DLR. It extends the MATLAB Control System Toolbox with the ability to handle descriptor systems and manipulate rational and polynomial matrices. The Toolbox is based on the RASP-DESCRIPT [110] routines and partly on the SLICOT library.

A MATLAB toolbox for polynomials and polynomial matrices is the *Polynomial Toolbox* for MATLAB by PolyX. It extends MATLAB with polynomial methods for systems, signals, and control analysis and design.

CACSD is a great source of applications for matrix equations including different eigenvalue and subspace problems, and condition estimation. In the following, we address some libraries and toolboxes that can be of interest for the reader.

HAPACK [14, 15] is a package of Fortran and MATLAB routines for (Skew-) Hamiltonian Eigenvalue Problems.

The upcoming *Matrix Equation Sparse Solver* (MESS) [19] is a MATLAB toolbox for solving large sparse matrix equations. The toolbox is the successor to the LyaPack Toolbox and includes solvers for the Lyapunov and the Riccati equations, and routines for model-reduction.

RECSY and SCASY [60] are state-of-the-art high-performance computing (HPC) software for solving Sylvester-type matrix equations. SCASY [62, 63] is a parallel ScaLAPACK-style library for Sylvester-type matrix equations. It includes solvers for 44 sign and transpose variants of eight different matrix equations, including continuous-time as well as discrete-time standard and generalized Sylvester and Lyapunov equations. SCASY employs the RECSY library [74, 75] to solve the matrix equations. The RECSY library uses recursion and OpenMP for shared memory parallelism.

For periodic systems there exist the *Periodic Systems* and *Periodic Eigenvalue Problem* (PEP) Toolboxes for MATLAB. The Periodic system toolbox [114, 116] is

[3] Maple is a registered trademark of Waterloo Maple Inc.
[4] Mathematica is a registered trademark of Wolfram Research, Inc.
[5] GNU Octave is freely redistributable under the terms of the GNU General Public License (GPL). Binaries and source code can be downloaded at Octave's web site http://www.octave.org.

based on the RASP-PERIODIC package and routines in SLICOT, complemented with recently developed MATLAB functions. The PEP Toolbox [64] is a set of Fortran and MATLAB routines for product and periodic eigenvalue problems. The toolbox is currently under development.

For the purpose to determine and present a stratification, the software tool *StratiGraph* has been developed [46, 71, 73]. The software is free for non-commercial usage and is under continued development, where the most recent version[6] can be found on StratiGraph's web site http://www.cs.umu.se/research/nla/singular_pairs/stratigraph/. StratiGraph is implemented in Java. The current version has support for stratification of matrices, matrix pencils and matrix pairs. From version 2.0, it is possible to load plug-ins which can extend StratiGraph with new problem setups and new functionalities. New possible problem setups are stratification of matrix triples and matrix quadruples and an already existing functionality extension is a MATLAB interface. The StratiGraph developer's guide [73] describes how to implement new plug-ins.

The *Matrix Canonical Structure* (MCS) Toolbox for MATLAB [72] provides in the current pre-release[7] (v 0.2) routines to compute, manipulate and present the canonical structures of matrices, matrix pencils, and matrix pairs. The aim of the MCS Toolbox is to provide a good understanding about the canonical structure of a broad range of setups. As StratiGraph now is an integrated part of the toolbox, the toolbox can give both *quantitative information* from the Matlab functions, and *qualitative information* from StratiGraph's graphical presentation. The Matlab functions are based on GUPTRI algorithms and are extended with newly developed functions, e.g., to impose canonical structures on input data and to compute the distance between two nearby canonical structures. StratiGraph can import canonical structures as well as quantitative data, in form of upper and lower bounds to nearby structures, from Matlab and present it together with the qualitative information.

The GUPTRI software package provides routines to compute the GUPTRI (generalized Schur-staircase) forms of general matrix pencils with error bounds [38, 39]. The GUPTRI package, which is implemented in Fortran 77, can be found on the web site http://www.cs.umu.se/research/nla/singular_pairs/guptri/. Gateway functions to access the GUPTRI Fortran functions from MATLAB is also provided on the web site. Similar routines to compute staircase forms can be found in the SLICOT library and the Descriptor Systems Toolbox for MATLAB.

