TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Informationstechnische Regelung

Control of interconnected systems with distributed model knowledge

Frederik Deroo

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Prof. Dr.-Ing. Dr. rer. nat. Holger Boche

Prüfer der Dissertation:

1. Prof. Dr.-Ing. Sandra Hirche (schriftliche Beurteilung)
   Prof. Dr.-Ing./Univ. Tokio Martin Buss (mündliche Prüfung)

2. Prof. Dr. Michael Ulbrich

Die Dissertation wurde am 11.01.2016 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 22.08.2016 angenommen.

Abstract

The development of new and advanced communication technologies has sparked an extensive renewed interest in distributed control of large-scale interconnected systems. The overall system typically consists of a number of possibly heterogeneous subsystems that are physically connected, that influence each others' behavior and that need to achieve a common goal. While numerous methods to design distributed control laws already exist, these approaches are commonly based on a complete system model and a centralized design. For several reasons, such as for computational restrictions and privacy concerns, it is desirable that the control *design* of the distributed control law is itself performed in a distributed fashion where the model knowledge is distributed among the subsystems and no central entity is required. Similarly, state of the art stability analysis methods do not allow to make a decision on stability of the overall system without relying on the complete system model. Despite the fact that the distributed design and analysis have transparent benefits in terms of privacy preservation and required computational capabilities, there is only a limited number of theoretical results available. Therefore, there is an urgent need for the development of innovative and novel methods to address these problems from a completely distributed perspective.

The present thesis investigates three relevant open problems in relation to the analysis and control of interconnected systems: First, distributed stability analysis of interconnected linear time-invariant systems with local model information and without a centralized decision-maker. Second, distributed optimal control design for a sparse control law optimizing the linear quadratic regulator problem using only local model information. Third, the analysis of signal decay in interconnected dynamical systems along paths of the graph describing the interconnection topology.

The main contributions of this thesis address the three issues stated above in the following way. For the stability analysis, we introduce two novel distributed stability tests. The first one is based on vector Lyapunov functions, the second one on the Lyapunov inequality subject to structural constraints. The main idea behind our approach consists of formulating optimization problems that contain the respective stability condition as a constraint and then applying advanced and modern distributed optimization techniques to solve the problems. This allows the subsystems to make a decision on the stability of the overall system using only local model information. To address the optimal control design, we introduce a new approach based on simulated trajectories of the states and adjoint states to compute a gradient descent direction. The presented gradient method is analyzed thoroughly and distributed step size selection methods as well as distributed convergence guarantees are shown. Furthermore, we present an approach to guarantee stability of the closed loop. All mentioned aspects can be implemented without a centralized entity and using only local model information. Last, we consider signal decay in interconnected systems. In our analysis, we approximate each subsystem by a scalar representation to evaluate how the subsystems interact. We consider two forms of disturbances and analyze their propagation in the steady state behavior of the interconnected system. All results are illustrated and validated using numerical demonstrations and examples.

Zusammenfassung

Die Entwicklung von neuen und fortschrittlichen Kommunikationstechnologien hat ein extensives erneutes Interesse an verteilter Regelung von großen verbundenen Systemen entfacht. Das Gesamtsystem besteht typischerweise aus einer Zahl von möglicherweise heterogenen Subsystemen, die physikalisch verbunden sind, die sich gegenseitig im Verhalten beeinflussen und die ein gemeinsames Ziel erreichen müssen. Obwohl es bereits eine Vielzahl an Methoden zum Entwurf verteilter Regelungsgesetze gibt, basieren diese Ansätze üblicherweise auf einem vollständigen Systemmodell und einem zentralisierten Entwurf. Aus mehreren Gründen, wie Begrenzungen von Rechenleistung und Privatsphärebedenken, ist es erwünscht, dass der *Reglerentwurf* des verteilten Regelgesetzes selbst verteilt durchgeführt wird, wobei das Modellwissen auf die Subsysteme verteilt und keine zentrale Einheit notwendig ist. Genauso wenig erlauben Methoden auf dem Stand der Technik im Bereich der Stabilitätsanalyse, eine Entscheidung über die Stabilität des Gesamtsystems ohne das vollständige Systemmodell zu treffen. Obwohl der verteilte Entwurf und die verteilte Analyse erkennbare Vorteile bezüglich Erhaltung von Privatsphäre und notwendigen Rechnerkapazitäten haben, gibt es bisher nur eine begrenzte Zahl an theoretischen Ergebnissen. Deshalb gibt es einen dringenden Bedarf an innovativen und neuen Methoden, um diese Probleme aus einer vollständig verteilten Perspektive zu behandeln.

Die vorliegende Arbeit untersucht drei relevante offene Probleme in Bezug auf Analyse und Regelung von verbundenen Systemen: Der erste Punkt ist verteilte Stabilitätsanalyse von verbundenen linearen zeitinvarianten Systemen mit lokaler Modellinformation und ohne zentralen Entscheidungsträger. Zweitens behandelt sie den verteilten optimalen Reglerentwurf für dünnbesetzte Regelgesetze mit lokalem Modellwissen, die ein linear-quadratisches Regelungsproblem optimieren. Der dritte Aspekt ist die Analyse von Signalabklingen in verbundenen Systemen entlang des Graphen, der die Verbindungstopologie beschreibt.

Die Hauptbeiträge dieser Arbeit behandeln die drei oben genannten Probleme auf folgende Art und Weise. Für die Stabilitätsanalyse führen wir zwei neue verteilte Stabilitätstests ein. Der Erste basiert auf Vektor Lyapunov Funktionen, der Zweite auf der Lyapunov Ungleichung, die strukturellen Beschränkungen unterliegt. Die Hauptidee ist, dass wir Optimierungsprobleme formulieren, die die jeweilige Stabilitätsbedingung als Beschränkung beinhalten, und dann fortschrittliche und moderne Techniken der verteilten Optimierung anwenden, um die Probleme zu lösen. Dies erlaubt den Subsystemen, nur mit lokalen Modellinformationen eine Entscheidung über die Stabilität des Gesamtsystems zu fällen. Zur Behandlung des optimalen Reglerentwurfs führen wir einen neuen Ansatz ein, der auf simulierten Trajektorien des Zustandes und des adjungierten Zustandes basiert, um eine Gradientenabstiegsrichtung zu berechnen. Das vorgestellte Gradientenverfahren wird tief gehend untersucht und wir stellen verteilte Verfahren zur Schrittweitenauswahl und Sicherstellung der Konvergenz vor. Des Weiteren präsentieren wir einen Ansatz, um Stabilität des geschlossenen Kreises zu garantieren. Alle erwähnten Aspekte können ohne zentrale Einheit und nur mit lokaler Modellinformation implementiert werden. Als Letztes betrachten wir Signalabklingen in verbundenen Systemen. In unserer Analyse approximieren wir jedes Subsystem durch eine skalare Darstellung, um zu bewerten, wie die Subsysteme interagieren. Wir behandeln zwei Formen von Störungen und analysieren ihre Ausbreitung im eingeschwungenen Zustand des verbundenen Systems. Alle Ergebnisse sind durch numerische Beispiele illustriert und validiert.

Contents

Notations

Abbreviations

LTI	Linear time-invariant
ODE	Ordinary differential equation
DAE	Differential-algebraic equation
LQ	Linear quadratic
LQR	Linear quadratic regulator
LMI	Linear matrix inequality
DCNA	Distributed constrained Nesterov algorithm
MPC	Model predictive control
BB	Barzilai-Borwein
CG	Conjugate gradient
FR	Fletcher-Reeves
PW	Powell-Wolfe
SE	Standard Euler method
EBE	Event-based Euler method

Symbols

Main variables

$u(t)$	system input
$y(t)$	system output
$x(t)$	system state
t	time
A	state matrix
A_{ii}	local state matrix of subsystem i
A_{ij}	coupling state matrix from subsystem j to subsystem i
B	input matrix
B_i	input matrix of subsystem i

C	output matrix
C_i	output matrix of subsystem i
s	Laplace operator
N	number of subsystems
n	number of overall system states
n_i	number of states of subsystem i
m	number of overall system inputs
m_i	number of inputs of subsystem i
$\mathcal{G}_{\mathrm{s}} = (\mathcal{V}_{\mathrm{s}}, \mathcal{E}_{\mathrm{s}})$	directed system graph
$\mathcal{G}_{\mathrm{s,u}} = (\mathcal{V}_{\mathrm{s}}, \mathcal{E}_{\mathrm{s,u}})$	undirected system graph
$\mathcal{G}_{\mathrm{comp}} = (\mathcal{V}_{\mathrm{comp}}, \mathcal{E}_{\mathrm{comp}})$	computation graph
$\mathcal{G}_{\mathrm{control}} = (\mathcal{V}_{\mathrm{control}}, \mathcal{E}_{\mathrm{control}})$	control graph
$\mathcal{G}_{\mathrm{LMI}} = (\mathcal{V}_{\mathrm{LMI}}, \mathcal{E}_{\mathrm{LMI}})$	LMI graph
Σ_i	subsystem i
L	Lipschitz constant