[6] Current version is v. 2.2. Version 3 is under development and will be available in the near future.

[7] The MCS Toolbox is currently under development and will be made available at the same web site as StratiGraph.

Chapter 7

Summary of Contributions

The papers in the Thesis are categorized into two parts, where each part deals with two separate types of problems. Part I consists of two papers presenting theory and algorithms for stratification of orbits and bundles of matrix and system pencils. Part II consists of two papers on numerical methods for solving periodic Riccati differential equations.

7.1 Part I – Canonical structure information and stratification

When analyzing a state-space system, the canonical structure of the associated system pencil is of great interest. Examples include the computation of the controllability and observability characteristics, which are two fundamental concepts in systems and control theory (see Papers I and II). In Paper I, two major forms to represent the canonical structure of a pencil are presented: the Kronecker canonical form [54] and the (generalized) Brunovsky canonical form [28]. These canonical forms are intended and well suited for theoretical analyses but should not be used in practice. Instead, when computing the canonical structure information so called staircase-type forms are used [10, 38, 39, 76, 80, 104]. These forms are computed using only orthogonal (unitary) transformation matrices and backward stable algorithms. A brief introduction to different staircase-type forms are given in Paper I.

When computing different characteristics of a system, like its canonical structure information, small changes in some data can drastically change the computed results. These small perturbations can for example arise from noise in the system or from the well known fact that computers use finite-precision arithmetics. Therefore, it is important to understand how system characteristics change under small perturbations. The qualitative information about nearby systems is revealed by the theory of stratification [43, 44, 46]. More precisely, a stratification gives the closure hierarchy of orbits and bundles of matrices, matrix pencils, or system pencils. For a detailed explanation we refer to Paper I, where an comprehensive introduction to stratification of orbits and bundles of matrices, matrix pencils and system pencils is given, which is the major contribution of Paper I.

The second major contribution of the Thesis is the derivation of the stratification rules for orbits and bundles of controllability pairs (A,B) and observability pairs (A,C), associated with the particular systems $\dot{x}(t) = Ax(t) + Bu(t)$ and $\dot{x}(t) = Ax(t)$,

$y(t) = Cx(t)$, respectively, of a state-space system. In contrast to previous known results on stratification of general matrices and matrix pencils, these new stratification rules consider the special block structure of the controllability and observability system pencils. The stratification of matrix pairs provide a qualitative feedback on nearby (un-)controllable and (un-)observable systems. The rules are derived in Paper II and also presented in Paper I.

Paper I ends with an example illustrating the stratification of controllability and observability pairs associated with a general state-space system. In Paper II, we illustrate the stratification rules on two examples from system and control applications: a mechanical system and a linearized model of a Boeing 747. The two examples demonstrate how the qualitative information that the stratification provides can be used to analyze the characteristics of an LTI system. For example, together with bounds on the distance to uncontrollability we can validate the robustness of the system. Moreover, in both examples, the individual system matrices in the state-space models have a fixed structure. We show how the fixed structure can be taken into account when the stratification graphs are determined. It follows that only parts of the graphs (for unrestricted system matrices) exist due to these structural restrictions.

7.2 Part II – Periodic Riccati differential equations

This part deals with an open problem; The development of algorithms and robust methods for solving the periodic Riccati differential equation (PRDE), which, e.g., arise from the linear quadratic regulator (LQR) problem for periodic linear time-varying systems. Existing methods have been unreliable and not suited for systems of high order or with large period. One such method is the one-shot generator method [69, 119], which is an invariant subspace approach. Recently, two new methods have been suggested that better cope with these problems. The third contribution of this Thesis is an extensive evaluation of these new methods.

The first method considered is the multi-shot method [115, 117], which is based on the one-shot method but uses discretization techniques and turns the continuous-time problem into an equivalent discrete-time problem. The method computes the transition matrices for a linear Hamiltonian system, and the solution of the PRDE is then computed from the stable invariant subspace of those transition matrices using the ordered periodic real Schur form. In Paper III, the one-shot and the multi-shot methods are evaluated on an artificial constructed problem with known solution, and a problem originating from a nonlinear stabilizing problem for a devil stick juggling model along a periodic trajectory. Paper III also demonstrates the importance of using a symplectic method to solve the linear Hamiltonian system. The one-shot and multi-shot methods have been implemented in MATLAB by the author of this Thesis.

An alternative method to the ordered periodic Schur form has also been proposed in [117]. This method implicitly construct a stable deflating subspace from an associated lifted pencil using the fast algorithm, which is an extension of the swapping and collapsing approach [11, 12]. We call the multi-shot method using the fast method for the fast multi-shot method.

The second proposed method for solving the PRDE attacks the problem with a completely different approach. Instead of solving an ODE it reformulates the problem into a semidefinite programming (SDP) problem with linear matrix inequality (LMI) constraints [49]. The stabilizing solution of the PRDE and its derivative are approximated by finite dimensional trigonometric functions. It follows that one advantage of this approach is that the base functions can be chosen such that the characteristics of the underlying system is emphasized. We call this method the SDP method.

In Paper IV, we have evaluated the multi-shot, the fast multi-shot, and the SDP methods. They are all tested on a set of artificial constructed problems with known solutions, and the impact of the size (order) and the periodicity of the systems are investigated. The solvers are then evaluated on two stabilizing problems originating from two experimental control systems: Orbital stabilization of the Furuta pendulum and synchronization of oscillations of pendulums on carts.

The results of computational experiments in Papers III and IV are summarized as follows. The one-shot method should only be used together with a symplectic ODE solver on small-sized systems with short periods. The multi-shot solvers, on the other hand, are less sensitive to the choice of ODE solver and can be used also for medium-sized systems. For small-sized systems with large period the SDP solver is to prefer. However, the memory requirement for the SDP solver increases rapidly together with the size of the system. This fact restricts the size of the PRDE system that can be solved effectively. In our evaluation, this limit is already attained for medium-sized problems.

References

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank. *Matrix Riccati Equations: in Control and Systems Theory*. Birkhäuser, Basel, Switzerland, 2003. ISBN 3-7643-0085-X.

[2] B. Anderson and Y. Feng. An iterative algorithm to solve periodic Riccati differential equations with an indefinite quadratic term. In *Proc. of the 47th IEEE Conference on Decision and Control, CDC'08*, Cancun, Mexico, 2008.

[3] B. Anderson and J. Moore. *Optimal Control: Linear Quadratic Methods*. Dover, 2007. ISBN 0486457664.

[4] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenny, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, 3rd edition, 1999.

[5] A. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM Publications, Philadelphia, PA, 2005. ISBN 978-0-898716-58-0.

[6] P. Antsaklis and A. Michel. *Linear Systems*. Birkhäuser, Boston, 2nd edition, 2006. ISBN 978-0-8176-4434-5.

[7] M. Arnold and B. Datta. Single-input eigenvalue assignment algorithms: A close look. *SIAM J. Matrix Anal. Appl.*, 19(2):444–467, 1998.

[8] V. I. Arnold. On matrices depending on parameters. *Russian Math. Surveys*, 26:29–43, 1971.

[9] V. Balakrishnan and L. Vandenberghe. Algorithms and software for LMI problems in control. *IEEE Control Syst. Mag.*, 17(5):89–95, 1997.

[10] T. Beelen and P. Van Dooren. Computational aspects of the Jordan canonical form. In M. Cox and S. Hammarling, editors, *Reliable numerical computations*, pages 57–72. Clarendon Press, Oxford, 1990.

[11] P. Benner and R. Byers. Evaluating products of matrix pencils and collapsing matrix products. *Numer. Linear Algebra Appl.*, 8:357–380, 2001.