Spaces

$\mathbb{R}^n$	n-dimensional Euclidian space
$\mathbb{R}^n_+$	space of n-dimensional vectors with non-negative entries
$\mathbb{R}^n_{++}$	space of n-dimensional vectors with positive entries
$\mathbb{R}^{n\times m}$	space of $n \times m$-dimensional matrices
$\mathbb{S}^n$	space of symmetric $n \times n$ matrices
$\mathbb{S}^n_+$	space of symmetric, positive semidefinite $n \times n$ matrices
$\mathbb{S}^n_{++}$	space of symmetric, positive definite $n \times n$ matrices

Operators

A^{T}	transpose of matrix A
A^{-1}	inverse of square matrix A
$\mathrm{tr}(A)$	trace of matrix A
I_n	identity matrix in $\mathbb{R}^{n\times n}$
$0_{n\times m}$	matrix in $\mathbb{R}^{n\times m}$ with all entries 0
1_n	vector in $\mathbb{R}^n$ with all entries 1
$\mathrm{diag}(A_1, A_2)$	(block)diagonal matrix with matrices A_1, A_2 on diagonal
$\mathrm{blockdiag}(A)$	(block)diagonal portion of matrix A with off-diagonal blocks set to zero
e_l	lth unit base vector, with 0 everywhere except the lth entry which is 1
$\mathrm{vec}(A)$	vectorization of matrix $A \in \mathbb{R}^{n\times m}$ into vector in $\mathbb{R}^{nm}$ by stacking columns
$\mathrm{mat}(A)$	reforming of a vector $A \in \mathbb{R}^{nm}$ into a matrix in $\mathbb{R}^{n\times m}$
$\mathrm{Re}(x)$	real part of complex number x
$[A_{ij}]$	matrix A consisting of blocks A_{ij}
$\lambda(A)$	set of eigenvalues of matrix A
$\lambda_m(A)$	minimum eigenvalue of matrix A
$\lambda_M(A)$	maximum eigenvalue of matrix A
$\sigma_M(A)$	largest singular value of matrix A

$x > 0$	equivalent to $x \in \mathbb{R}^n_{++}$.
$X \succ 0$	equivalent to $X \in \mathbb{S}^n_{++}$
$X \succeq 0$	equivalent to $X \in \mathbb{S}^n_{+}$
$X \prec 0$	equivalent to $-X \in \mathbb{S}^n_{++}$
$X \preceq 0$	equivalent to $-X \in \mathbb{S}^n_{+}$

List of Figures

List of Tables

1

Introduction

The purpose of this chapter is multifold. First, a short introduction to feedback control is provided. Then, we explain our definition of interconnected dynamical systems and of distributed approaches with regards to control design and stability analysis. Subsequently, the motivation for the three pillars of this thesis is given, namely distributed stability analysis, distributed control design, and signal decay analysis in interconnected systems. Finally, we provide an outline of this thesis and state its major contributions.

1.1 A brief introduction to feedback control

Feedback control theory is a field that combines engineering and mathematics with the goal of influencing the behavior of a dynamical system. Everyday examples for feedback control engineering are thermostats or air conditioning used to control the temperature in a room, cruise control in cars, and the float regulator in a toilet. The main concept of feedback control is measuring the output of a system and then comparing it to a desired reference via feedback. Then, the task of the controller is to steer the output towards the desired value. A schematic of a typical control loop is shown in Figure 1.1. The measured output y, possibly corrupted by a measurement noise n, is compared to the reference r. The purpose of the controller K is to alter the behavior of the system G, sometimes referred to as the plant, such that y attains the desired value in spite of a possible disturbance d. Applied to the example of air conditioning, the disturbance d could be an open window or an unusually high number of people, and the task of the air conditioning system (K) is to keep the temperature (y) of the room (G) at a desired value r.

The centrifugal governor used in steam engines is one of the earliest examples of a control mechanism that has been thoroughly analyzed. Its purpose is to regulate the speed of the

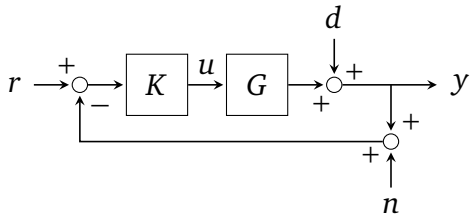


Figure 1.1: Basic control loop.

engine by influencing the throughput of a valve. This is achieved by coupling the rotation of the steam engine to a pair of rotating masses. With increasing speed, the two masses move outwards and act on a lever which reduces the opening of the valve. On the other hand, when the rotation slows down, the masses move inwards and open the valve further. The purpose of the governor as a part of the steam engine is, therefore, to be *self-limiting*, which is a result of the negative feedback.

The field of control theory is mainly concerned with two aspects: (1) The design of the feedback control law K to achieve the desired behavior. (2) Analysis of the system behavior, for example in terms of stability or performance. Control theory and automatic control, meaning control without requiring human intervention, have seen significant progress leading to a vast array of control design methods, system analysis approaches and many other related aspects. It is not only a discipline that improves the behavior of systems, but in many cases automatic control enables behaviors that are not possible with human control alone, for example with regards to precision or reaction speeds.

1.2 Interconnected dynamical systems

The tasks of control theory, e.g. control design and system analysis, become increasingly difficult when multiple systems or control loops are interconnected, as depicted in Figure 1.2. The interconnections between the systems, depicted by dashed lines, can represent two different characteristics. They can be either physical, meaning there is a direct influence between the system behaviors, or they can be purely through communication, e.g. when a common control law is used. When we discuss interconnected systems in this thesis, we consider the cases in which either one or both of these aspects are in place. Thus, the overall unifying property is that a number of subsystems in an overall system influence each other's state in some form. The number of subsystems also varies depending on the application, but it is very large in some fields.

Interconnected dynamical systems represent a class of systems that govern many aspects of our everyday lives. Numerous types of systems with practical relevance and with remarkably different properties belong to it. Examples from technical disciplines include infrastructure systems such as distribution systems for electrical power [1] and water [2], transportation infrastructure [3], building automation [4], mechatronic systems such as large-scale telescopes [5] and adaptive mechanical structures [6], the internet [7], and multi-robot systems [8]. Furthermore, interconnected systems can also represent biological [9], economical [10], or social networks [11], but these are not the focus of this thesis.

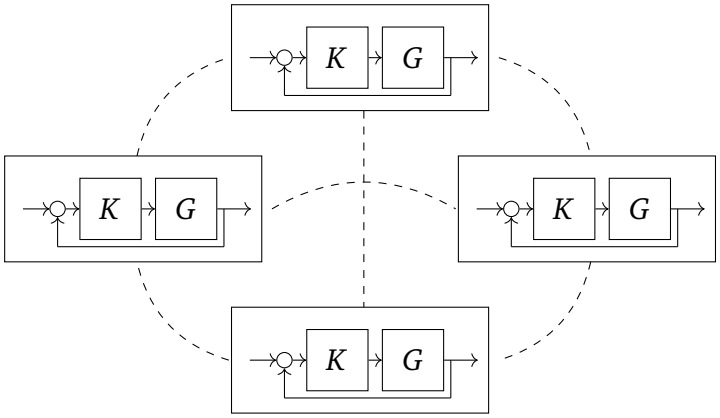


Figure 1.2: Interconnected system. Interconnections represented by dashed lines.

In order to explain interconnected systems in more detail, we provide two more concrete examples in the following.

Example 1.1. Electrical power system. Among the most important and complex interconnected dynamical systems is the electrical power system. The subsystems are comprised of numerous different components with highly differing behaviors, for example generators, loads, and transformers. These components are connected by transmission lines. There is a number of control goals that need to be met for a satisfactory overall behavior of the power system. Among others, these include stability of the frequency, stability of the voltage level, and control of active and reactive power. These goals need to be satisfied under uncertain conditions caused by faults, unpredictable consumer behavior, or variable production of renewable energy. For an in-depth treatment of the control of power systems, we refer to [12].