[12] P. Benner, R. Byers, R. Mayo, E. S. Quintana-Orti, and V. Hernandez. Parallel algorithms for LQ optimal control of discrete-time periodic linear systems. *Journal of Parallel and Distributed Computing*, 62:306–325, 2002.

[13] P. Benner, R. Freund, D. Sorensen, and A. Varga, editors. *Special Issue on Order Reduction of Large-Scale Systems*. Elsevier, 2006. Vol. 415, issues 2–3 of Linear Algebra Appl.

[14] P. Benner and D. Kressner. New Hamiltonian eigensolvers with applications in control. In *Proc. of 44th IEEE Conference on Decision and Control and European Control Conference, CDC-ECC'05*, pages 6551–6556, 2005.

[15] P. Benner and D. Kressner. Algorithm 854: Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices II. *ACM Trans. Math. Softw.*, 32(2):352–373, 2006.

[16] P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. *Numer. Linear Algebra Appl.*, 15(9):755–777, 2008.

[17] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT — A subroutine library in systems and control theory. In B. Datta, editor, *Applied and computational control, signals, and circuits*, pages 505–546. Birkhäuser Verlag Basel, 1999.

[18] P. Benner, V. Mehrmann, and D. Sorensen, editors. *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin/Heidelberg, 2005.

[19] P. Benner, H. Mena, and J. Saak. M.E.S.S. 1.0 user manual. Technical report, Chemnitz Scientific Computing, TU Chemnitz, 2008. To appear.

[20] P. Benner, H. Mena, and J. Saak. On the parameter selection problem in the Newton-ADI iteration for large-scale Riccati equations. *Electronic Transactions on Numerical Analysis*, 29:136–149, 2008.

[21] S. Bittanti, P. Colaneri, and G. De Nicolao. The periodic Riccati equation. In S. Bittanti, A. J. Laub, and J. C. Willems, editors, *The Riccati Equation*, chapter 6, pages 127–162. Springer-Verlag, Berlin, 1991.

[22] S. Bittanti, A. Laub, and J. Willems, editors. *The Riccati equation*. Springer-Verlag, Berlin, 1991.

[23] A. Bojanczyk, G. H. Golub, and P. Van Dooren. The periodic Schur decomposition; algorithm and applications. In F. T. Luk, editor, *Proc. SPIE Conference*, volume 1770, pages 31–42, 1992.

[24] D. Boley. *Computing the controllability/observability of a linear time-invariant dynamic system: A numerical approach*. PhD thesis, Computer Science Department, Stanford University, Stanford, CA, June 1981.

[25] D. Boley and W. Lu. Measuring how far a controllable system is from an uncontrollable one. *IEEE Trans. Autom. Contr.*, AC-31(3):249–251, 1986.

[26] S. Boyd, L. Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM studies in applied mathematics. SIAM, Philadelphia, 1994. ISBN 0-89871-334-X.

[27] S. Boyd and L. Vandenberghe. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[28] P. Brunovsky. A classification of linear controllable systems. *Kybernetika*, 3(6):173–188, 1970.

[29] A. Bunse-Gerstner, R. Byers, V. Mehrmann, and N. K. Nichols. Feedback design for regularizing descriptor systems. *Linear Algebra Appl.*, 299:119–151, 1999.

[30] J. V. Burke, A. S. Lewis, and M. L. Overton. Pseudospectral components and the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 26(2):350–361, 2004.

[31] R. Byers. A bisection method for measuring the distance of a stable matrix to the unstable matrices. *SIAM J. Sci. Stat. Comput.*, 9:875–881, 1988.

[32] C. Canudas-de Wit, J. Perram, and A. Shiriaev. Constructive tool for orbital stabilization of underactuated nonlinear systems. *IEEE Trans. Autom. Contr.*, 50(8):1164–1176, 2005.

[33] E.K.W. Chu, H.Y. Fan, W.W. Lin, and C.S. Wang. Structure-preserving algorithms for periodic discrete-time algebraic Riccati equations. *Int. J. Control*, 77:767–788, 2004.

[34] J. Clotet, M. I. García-Planas, and M. D. Magret. Estimating distances from quadruples satisfying stability properties to quadruples not satisfying them. *Linear Algebra Appl.*, 332–334:541–567, 2001.

[35] B. Datta. *Numerical methods for linear control systems*. Elsevier, New York, 2004. ISBN 0-12-203590-9.

[36] J. Demmel. A lower bound on the distance to the nearest uncontrollable system. Technical report, Computer Science Department, Courant Institute, New York, 1987.

[37] J. Demmel and B. Kågström. Accurate solutions of ill-posed problems in control theory. *SIAM J. Matrix Anal. Appl.*, 9(1):126–145, January 1988.

[38] J. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: Robust software with error bounds and applications. Part I: Theory and algorithms. *ACM Trans. Math. Software*, 19(2):160–174, June 1993.

[39] J. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: Robust software with error bounds and applications. Part II: Software and applications. *ACM Trans. Math. Software*, 19(2):175–201, June 1993.

[40] J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling. A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Softw.*, 16(1):1–17, 1990.

[41] J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson. Algorithm 656: An extended set of Fortran basic linear algebra subprograms. *ACM Trans. Math. Softw.*, 14(1):1–32, 1988.

[42] Y. Ebihara. Computing the distance to uncontrollability via LMIs: Lower bound computation with exactness verification. *Systems & Control Lett.*, 57:763–771, 2008.

[43] A. Edelman, E. Elmroth, and B. Kågström. A geometric approach to perturbation theory of matrices and matrix pencils. Part I: Versal deformations. *SIAM J. Matrix Anal. Appl.*, 18:653–692, 1997.

[44] A. Edelman, E. Elmroth, and B. Kågström. A geometric approach to perturbation theory of matrices and matrix pencils. Part II: A stratification-enhanced staircase algorithm. *SIAM J. Matrix Anal. Appl.*, 20:667–669, 1999.

[45] R. Eising. Between controllable and uncontrollable. *Systems Control Lett.*, 4:263–264, 1984.

[46] E. Elmroth, P. Johansson, and B. Kågström. Computation and presentation of graph displaying closure hierarchies of Jordan and Kronecker structures. *Numer. Linear Algebra Appl.*, 8(6–7):381–399, 2001.

[47] E. Elmroth and B. Kågström. The set of 2-by-3 matrix pencils — Kronecker structures and their transitions under perturbations. *SIAM J. Matrix Anal. Appl.*, 17(1):1–34, 1996.

[48] A. Emami-Naeini and P. Van Dooren. Computation of zeros of linear multivariable systems. *Automatica*, 18(4):415–430, 1982.

[49] L. Freidovich, S. Gusev, and A. Shiriaev. LMI approach for solving periodic matrix Riccati equation. In *Proc. of the 3rd IFAC Workshop on periodic control systems, PSYCO'07*, St Petersburg, Russia, 2007.

[50] L. Freidovich, S. Gusev, and A. Shiriaev. Computing a transverse linearization for mechanical systems with two and more underactuated degrees of freedom. Submitted, 2008.

[51] L. Freidovich, R. Johansson, A. Robertsson, and A. Shiriaev. Generating stable propeller motions for devil stick. In *Proc. of 3rd IFAC LHMNLC'06*, Nagoya, Japan, July 2006.

[52] L. Freidovich, R. Johansson, A. Robertsson, and A. Shiriaev. Periodic motions of the Pendubot via virtual holonimic constraints. *Automatica*, 44(3):785–791, 2008.

[53] L. Freidovich, P. La Hera, U. Mettin, and A. Shiriaev. New approach for swinging up the Furuta pendulum: Theory and experiments. Submitted to *IFAC Journal of Mechatronics*, 2009.

[54] F. Gantmacher. *The theory of matrices, Vol. I and II (transl.)*. Chelsea, New York, 1959.