Example 1.2. Formation control. A second problem class related to interconnected dynamical systems is formation control. This problem can, for example, pertain vehicle platoons on a road, or unmanned aerial vehicles (UAVs). In the case of vehicle platoons, formation control is used to regulate and maintain the distance between vehicles at a desired value. The systems may be interacting physically, e.g. through wind drag, or their interconnection can be purely communicative. For UAVs, the latter is typically the case. That means that the interconnection stems from a control law used to achieve the common goal. In the case of UAVs, formation control is beneficial for localization, surveillance or exploration tasks. For more details, we refer to [13, 14].

In numerous cases of the aforementioned application areas, the considered systems are very large, with many of them still growing, and the complexity of the system dynamics increases with size. Therefore, the goal of many research efforts is to develop distributed methods for both control design and system analysis. Eventually, we aspire to individual components which are able to *self-organize* to achieve a common goal without or with very limited centralized intervention. Recalling the early purposes of control, we, as part of the control community, try to take a step from *self-limiting* behavior towards *self-organizing* behavior.

1.3 Analysis and control of interconnected dynamical systems

Traditionally, the control design and the analysis of the system class of interconnected systems are achieved from a centralized perspective. By centralized, we mean that full information about the dynamical model of the whole system is available to a single central entity, and the information can be used to analyze or control the overall system, as depicted in Figure 1.3.

Nevertheless, for many decades already, it has been the goal of research on interconnected systems to decompose the system into its components and to exploit the inherent structure in one form or another. Considering the structure of the control law, the control of interconnected systems is traditionally accomplished in a decentralized fashion [15]. In other words, every subsystem uses only local measurement signals to compute the control signal, and there is no measurement information exchange between the individual subsystems, see Figure 1.4a. This strategy is used, for example, in the primary control of electrical power grids, that is the lowest hierarchy in power system control [16], as well as in the package control of the internet [17]. Given the resource constraints, remarkable performance has been achieved with this approach. However, inherent problems of decentralized control are a lack of robustness, stability, and degraded performance in comparison to classical centralized full-information control. With the introduction of reliable, widespread, and fast communication networks came the advent of the research into distributed control. Distributed control is a compromise between centralized and decentralized control. Namely, the subsystems are able to exchange measurement information with some – but not all – other subsystems in the network, see Figure 1.4b. While a considerable effort has been invested into the research on designing distributed control laws, the design process itself is usually still performed from the original centralized perspective, as illustrated in Figure 1.3.

However, there is ongoing research, including this thesis, which investigates approaches that go one step further in terms of distribution. Instead of using the information exchange capabilities between the subsystems only for the online implementation of the control law, we also make use of the communication technology during the design of the control law itself, in order to eliminate the central designer. Thus, instead of taking the centralized point of view from Figure 1.3, we propose the scenario in Figure 1.5, where subsystems

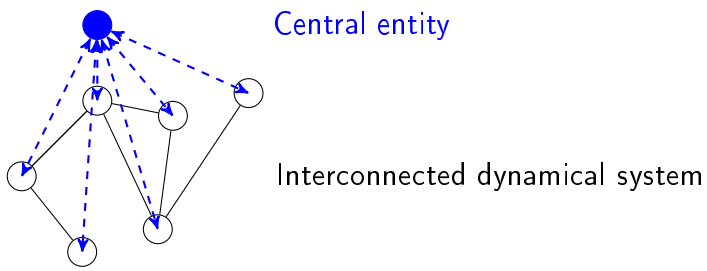


Figure 1.3: Traditional control and analysis approach for interconnected systems. Black (solid) lines indicate physical couplings, blue (dashed) lines indicate information flows, e.g. model data.

(a) Decentralized control.

(b) Distributed control.

Figure 1.4: Decentralized and distributed control. Black (solid) arrows indicate physical couplings and green (dashed) arrows denote control communication.

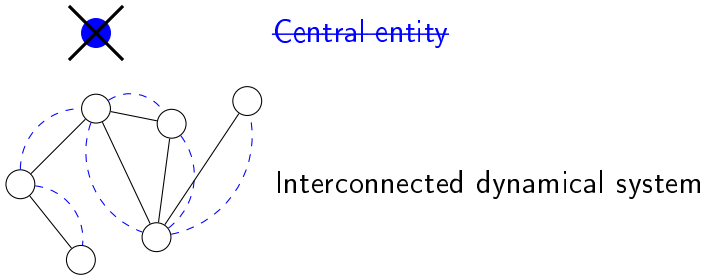


Figure 1.5: Proposed control and analysis approach for interconnected systems. Black (solid) lines indicate physical couplings, blue (dashed) lines indicate information flows, e.g. model data.

exchange information such as model data to design their control laws without a central entity. Furthermore, we adopt the distributed point of view from the control design problem to the system analysis problem, which is a frontier research direction with very limited results so far.

In summary, the main challenge addressed in this thesis is one of distributed decision-making. The subsystems have to make a judgment on a system property such as stability, or they need to decide how to set their control law to obtain a performance that is in accordance with a shared goal. This distributed perspective opens up a plethora of new and exciting research questions and we list a few in the following. With regards to the analysis of certain system properties and of the system behavior, the following questions are current and very relevant:

1. Is it possible to make a decision on whether a given system is stable using only local model information exchange? How general can the system dynamics be in order to allow a distributed stability analysis?

2. Is a distributed decision on stability necessarily conservative, and if so, how conservative is it?

3. Are there other system properties that can be evaluated distributedly?

4. How strong is the interaction between subsystems that are not directly connected? How does a disturbance propagate through a dynamical system?

In terms of optimal control design using local model information, the problems listed below are very timely and important:

5. Can we design optimal control laws in a distributed fashion with only local model information exchange?

6. Can closed-loop stability be guaranteed without central model information?

7. How good is the performance of these control laws in comparison to centralized control laws and distributed control laws that are designed with full model information?

8. Are there nodes that are more important than others with regards to performance, i.e. nodes that serve as hubs?

9. How can control laws be distributedly adapted to system dynamics that change over time or instantly? For example, what can be done when a subsystem leaves the overall system or a new one joins?

10. What is the optimal connection structure of the distributed control law? Should it be identical to the subsystem interconnection structure or should it be different?

As one important point of motivation for the use of only local model information is privacy of the subsystems, a question relevant for both system analysis and control design is the following:

11. What are meaningful notions of privacy? As agents need to interact with each other, subsystems may trust some of the other subsystems, but not all of them. How can one find a compromise with regards to privacy of signals and model information while using them for control purposes?

We do not claim to address all questions stated above. The purpose of the list of questions is to emphasize the wide variety of open problems in this field. Some of the questions are addressed in the literature, and more importantly in this thesis. After stating our main motivations in the next section, we provide a detailed explanation of the contributions of this thesis and how they relate to the problems stated above.

1.4 Motivation

The three pillars of this thesis are: (1) Distributed system analysis, (2) distributed optimal control design, and (3) decay analysis in interconnected systems. In this section, we motivate the research for the three parts individually. At a first glance, these three different aspects could in principle be regarded as being completely independent from one another. However, we will see throughout the thesis that there are close connections between them. For example, using some of the results from the first part leads to a significant enhancement

of the control design in terms of guaranteeing stability. Furthermore, the goal of the decay analysis is to give insight into the behavior of interconnected systems. This is also helpful for the first two aspects and it makes use of a stability notion used in the first part as well. At the same time, even though the interconnections are helpful and enrich our understanding of the respective topics, it needs to be emphasized that each of the three groups of results can stand on its own. We, therefore, present them individually such that the readers can pick the aspects relevant to their interests more easily.

1.4.1 Distributed stability analysis

There are several different well-known conditions available to check stability of a dynamical system, which can be found in various textbooks, e.g. [18, Chapter 4], [19, Chapter 3]. However, these conditions always assume that the whole model of the system dynamics is available for the analysis. This requirement, however, is undesired for large-scale interconnected systems. Testing stability distributedly and with local model information where there is no central model of the system and the subsystems only share their model data with a relatively small part of the overall system is important for several reasons. Our main motivation is model data privacy of the subsystems. This concern is relevant, for example, in a scenario where competitors need to form an interconnected dynamical system, as it is the case in an electrical power grid, but are hesitant to share their exact model data with a central entity or all other participants. The same concerns apply when more and more private participants take part in the overall system, as is also the case in the power grid. Another argument for the distribution of the analysis is that the computation effort for the system analysis can be distributed among the subsystems. Additionally, these distributed methods are generally more flexible when a system changes because only parts of the system need to adapt. Last, a distributed approach promises better scalability.