[55] G. Golub and J. H. Wilkinson. Ill-conditioned eigensystems and the computation of the Jordan canonical form. *SIAM Review*, 18(4):578–619, 1976.

[56] G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[57] C. Gomez, editor. *Engineering and Scientific Computing with Scilab*. Birkhäuser, Boston, 1999.

[58] G. Goodwin, S. Graebe, and M. Salgado. *Control System Design*. Prentice Hall, 2001. ISBN 0-13-958653-9.

[59] J-M. Gracia and I. De Hoyos. Nearest pair with more nonconstant invariant factors and pseudospectrum. *Linear Algebra Appl.*, 298:143–158, 1999.

[60] R. Granat, I. Jonsson, and B. Kågström. RECSY and SCASY library software: Recursive blocked and parallel algorithms for Sylvester-type matrix equations with some applications. In R. Ciegis et al., editor, *Parallel Scientific Computing and Optimization*, Optimization and Its Applications, Vol. 27, pages 3–24. Springer, 2008.

[61] R. Granat and B. Kågström. Direct eigenvalue reordering in a product of matrices in periodic Schur form. *SIAM J. Matrix Anal. Appl.*, 28(1):285–300, 2006.

[62] R. Granat and B. Kågström. Parallel solvers for Sylvester-type matrix equations with applications in condition estimation. Part I: Theory and algorithms. Submitted to *ACM TOMS*, June 2007.

[63] R. Granat and B. Kågström. Parallel solvers for Sylvester-type matrix equations with applications in condition estimation. Part II: The SCASY software library. Submitted to *ACM TOMS*, June 2007.

[64] R. Granat, B. Kågström, and D. Kressner. Matlab tools for solving periodic eigenvalue problems. In *Proc. of the 3rd IFAC Workshop, PSYCO'07*, St Petersburg, Russia, 2007.

[65] L. Grasedyck. Nonlinear multigrid for the solution of large-scale Riccati equations in low-rank and Hscr-matrix format. *Numer. Linear Algebra Appl.*, 15(9), 2008.

[66] M. Gu. New methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 21(3):989–1003, 2000.

[67] M. Gu, E. Mengi, M. L. Overton, J. Xia, and J. Zhu. Fast methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 28(2):477–502, 2006.

[68] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-preserving algorithms for ordinary differential equations*. Springer-Verlag, Berlin, 2nd edition, 2006. ISBN 3-540-30663-3.

[69] J. J. Hench, C. S. Kenney, and A. J. Laub. Methods for the numerical integration of Hamiltonian systems. *Circuits Systems Signal Process*, 13(6):695–732, 1994.

[70] J. J. Hench and A. J. Laub. Numerical solution of the discrete-time periodic Riccati equation. *IEEE Trans. Autom. Contr.*, 39(6):1197–1209, 1994.

[71] P. Johansson. StratiGraph User's Guide. Technical Report UMINF 03.21, Department of Computing Science, Umeå University, Sweden, 2003.

[72] P. Johansson. Matrix Canonical Structure Toolbox. Technical Report UMINF 06.15, Department of Computing Science, Umeå University, Sweden, 2006.

[73] P. Johansson. StratiGraph Developer's Guide. Technical Report UMINF 06.14, Department of Computing Science, Umeå University, Sweden, 2006.

[74] I. Jonsson and B. Kågström. Recursive blocked algorithms for solving triangular systems. Part I: One-sided and coupled Sylvester-type matrix equations. *ACM Trans. Math. Softw.*, 28(4):392–415, 2002.

[75] I. Jonsson and B. Kågström. Recursive blocked algorithms for solving triangular systems. Part II: Two-sided and generalized Sylvester and Lyapunov matrix equations. *ACM Trans. Math. Softw.*, 28(4):416–435, 2002.

[76] B. Kågström. Singular matrix pencils. In Z. Bai, J. Demmel, A. Dongarra, J. Ruhe, and H. van der Vorst, editors, *Templates for the solution of algebraic eigenvalue problems: A practical guide*, pages 260–277. SIAM, Philadelphia, 2000.