1.4.2 Distributed control design

New and distributed control design methods for interconnected systems are required because of the following three problems: (1) Due to the size of some of these systems, classical centralized control approaches with their inherent communication complexity are no longer suitable. On the other hand, as previously stated, decentralized approaches have several disadvantages, e.g. with regards to robustness and performance. Distributed control aims at a compromise between these two. (2) Not only the online control law implementations but also the centralized design methods themselves generally do not scale well and thus lack the flexibility that is required for large-scale systems. (3) As is the case for the system analysis, a key problem in the control design is that most of the known design approaches require a complete system model, which does not take into account the fact that subsystems might be unwilling to share their model data globally or centrally. To ensure the model data privacy, the design of the control law should be distributed. Control design where the model data is distributed among agents is still an open and challenging problem because model data is typically required for the design of optimal control laws and also to evaluate and guarantee stability.

1.4.3 Decay analysis in interconnected systems

One important question of interest for interconnected systems is under which conditions a disturbance at one node will have little impact on another node that is far away in terms of the number of hops between the nodes. The investigation of this problem is relevant for multiple reasons. First, it will help in understanding how disturbances, or changes in system dynamics at one subsystem, affect the behavior at distant nodes. In other words, it is of interest how the performance of a distant node is affected by local disturbances. This will lead to an intuition concerning how networks should be designed, e.g. with regards to disturbance attenuation, since it will clearly be preferred in most applications that local changes in dynamics do not have a large effect over large distances. Second, a property of this type would allow the agents more freedom even though they are connected. In addition, as privacy is an important issue, subsystems may not want all other participants in an interconnected system to be able to sense their state. Finally, it helps in the development or modification of a local control design method at one agent with local model information, as considered in the second part of the thesis, and it may enable local control law adaptations without significantly sacrificing overall system performance.

1.5 Outline and contribution

The goal of this thesis is to develop novel and innovative distributed methods that allow stability analysis and control design using only local model information without any centralized knowledge. The key foundation to these methods is the usage of modern distributed optimization techniques.

Chapter 2 consists of background information. In addition, the overall system class is introduced to illustrate and emphasize the common problem properties. The chapter also introduces the definition of *local model information*. The distributed stability analysis is investigated in Chapter 3. Chapter 4 addresses distributed control design. Afterwards, Chapter 5 presents results on signal decay in interconnected systems. Chapter 6 provides a brief summary, a concluding discussion, and presents open questions not addressed in this thesis. In the remainder of this section, the major contributions of each chapter are outlined in more detail.

Chapter 3: Distributed stability analysis with local model information

In this chapter, two different distributed tests for the stability analysis of interconnected linear time-invariant (LTI) systems are developed. The conditions are based on two well-known classical stability conditions: Vector Lyapunov functions and the Lyapunov inequality. The novelty of the developed approaches is that the inherent structure of the vector Lyapunov function and a prescribed structure in the Lyapunov inequality are explicitly exploited to achieve distributed tests requiring only local model information. In the first step, the original conditions are reformulated in the form of optimization problems. Subsequently, using the distributed constrained Nesterov algorithm, the optimization problems are solved in a distributed fashion, making use of the problem structure and thereby limiting the required

model knowledge to local model information. As the two developed tests are only sufficient, they are compared in terms of conservativeness, and the extent of their conservativeness is investigated using numerical experiments. In addition, we derive an analytical necessary condition for the second stability notion, the so-called α-block diagonal Lyapunov stability. Last, we show that it is possible to apply the same distributed optimization algorithm used in the stability analysis to perform distributed model reduction based on balanced truncation.

The content of this chapter addresses questions 1, 2, and 3 from Section 1.3. The results in this chapter have been partially published in [20].

Chapter 4: Stabilizing distributed optimal control design with local model information

This chapter presents a distributed optimal control design method for interconnected LTI systems that relies only on local model information. The design goal of the control design is the minimization of a linear quadratic cost functional with a specific control law structure. The approach is based on iterative distributed optimization using a gradient descent method. Using simulated trajectories of the state and the adjoint state, the gradient can be distributedly computed. Furthermore, the step size and the stopping criterion can also be evaluated by the agents in a distributed setting without a centralized decision-maker. Using results from Chapter 3, a terminal cost term that is determined in a distributed fashion is used to guarantee closed-loop stability of the approach. As the computation of the trajectories leads to a large communication effort, an event-based adaptation of the Euler discretization method is employed to reduce it. It is shown that the error of the method is on the same order as the standard Euler method while heavily reducing the communication effort. An extension of the control design method to singular systems is also presented. Then, the design approach is adapted to bi-quadratic terms in the cost functional and is applied to the problem of optimal formation control of a multi-robot system. Last, we generalize the method to time-varying control laws and nonlinear system dynamics. All results are validated and illustrated with numerical examples. The content of this chapter addresses questions 5, 6, and 7 from Section 1.3. The results in this chapter have been partly published in [21, 22, 23, 24, 25] and some results will be published in [26].

Chapter 5: Decay analysis in interconnected systems

The main contribution of this chapter is an analytical investigation of the decay between subsystems in response to a steady state disturbance. We consider the case of a constant input at a single node, and then look at the generalization to a sinusoidal input. It is shown that the value of the steady state response at a node can be bounded by a linear combination with positive weights of steady state values at nodes closer to the source node, and that the coefficients in that combination sum to less than one. The results are illustrated using numerical simulations. The content of this chapter mainly addresses question 4 from Section 1.3. The results of this chapter have been partly published in [27].

2

Background and related work

In this chapter, we provide necessary background information for this thesis. Especially for the first two topics addressed in this thesis, namely stability analysis and optimal control design, we summarize classical and well-established results and give an overview on relevant work related to the respective topic. In addition, we give an introduction to different privacy notions that are presented in the literature in relation to optimization and control, and contrast them to the model data privacy restrictions considered in this thesis. Then, we introduce the considered system dynamics, the graphical description to capture the system structure, and introduce the term *local model information* that is the central theme of this thesis. Last, we explain a distributed optimization algorithm that is used in several places in this thesis; the distributed constrained Nesterov algorithm.

2.1 Stability analysis of interconnected systems

In this section, we give some introductory background on stability analysis in general, and more specifically stability analysis for interconnected systems.

Stability analysis of dynamical systems is a problem that has been studied thoroughly and represents one of the central questions in the field of control. There is a large number of different stability conditions, appropriate for varying stability notions, system dynamics, or problem settings. For more details, we refer the reader to textbooks, such as [18, 28]. Probably the most seminal result, and the basis for the majority of modern stability analysis, especially in the nonlinear case, is Lyapunov stability [29]. Lyapunov stability states the following [30]: Consider the dynamical system

$$\dot{x} = f(x), \tag{2.1}$$

where $f : D \to \mathbb{R}^n$ is a locally Lipschitz map from a domain $D \subseteq \mathbb{R}^n$ into $\mathbb{R}^n$ and which has an equilibrium point $x = 0$, i.e. $f(0) = 0$. Then, if there is a function $V : D \to \mathbb{R}$ such that

$$V(0) = 0 \text{ and } V(x) > 0 \text{ in } D \setminus \{0\},$$
$$\dot{V}(x) < 0 \text{ in } D \setminus \{0\},$$

we have that $x = 0$ is asymptotically stable.

Throughout this thesis, instead of the nonlinear dynamics (2.1), we consider the special case of linear time-invariant (LTI) systems, generally written as

$$\dot{x}(t) = Ax(t), \tag{2.2}$$

where $x \in \mathbb{R}^n$ is the state and $A \in \mathbb{R}^{n \times n}$ is the state matrix. One way of showing Lyapunov stability of system (2.2) is to assume a Lyapunov function of the form $V(x) = x^\mathsf{T} P x$ and to find the positive definite matrix $P \in \mathbb{S}^n_{++}$ such that the matrix $A^\mathsf{T} P + PA$ is negative definite. In this linear case, Lyapunov stability is necessary and sufficient, and completely equivalent to the condition that requires negative real parts of all eigenvalues of the matrix A. There is a special case of Lyapunov stability of LTI systems that is relevant to this thesis, namely diagonal Lyapunov stability [31]. This notion simply states that the system (2.2) is diagonally Lyapunov stable if there is a *diagonal* matrix $P \in \mathbb{S}^n_{++}$ such that $A^\mathsf{T} P + PA$ is negative definite. In general, diagonal Lyapunov stability is clearly only a sufficient stability condition, because the solution space is a subspace of the original, complete solution space $\mathbb{S}^n_{++}$.

System analysis based on the complete overall system model (2.2), be it based on Lyapunov stability, eigenvalue analysis, or a different approach, works well for small and reasonably sized systems. However, for large-scale systems consisting of many individual subsystems, different and dedicated methods are required, as is explained in Section 1.4.1. Traditionally, large-scale dynamical systems are analyzed from a centralized point of view, assuming that there is a central entity with access to the whole system model or at least to the whole interconnection structure. A well-known result from the 1970s involves vector Lyapunov functions and M-matrices. Due to its relevance to this thesis, we shortly recapitulate the foundations of this concept [15].

While the overall interconnected system is described by (2.2), it is convenient to split up the model at the subsystem level. The division into subsystems is highly dependent on the application and the desired level of abstraction. It could result from spatial distance of components, from different functions of the components, or from different owners of the components, to name just a few reasons. Each subsystem has the dynamics

$$\dot{x}_i(t) = A_{ii} x_i(t) + \sum_{\substack{j=1 \\ j \neq i}}^{N} A_{ij} x_j(t),\ i = 1, ..., N, \tag{2.3}$$

where $x_i \in \mathbb{R}^{n_i}$ and $A_{ij} \in \mathbb{R}^{n_i \times n_j}$. Vector Lyapunov functions allow the analysis of the interconnected system to see if it is connectively stable which is defined as follows.

Definition 2.1. [15] System (2.2),(2.3) is *connectively stable* if it is stable in the sense of Lyapunov for all interconnection terms αA_{ij} with $\alpha \in [0, 1]$ and A_{ij} defined in (2.3).

Connective stability, therefore, implies that the system remains asymptotically stable even if interconnection strengths are changed to smaller values. The idea of vector Lyapunov functions is to construct a Lyapunov function for the overall interconnected system as a weighted sum of individual Lyapunov functions of the isolated subsystems $\dot{x}_{i,\text{iso}} = A_{ii}x_{i,\text{iso}}$. The main tool to determine whether a vector Lyapunov function exists are M-matrices.

Lemma 2.1. *[15] A matrix $W \in \mathbb{R}^{N\times N}$ with nonpositive off-diagonal elements is an M-matrix if there exists a vector $d \in \mathbb{R}^N$ with strictly positive entries such that $Wd > 0$.*

The following assumption is made in the literature in the context of vector Lyapunov functions [15].

Assumption 2.1. The decoupled individual subsystems are asymptotically stable, i.e. $\text{Re}(\lambda(A_{ii})) < 0$, and all eigenvalues of A_{ii} are distinct for all $i \in \{1,\dots,N\}$.

The regular transformation matrices $T_i \in \mathbb{R}^{n_i\times n_i}$ are defined which transform the isolated system i described by $\dot{x}_{i,\text{iso}} = A_{ii}x_{i,\text{iso}}$ into $\dot{\tilde{x}}_{i,\text{iso}} = \tilde{A}_{ii}\tilde{x}_{i,\text{iso}}$. Given the second part of Assumption 2.1, there is a T_i such that $\tilde{A}_{ii} = T_i^{-1}A_{ii}T_i$ has a real Jordan form and the transformation mapping is $x_{i,\text{iso}} = T_i\tilde{x}_{i,\text{iso}}$. This transformation is used because it reduces conservativeness of the stability condition [15]. With that, the following matrix $W = [w_{ij}]$ is constructed as

$$w_{ij} = \begin{cases} -\max(\text{Re}(\lambda(\tilde{A}_{ii}))), & \text{if } i = j, \\ -\sqrt{\lambda_M(\tilde{A}_{ij}^{\mathsf{T}}\tilde{A}_{ij})}, & \text{if } j \in \mathcal{N}_{\text{in},i}, \\ 0, & \text{else,} \end{cases} \tag{2.4}$$

where $\tilde{A}_{ij} = T_i^{-1}A_{ij}T_j$. Based on that, the following theorem is formulated.

Theorem 2.1. *[15] Given Assumption 2.1, the system (2.2),(2.3) is connectively stable if there is a vector $d > 0$ such that $Wd > 0$ with W defined in (2.4).*

Theorem 2.1 essentially formulates a sufficient stability condition which relates the ratio of stability of the local dynamics of each subsystem with the effect of the incoming interconnections expressed in the required M-matrix property. Testing a matrix to see if it is an M-matrix can be accomplished, for example, using linear programming which is, however, a global problem.

2.2 Optimal control design

We start this section with some necessary background information on optimal control design and introduce a general form of the considered problem. We then present an in-depth literature overview of work on distributed control.

2.2.1 Background

It can be argued that the ultimate understanding we can have of a system is when we are able to control it [32]. As explained in Section 1.1, achieving a desired behavior requires the

design of a feedback law, often denoted by K. There are numerous methods with various control goals available in the literature and we refer to textbooks such as [18, 28] for an overview on basic control design methods. An idea to avoid trial-and-error during the control design is to determine the feedback law in an optimal way where optimality is defined by a cost functional. Optimal control is quite an old topic of research, with first results stemming from more than 300 years ago [33]. There are various types of cost functionals such as H_2 or H_∞ optimality and there are closed-form or at least efficient solutions for many of them. We refer to the textbook [19] for details. These methods are thoroughly understood and work well at least for small-scale systems.

In this thesis, we restrict our attention to LTI systems, where the overall system dynamics are described by

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

and the goal of optimal feedback control design is to find a feedback law $u(t) = -Kx(t)$ that minimizes a given cost functional. One very common cost functional, especially for linear systems, and the cost functional considered throughout most of this thesis, is the linear quadratic (LQ) cost, typically written as

$$J(x,u) = x^\mathsf{T}(t_\mathrm{f})Sx(t_\mathrm{f}) + \int_0^{t_\mathrm{f}} x^\mathsf{T}(t)Qx(t) + u^\mathsf{T}(t)Ru(t)\mathrm{d}t, \tag{2.5}$$

where $Q \in \mathbb{S}^n_+$, $S \in \mathbb{S}^n_+$, $R \in \mathbb{S}^n_{++}$ and t_f is the optimization horizon. The matrices S, Q and R are weighting matrices which are used to describe the desired behavior. In this finite horizon form, the terminal cost term involving S penalizes the state at the end of the horizon, while Q and R respectively penalize the state and input over the whole period t_f. The infinite horizon form is obtained for the case that $t_\mathrm{f} \to \infty$ and is given by

$$J(x,u) = \int_0^{\infty} x^\mathsf{T}(t)Qx(t) + u^\mathsf{T}(t)Ru(t)\mathrm{d}t. \tag{2.6}$$

Solutions for both (2.5) and (2.6) are readily available with several extensions, see e.g. [34, 35]. For instance, the solution for the cost functional (2.6) can be computed as

$$K = R^{-1}B^\mathsf{T}P,$$

where $P \in \mathbb{S}^n_{++}$ is the solution to the Riccati equation

$$A^\mathsf{T}P + PA - PBR^{-1}B^\mathsf{T}P + Q = 0. \tag{2.7}$$

For the finite horizon case (2.5), the optimal feedback is time-varying with $K(t) = R^{-1}B^\mathsf{T}P(t)$ and we need to solve the Riccati differential equation

$$A^\mathsf{T}P(t) + P(t)A - P(t)BR^{-1}B^\mathsf{T}P(t) + Q = -\dot{P}(t),$$

where we have the boundary condition $P(t_\mathrm{f}) = S$. For other optimization goals such as H_2 and H_∞ similar solutions are available.

2.2.2 Relevant work on distributed optimal control

All the methods mentioned so far assume small-scale systems and take a completely centralized point of view. In these cases, the feedback law has no structure and full information is available at every component of the system. For large-scale interconnected systems, this is undesirable because of the lack of technological feasibility in terms of communication and computation effort, and because of the lack of scalability, among other things. The current main approach in the research community to deal with these problems is distributed control. Therefore, results on distributed control of interconnected systems are reviewed in the following. As mentioned in the previous section, distributed control is a compromise between decentralized and centralized control. For an overview on decentralized control, we refer to textbooks, e.g. [15], or the survey article [36].

Early work on distributed control in the 1960s [37] already illustrates the inherent difficulty of the problem by showing that even in a seemingly simple case of two linear agents, the optimal control law can be nonlinear when a certain information structure is imposed. Later, it was shown that for a partially nested structure, a property which describes that there are no cycles in the directed system interconnection graph, the optimal control law is linear [38], but a general result is still open. Therefore, research has concentrated on solutions to special cases of the overall problem of finding an optimal control law.

Quadratic invariance. One typical approach to find optimal control laws is to parameterize all possible controllers using so-called Youla-parameters [19]. For distributed control laws, it is necessary to impose sparsity constraints on these Youla parameters [39, 40, 41, 42, 43]. In [39], a sufficient condition called "Quadratic invariance" is derived under which the constraint on the parameters is convex, such that it is in principle possible to formulate the control design as a convex optimization problem. It is shown in [40] that, given the Youla parameterization approach, this condition of quadratic invariance is also necessary for a convex problem. For a special case of two subsystems, the explicit state space solution for the H_2 case is presented in [41]. These last results are generalized to systems that have a partial order, which corresponds to a special case of quadratic invariance, and explicit solutions as well as the optimal controller architecture to the H_2 design problem are stated in [42, 43].