[77] B. Kågström and A. Ruhe. ALGORITHM 560: An algorithm for the numerical computation of the Jordan normal form of a complex matrix [F2]. *ACM Trans. Math. Softw.*, 6(3):437–443, 1980.

[78] M. Karow and D. Kressner. On the structured distance to uncontrollability. Technical Report 2008-19, Seminar for Applied Mathematics, ETH Zurich, Switzerland, 2008. Submitted to Systems & Control Lett.

[79] E. Klerk. *Aspects of semidefinite programming: Interior point algorithms and selected applications*. Applied optimization. Kluwer Academic Pub., Dordrecht, 2002. ISBN 1-402-00547-4.

[80] V. N. Kublanovskaya. On a method of solving the complete eigenvalue problem for a degenerate matrix (in russian). *Zh. Vychisl. Mat. Fiz.*, 6:611–620, 1966. (USSR Comput. Math. Phys., 6(4):1–16, 1968).

[81] P. La Hera. *Contributions to motion planning and orbital stabilization. Case studies: Furuta pendulum swing up, inertia wheel oscillations and biped robot walking*. Licentiate Thesis, Department of Applied Physics and Electronics, Umeå University, Sweden, 2008.

[82] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, Oxford, 1995.

[83] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic linear algebra subprograms for Fortran usage. *ACM Trans. Math. Softw.*, 5(3):308–323, 1979.

[84] J. Löfberg. Yalmip: A toolbox for modeling and optimization in MATLAB. In *Proc. of the 13th IEEE International Symposium on Computer Aided Control System Design, CACSD'04*, Taipei, Taiwan, 2004.

[85] J. Löfberg. Yalmip homepage. Automatic Control Laboratory, ETH Zurich, Switzerland, January 2009. http://control.ee.ethz.ch/˜joloef/yalmip.php.

[86] K. Lust. psSchur Homepage. Department of Mathematics, K.U.Leuven, Belgium, January 2009.
http://perswww.kuleuven.be/˜u0006235/ACADEMIC/r_psSchur.html.

[87] A. Mahindrakar and R. Banavar. A swing-up of the Acrobot based on a simple pendulum strategy. *Int. J. Control*, 78(6):424–429, 2005.

[88] V. Mehrmann. *The Autonomous Linear Quadratic Control Problem: Theory and Numerical Solution*, volume 163 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1991.

[89] V. Mehrmann and H. Xu. An analysis of the pole placement problem. I. The single-input case. *Electron. Trans. Numer. Anal.*, 4:89–105, 1996.

[90] V. Mehrmann and H. Xu. An analysis of the pole placement problem. II. The multi-input case. *Electron. Trans. Numer. Anal.*, 5:77–97, 1997.

[91] E. Mengi. *Measures for robust stability and controllability*. PhD thesis, Courant Institute of Mathematical Sciences, New York, NY, 2006.

[92] U. Mettin. *Applications of the virtual holonomic contraints approach: Analysis of human motor patterns and passive walking gaits*. Licentiate Thesis, Department of Applied Physics and Electronics, Umeå University, Sweden, 2008.

[93] A. S. Morse. Structural invariants of linear multivariable systems. *SIAM J. Control*, 11(3):446–465, 1973.

[94] S. Nakaura, Y. Kawaida, T. Matsumoto, and M. Sampei. Enduring rotatory motion control of Devil stick. In *Proc. of the 6th IFAC NOLCOS Symposium*, pages 1073–1078, Stuttgart, Germany, 2004.

[95] G. Obinata and B. Anderson. *Model Reduction for Control System Design*. Communications and Control Engineering. Springer-Verlag, London, UK, 2001. ISBN 978-1-85233-371-3.

[96] C. Paige. Properties of numerical algorithms related to computing controllability. *IEEE Trans. Autom. Contr.*, AC-26(1):130–138, 1981.

[97] R. V. Patel, A. J. Laub, and P. Van Dooren, editors. *Numerical linear algebra techniques for systems and control*. Reprint Book Series. IEEE Press, New York, 1994.