Structured linear matrix inequalities. Other approaches use linear matrix inequalities (LMIs) with constraints on the solution variables to achieve the desired structure in the control laws [44, 45, 46, 47]. The approach in [44] states an LMI that results in a control law with H_∞ optimality that has the same structure as the original physical subsystem interconnection. For H_2 optimal controllers, LMIs are stated in [45] that give sparse control laws. For identical interconnected subsystems and a control law structure that coincides with the system structure, [46] gives LMI conditions that can be decomposed to allow a more efficient solution. Similarly, for identical decoupled systems with a coupling in the cost term, a small dimensional solution is given in [47].

Dual decomposition. Yet another approach is to use dual decomposition in the design of control laws, e.g. [48, 49]. In [48], the optimal control of vehicle systems is solved using dual

decomposition. The interconnection structure in this system class is quite simple in the sense that the subsystems are arranged on a line graph. For more general topologies [49] presents an optimal control design method that relies on the alternating direction method of multipliers (ADMM). In the latter, the optimization problem not only considers the performance of the control law but also attempts to optimize the sparsity. The use of dual decomposition, however, is mainly to improve the computational performance and it does not lead to a complete distribution.

Distributed adaptive control. A promising research direction related to distributed control that should be mentioned is distributed adaptive control. Most results so far concentrate on synchronization problems [50, 51]. The authors present adaptive control protocols to guarantee synchronization of systems whose dynamics are unknown. However, there is no attention to optimality and the design of the protocol is performed from a centralized perspective. Nevertheless, the results show that adaptive approaches are applicable to interconnected systems and give promising insights. A different approach, and more closely related to the work in this thesis is reinforcement learning [52] where adaptive control is used to improve a control law based on past online system behavior. However, no results for distributed control are available.

Topology design. The work in this overview so far assumes a fixed control law topology which in general is identical to the physical system topology. But there are also approaches that design the control law topology, e.g. [53, 54]. In [53], the control design is performed in two steps. First, a stabilizing decentralized control law is designed. In a second step, a limited number of edges is added. The edges are chosen to be optimal with respect to a cost functional which in this case is the maximum eigenvalue of the closed-loop system. However, this is computationally expensive because it is a mixed-integer problem. In [54], an approach is presented that minimizes the number of connections given an allowed performance deterioration in comparison to a full control law. A convex relaxation is used to make the approach more appealing in terms of the computational effort.

2.3 Privacy notions in control

The literature addresses several different notions of privacy, mainly to achieve privacy of the measured behavior of the subsystems, e.g. electrical power consumption of a consumer over time. The first main concept is *differential privacy* [55, 56, 57, 58]. Differential privacy is a statistical approach which generally assumes that a central entity aggregates the available data, e.g. signals or trajectories, and produces some form of output. A mechanism is differentially private when the addition, change or removal of a single piece of data does not lead to a large deviation in the output distribution. For more details on differential privacy and an overview of several results, see [55]. In [56], the authors apply the concept to a convex optimization problem and to them privacy preservation means that the agents do not give up their complete private cost function. Methods to design filters that preserve differential privacy are presented in [57]. The authors of [58] consider a control scenario of linear systems

with a quadratic cost functional where the individual agents are influenced by a common aggregate state. To them, privacy means that the state preference of the agents cannot be easily deduced. The common approach to achieve differential privacy is to add a noise signal to the sensitive data. A different concept of privacy uses observability arguments, e.g. [59]. Their approach uses graph theoretical ideas and topology design to ensure that as many nodes as possible are part of the unobservable subspace to preserve privacy of the initial conditions in a consensus problem. The addition of noise to achieve the same goal in an average consensus algorithm is used in [60]. In [61], the privacy of trajectories is investigated in a dynamic average consensus problem where an average of dynamic inputs is obtained while the agents add static or dynamic signals to their inputs to hide their true values. Two other approaches deal with privacy in smart grids. The first one [62] uses storage devices such as batteries to smoothen the revealed power consumption and hide specific usage profiles. Similarly, the authors of [63] propose delaying usage data signals and using different sampling periods to hide specific private usage events.

While privacy of signals and behaviors is certainly an important problem, model data privacy is also an interesting and relevant aspect that is often neglected. There is one result that considers privacy of dynamic model data which is given in [64]. This paper considers the scenario in which several scalar subsystems share a common scalar input and a common scalar output, and it presents methods to achieve differential privacy in terms of the individual system parameters.

In this thesis, however, we consider a different notion of privacy, namely local model information. This means that subsystems need to share their model data only with a limited subset of all nodes, possibly along a specific cooperation graph structure without global or central aggregation of model data.

2.4 System description

In this section, we present the general system dynamics that are considered in this thesis. We present the relevant differential equations and introduce the structural description of the system using several graphs.

2.4.1 Differential equation model

Throughout the majority of this thesis we consider interconnected LTI systems consisting of N subsystems. Each subsystem has the following dynamics

$$\dot{x}_i(t) = A_{ii}x_i(t) + \sum_{\substack{i=1 \\ i\neq j}}^{N} A_{ij}x_j(t) + B_i u_i(t), \tag{2.8}$$

where $x_i \in \mathbb{R}^{n_i}$ is the subsystem state, $u_i \in \mathbb{R}^{m_i}$ is the local input, $A_{ii} \in \mathbb{R}^{n_i \times n_i}$ represents local dynamics, $A_{ij} \in \mathbb{R}^{n_i \times n_j}$ denotes coupling to other subsystems and $B_i \in \mathbb{R}^{n_i \times m_i}$ is the input matrix. By concatenating the subsystem states, the overall system is compactly written as

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{2.9}$$

where $x = [x_1^\mathsf{T}, ..., x_N^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^n$, $u = [u_1^\mathsf{T}, ..., u_N^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^m$, $\sum_{i=1}^N n_i = n$ and $\sum_{i=1}^N m_i = m$.

Because the structure of large-scale systems in practice is often sparse [65], we also assume that system (2.9) has a sparsity structure, that is no subsystem is connected to every other subsystem.

The stated system dynamics are slightly adjusted for the stability analysis in Chapter 3 and for the decay analysis in Chapter 5 in that we remove the input $u(t)$ from the system equations (2.9). For most of the control design results in Chapter 4 the system dynamics remain unchanged. However, the results are then extended to DAE systems with an additional equality constraint, and later to general nonlinear systems.

2.4.2 Structural description using graphs

The structure of the system, and the structure of the control design and system analysis problems are described using graphs. For a short introduction to the relevant graph theoretical concepts for this thesis, please see Section A.2.

In the following, we define several graphs that are used throughout the thesis. The first one is the system graph which is defined as follows.

Definition 2.2. The *system graph* $\mathcal{G}_\mathrm{s} = (\mathcal{V}_\mathrm{s}, \mathcal{E}_\mathrm{s})$ describes the given physical, open-loop interconnection structure of the individual subsystems. An edge (j, i) belongs to the edge set $\mathcal{E}_\mathrm{s}$ if and only if the matrix $A_{ij} \neq 0$.

Hence, the graph $\mathcal{G}_\mathrm{s}$ describes the block structure of the state matrix A. Based on that, we define the following subsets of the system graph.

Definition 2.3. $\mathcal{N}_{\mathrm{out},i} = \{j | (i, j) \in \mathcal{E}_\mathrm{s}\}$ are the nodes j influenced by i, $\mathcal{N}_{\mathrm{in},i} = \{j | (j, i) \in \mathcal{E}_\mathrm{s}\}$ are the nodes j that influence node i and we define the set of neighboring nodes as the union of both as

$$\mathcal{N}_i = \{j | (i, j) \in \mathcal{E}_\mathrm{s} \vee (j, i) \in \mathcal{E}_\mathrm{s}\} = \mathcal{N}_{\mathrm{in},i} \cup \mathcal{N}_{\mathrm{out},i}.$$

The second graph that we define is the computation graph.

Definition 2.4. The *computation graph* $\mathcal{G}_\mathrm{comp} = (\mathcal{V}_\mathrm{comp}, \mathcal{E}_\mathrm{comp})$ is an undirected graph that describes which subsystem communicates with which other subsystem to compute the solution to a common problem.

The mentioned problem in Definition 2.4 may be a system analysis problem, a control design problem, or a consensus problem among other things. The communication along this graph usually takes place offline and it entails the exchange of model data, optimization variables or simulated trajectories.

The third graph is the control graph.

Definition 2.5. The *control graph* $\mathcal{G}_\mathrm{control} = (\mathcal{V}_\mathrm{control}, \mathcal{E}_\mathrm{control})$ determines which subsystem communicates with which other subsystem to compute its input signal.

Subsystems exchange their state measurements along the control graph which naturally takes place online while the system is running. This graph furthermore determines the structure of the control law.

In addition, we also define the undirected version of the system graph.

Definition 2.6. The undirected system graph is given by $\mathcal{G}_{s,u} = (\mathcal{V}_{s,u}, \mathcal{E}_{s,u})$ with $\mathcal{E}_{s,u} = \mathcal{E}_s \cup \mathcal{E}_s^T$, where $\mathcal{E}_s^T$ is the edge set of the transpose graph of $\mathcal{G}_s$.

For all graphs so far, the nodes of the graphs represent the subsystems and the node sets coincide, i.e. $\mathcal{V}_s = \mathcal{V}_{control} = \mathcal{V}_{comp} = \mathcal{V}_{s,u}$.

Using the problem of control design as an example, an illustration of the three different graphs is shown in Figure 2.1. A common approach to control large-scale systems is shown in Figure 2.1a. Here, we have a decentralized control law and a single entity, in this case subsystem one, that has knowledge of the overall system model as every other node shares its model data with subsystem one during the computation phase. After the design, subsystem one passes the individual control laws to the other agents. Equivalently, instead of subsystem one, a superordinate central entity could design the control law and pass it to the subsystems. In the case of a centralized design, the computation graph $\mathcal{G}_{comp}$ contains a star topology where one node is directly connected to every other node. In Figure 2.1b, we have a disconnected control graph $\mathcal{G}_{control}$ indicating a decentralized control law and an incomplete computation graph $\mathcal{G}_{comp}$ not containing a star topology. In this example, subsystem three gets information from subsystem one during the design phase, identically to how subsystem one receives information from subsystem four, in addition to the information that every subsystem has about itself already. In recent years, many distributed control approaches have been introduced, assuming full model knowledge, which is depicted in Figure 2.1c. In this thesis, we consider the case that the graph $\mathcal{G}_{control}$ is incomplete and $\mathcal{G}_{comp}$ does not contain a star topology, as illustrated in Figure 2.1d.

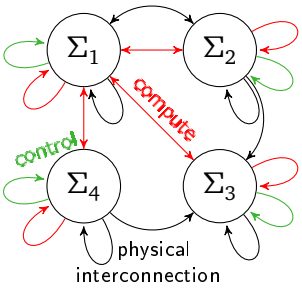
One last graph is introduced, based on the matrix $A^{sym} = (A + A^T)$.

Definition 2.7. The graph $\mathcal{G}_{LMI} = (\mathcal{V}_{LMI}, \mathcal{E}_{LMI})$ describes the individual element structure of the matrix A^{sym}, i.e. $(i, j) \in \mathcal{E}_{LMI}$ if and only if the individual entry $A_{ij}^{sym} \neq 0$.

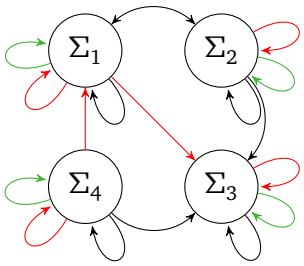
While $\mathcal{G}_{s,u}$ describes the block structure of A^{sym}, $\mathcal{G}_{LMI}$ describes the individual element structure. Therefore, $\mathcal{G}_{s,u}$ is a hypergraph to $\mathcal{G}_{LMI}$ that is obtained by merging the nodes that correspond to the states of one individual subsystem to a compressed subsystem node. Given these graphs, the following definition represents the overall theme of this thesis.

Definition 2.8. A system analysis or control design method uses only *local model information* when the offline information exchange is in accordance with the computation graph $\mathcal{G}_{comp}$ and there is no central entity with global knowledge. The subsystems can know the overall system size N and the total number of states n. If the method requires online information exchange, it has to be in accordance with the control graph $\mathcal{G}_{control}$.

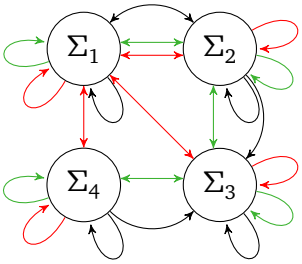
Methods using local model information ensure privacy of the subsystems in the sense that the subsystems need to share their dynamic model (2.8) and other information only with a small subset of other subsystems, and that there is no central entity which knows the overall system. This definition will be stated in a more precise and adapted way suitable to the specific problems in Sections 3.1 and 4.1.



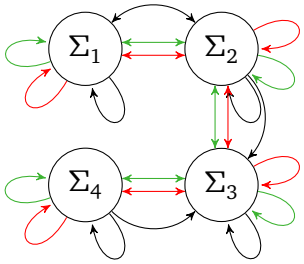
(a) Disconnected $\mathcal{G}_{\text{control}}$, $\mathcal{G}_{\text{comp}}$ has star topology



(b) Disconnected $\mathcal{G}_{\text{control}}$, arbitrary $\mathcal{G}_{\text{comp}}$



(c) 1-hop $\mathcal{G}_{\text{control}}$, $\mathcal{G}_{\text{comp}}$ has star topology



(d) 1-hop $\mathcal{G}_{\text{control}}$, 1-hop $\mathcal{G}_{\text{comp}}$

Figure 2.1: Different graphs: System graph $\mathcal{G}_s$ in black, control graph $\mathcal{G}_{\text{control}}$ in green, computation graph $\mathcal{G}_{\text{comp}}$ in red.

2.5 Distributed constrained Nesterov algorithm

In Chapters 3 and 4, we formulate optimization problems to answer control questions in a distributed fashion, namely stability analysis and control design. One key element of the distributed solution is the application of the distributed constrained Nesterov algorithm (DCNA). This introduction on the algorithm follows [66, 67, 68] and we refer to these references for more details. The DCNA incorporates dual decomposition and also its efficiency is higher by an order of magnitude compared to the standard gradient method. We start with the introduction of the unconstrained Nesterov algorithm [66].

We consider an optimization problem of the form

$$\min_{x \in X} f(x),$$

where $f : X \to \mathbb{R}$ is a convex and continuously differentiable function on a closed and convex set $X \subseteq \mathbb{R}^n$. We assume that the gradient of f is Lipschitz continuous with Lipschitz constant L. We introduce a prox-function $d(x)$ which is a continuous and strongly convex function on X with convexity parameter $\sigma > 0$. Given this, the Nesterov algorithm is stated in Algorithm 1 [66].

Algorithm 1 has a complexity of $O(\sqrt{\frac{L}{\epsilon}})$. The algorithm serves as the basis for the DCNA, but the DCNA is aimed at distributed optimization applications including constraints. For the DCNA, we consider a partially separable convex optimization problem of the following

Algorithm 1 Unconstrained Nesterov algorithm.

For $k \geq 0$ do

1. Compute $\nabla f(x^k)$.
2. Find
$$y^k = \arg\min_{y \in X} \left[\nabla f(x^k)^\mathsf{T} y + \frac{L}{2} ||y - x^k||^2 \right].$$
3. Find
$$z^k = \arg\min_{z \in X} \left[\frac{L}{\sigma} d(z) + \sum_{j=1}^{k} \nabla f(x^j)^\mathsf{T} z \right].$$
4. Set
$$x^{k+1} = \frac{2}{k+3} z^k + \frac{k+1}{k+3} y^k.$$

form

$$\min_{x_i \in X_i, (i=1,\ldots,n)} \sum_{i=1}^{n} f_i(x_i) \tag{2.10a}$$

$$\text{s.t.} \sum_{i=1}^{n} A_{i,\text{eq}} x_i = b_{\text{eq}}, \tag{2.10b}$$

$$\sum_{i=1}^{n} A_{i,\text{ineq}} x_i \leq b_{\text{ineq}}, \tag{2.10c}$$

where $f_i : \mathbb{R}^{m_i} \to \mathbb{R}$ is a convex and continuously differentiable function on a given compact and convex set X_i for all $i = 1, \ldots, n$. To describe the equality and inequality constraints, we use the matrices $A_{i,\text{eq}} \in \mathbb{R}^{m_{\text{eq}} \times m_i}$ and $A_{i,\text{ineq}} \in \mathbb{R}^{m_{\text{ineq}} \times m_i}$ as well as the vectors $b_{i,\text{eq}} \in \mathbb{R}^{m_{\text{eq}}}$ and $b_{i,\text{ineq}} \in \mathbb{R}^{m_{\text{ineq}}}$. Furthermore, we introduce prox-functions $d_i(x_i)$ which are continuous and strongly convex functions on X_i with convexity parameters $\sigma_i > 0$. The Lagrangian of (2.10) is given by

$$\mathcal{L}(x_1, \ldots, x_n, \mu, \lambda) = \sum_{i=1}^{n} f_i(x_i) + \mu^\mathsf{T} \left(\sum_{i=1}^{n} A_{i,\text{eq}} x_i - b_{\text{eq}} \right) + \lambda^\mathsf{T} \left(\sum_{i=1}^{n} A_{i,\text{ineq}} x_i - b_{\text{ineq}} \right).$$