[98] P. Hr. Petkov, N. D. Christov, and M. M. Konstantinov. *Computational methods for linear control systems*. Prentice Hall, Hertfordshire, UK, 1991. ISBN 0-13-161803-2.

[99] H. H. Rosenbrock. *State-space and multivariable theory*. Wiley, New York, NY, 1970.

[100] SeDuMi homepage. Advanced Optimization Laboratory, McMaster University, Canada, January 2009. http://sedumi.ie.lehigh.edu/.

[101] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.

[102] J. Sturm. Using SeDuMi 1.02, a Matlab toolbox for optimizaition over symmetric cones (updated for version 1.05). Technical report, Department of Econometrics, Tilburg University, Tilburg, The Netherlands, 2001.

[103] J. S. Thorp. The singular pencil of a linear dynamical system. *Int. J. Control*, 18(3):577–596, 1973.

[104] P. Van Dooren. The computation of Kronecker's canonical form of a singular pencil. *Linear Algebra Appl.*, 27:103–141, 1979.

[105] P. Van Dooren. The generalized eigenstructure problem in linear system theory. *IEEE Trans. Autom. Contr.*, AC-26(1):111–129, 1981.

[106] P. Van Dooren. Numerical linear algebra for signals systems and control. Draft notes prepered for the Graduate School in Systems and Control, University of Louvain, Belgium, Spring 2003.

[107] P. Van Dooren and M. Verhaegen. On the use of unitary state-space transformations. In *Special Issue of Contemporary Mathematics in Linear Algebra and Its Role in Systems Theory*, volume 47. Amer. Math. Soc., Providence, R.I., 1985.

[108] S. Van Huffel, V. Sima, A. Varga, S. Hammarling, and F. Delebecque. High-performance numerical software for control. *IEEE Control Syst. Mag.*, 24:60–76, 2004.

[109] C. Van Loan. How near is a stable matrix to an unstable matrix? *Contemporary Math*, 47:465–477, 1985.

[110] A. Varga. Numerical algorithms and software tools for analysis and modelling of descriptor systems. In *Proc. of 2nd IFAC Workshop on System Structure and Control*, pages 392–395, Prague, Czechoslovakia, 1992.

[111] A. Varga. Computation of Kronecker-like forms of a system pencil: Applications, algorithms and software. In *Proc. of IEEE International Symposium on Computer Aided Control System Design, CACSD'96*, pages 77–82, Dearborn, MI, 1996.

[112] A. Varga. A descriptor systems toolbox for MATLAB. In *Proc. of IEEE International Symposium on Computer Aided Control System Design, CACSD'2000*, pages 150–155, Anchorage, Alaska, 2000.

[113] A. Varga. Numerical awareness in control. *IEEE Control Syst. Mag.*, 24:14–17, 2004.

[114] A. Varga. A periodic systems toolbox for MATLAB. Technical Report IB-515-04-33, Inst. Robotics and System Dynamics, 2004.

[115] A. Varga. On solving periodic differential matrix equations with applications to periodic system norms computation. In *Proc. of CDC'05*, Seville, Spain, 2005.

[116] A. Varga. A periodic systems toolbox for MATLAB. In *Proc. of 16th IFAC World Congress*, Prague, Czech Republic, July 2005.

[117] A. Varga. On solving periodic Riccati equations. *Numer. Linear Algebra Appl.*, 15(9):809–835, 2008.

[118] G. Verghese, P. Van Dooren, and T. Kailath. Properties of the system matrix of a generalized state-space system. *Int. J. Control*, 30(2):235–243, October 1979.

[119] V.A. Yakubovich. Linear-quadratic optimization problem and frequency theorem for periodic systems. *Siberian Mathematical Journal*, 27(4):181–200, 1986.

[120] F. Zhang. *The Schur Complement and Its Applications*. Numerical Methods and Algorithms. Springer-Verlag, New York, 2005. ISBN 978-0-387-24271-2.