As can be observed, the Lagrangian is separable in the variables x_i with $i = 1, \ldots, n$. The corresponding dual function is then

$$\phi(\mu, \lambda) = \min_{x_i \in X_i, (i=1,\ldots,n)} \mathcal{L}(x_1, \ldots, x_n, \lambda, \mu),$$

which can also be evaluated in parallel. It can be shown that the gradient of the dual function described by

$$\nabla \phi(\mu, \lambda) = \begin{pmatrix} \sum_{i=1}^{n} A_{i,\text{eq}} x_i(\mu, \lambda) - b_{\text{eq}} \\ \sum_{i=1}^{n} A_{i,\text{ineq}} x_i(\mu, \lambda) - b_{\text{ineq}} \end{pmatrix}$$

is Lipschitz continuous with the Lipschitz constant

$$L = \sum_{i=1}^{n} \frac{||A_{i,\text{eq}}||^2 + ||A_{i,\text{ineq}}||^2}{\sigma_i}.$$

If we then apply the Nesterov algorithm described in Algorithm 1 to the dual problem

$$\arg\max_{(\mu,\lambda)} \phi(\mu,\lambda) = \arg\min_{(\mu,\lambda)} -\phi(\mu,\lambda),$$

we arrive at the DCNA in Algorithm 2.

Note that in comparison to the references [66, 67, 68], we have left out certain constant terms in the subproblems to simplify the readability of the algorithm. Also note that the literature [67, 68] does not use the term DCNA. They rather present a proximal center algorithm or (D)PCA. In their case, proximal center algorithms are needed to tackle optimization problems that are non-smooth so that they require smoothing techniques leading to the (D)PCA. The optimization problems considered in this thesis, however, do not require smoothing and, therefore, we do not introduce those techniques. We do, however, make use of the dual decomposition aspect of their approach. To reflect these differences to the literature we refer to the algorithm as DCNA.

Algorithm 2 Distributed constrained Nesterov algorithm.

For $k \geq 0$ do

1. Given λ^k and μ^k compute for $i = 1, \ldots, n$

$$x_i^{k+1} = \arg\min_{x_i \in X_i} \left[f_i(x_i) + {\mu^k}^{\mathsf{T}} A_{i,\text{eq}} x_i + {\lambda^k}^{\mathsf{T}} A_{i,\text{ineq}} x_i \right].$$

2. Compute

$$\nabla\phi(\mu^k,\lambda^k) = \begin{pmatrix} \sum_{i=1}^{n} A_{i,\text{eq}} x_i^{k+1} - b_{\text{eq}} \\ \sum_{i=1}^{n} A_{i,\text{ineq}} x_i^{k+1} - b_{\text{ineq}} \end{pmatrix}.$$

3. Find

$$(\tilde{\mu}^k, \tilde{\lambda}^k) = \arg\max_{(\mu,\lambda)} \left[\nabla\phi(\mu^k,\lambda^k)^{\mathsf{T}} (\mu^{\mathsf{T}},\lambda^{\mathsf{T}})^{\mathsf{T}} - \frac{L}{2} ||(\mu^{\mathsf{T}},\lambda^{\mathsf{T}})^{\mathsf{T}} - ({\mu^k}^{\mathsf{T}},{\lambda^k}^{\mathsf{T}})^{\mathsf{T}}||^2 \right].$$

4. Find

$$(v^k, t^k) = \arg\max_{(v,t)} \left[-\frac{L}{\sigma} d(v,t) + \sum_{j=1}^{k} \nabla\phi(\mu^k,\lambda^k)^{\mathsf{T}} (v^{\mathsf{T}},t^{\mathsf{T}})^{\mathsf{T}} \right].$$

5. Set

$$(\mu^{k+1},\lambda^{k+1}) = \frac{2}{k+3}(v^k,t^k) + \frac{k+1}{k+3}(\tilde{\mu}^k,\tilde{\lambda}^k).$$

3

Distributed stability analysis with local model information

This chapter addresses the first main contribution of this thesis. It presents two different distributed tests for stability for the general class of interconnected LTI systems using local model information. There are no restrictions on the systems themselves and only limited information exchange with very few subsystems is necessary. The first stability test is based on the well-known vector Lyapunov condition and the corresponding M-Matrix test from [15]. Instead of testing whether a matrix is an M-matrix in a centralized fashion, we formulate an optimization problem that contains the stability conditions as constraints. The second stability test is based on the Lyapunov inequality and is analogously formulated as an optimization problem. More specifically, we consider α-block diagonal Lyapunov stability [69, 70] because of its structural properties. Both optimization problems have a structure which can be exploited using the distributed constrained Nesterov algorithm (DCNA) from [67, 68], which is explained in Section 2.5. This yields two different distributed stability tests that allow us to check if an LTI system is asymptotically stable using only local model information in an iterative distributed optimization scheme. We stress that the use of distributed optimization techniques is not mainly aimed at reducing computational effort, but the idea is to obtain methods that require only local model information and preserve model data privacy for the subsystems in that way as much as possible. This is achieved in so far that the subsystems only need to share their dynamic model with their direct neighbors according to the computation graph, and not globally or centrally. Furthermore, we will see that the developed tools can also be used to perform a distributed model reduction.

The remainder of this chapter is organized as follows. Section 3.1 contains the problem statement, followed by an in-depth overview on related work in Section 3.2. Then, we present the reformulation of the stability conditions as optimization problems in Section 3.3.

Their distributed solution is addressed in Section 3.4. These results are illustrated and analyzed numerically in Section 3.5 before we investigate the conservativeness of the stability tests in Section 3.6. Finally, Section 3.7 contains our results on distributed model reduction, and we conclude with a summary in Section 3.8.

3.1 Problem formulation

In this section, we introduce the considered system dynamics for this chapter and specify the exact problem description of this part of the thesis.

For the system analysis with regards to stability, we consider the system dynamics from (2.8),(2.9) without an input, that is the subsystem dynamics are

$$\dot{x}_i(t) = A_{ii}x_i(t) + \sum_{\substack{j=1 \\ j\neq i}}^{N} A_{ij}x_j(t),\ i = 1,...,N, \tag{3.1}$$

where $x_i \in \mathbb{R}^{n_i}$ and $A_{ij} \in \mathbb{R}^{n_i \times n_j}$, and the overall system is

$$\dot{x} = Ax. \tag{3.2}$$

Remark 3.1. The division into the N subsystems and the resulting interconnection graph $\mathscr{G}_{\mathrm{s}}$ is motivated by the desire to preserve model data privacy, and the use of only local model information. Classically, the structuring of the overall system (3.2) into N subsystems (3.1) is motivated by physics, function or geography. In our setting, in contrast, subsystem (3.1) may comprise multiple components when we deal with stability analysis. These can be physical, functional or geographically distant agents which, however, are willing to share their model data completely among each other, i.e. the clustering of the overall system is induced by privacy constraints.

Based on these remarks, we make the following assumption for the rest of this chapter.

Assumption 3.1. The computation graph $\mathscr{G}_{\mathrm{comp}}$ coincides with the undirected system graph $\mathscr{G}_{\mathrm{s,u}}$.

Note that the control graph is irrelevant in this context. We refine Definition 2.8 of local model information for the case of the system analysis.

Definition 3.1. Given system (3.1),(3.2), a system analysis method is considered to use *local model information* if subsystem i has knowledge of A_{ii}, all A_{ij}, A_{ji} with $(i,j) \in \mathscr{E}_{\mathrm{comp}}$, and the overall number of subsystems N. Furthermore, it exchanges information only with the subsystems with $(i,j) \in \mathscr{E}_{\mathrm{comp}}$.

The goal of this chapter is summarized concisely in the following problem.

Problem 1.
Given Assumption 3.1, decide if system (3.1),(3.2) is stable using only local model information in accordance with Definition 3.1.

To illustrate the results of this chapter, we introduce a small-scale academic example system that we use throughout the chapter.

Example 3.1. Consider a system with $N = 4$ agents arranged along a line-graph as previously shown in Figure 2.1d. The system graph $\mathcal{G}_{\text{s}}$, the undirected system graph $\mathcal{G}_{\text{s,u}}$, the computation graph $\mathcal{G}_{\text{comp}}$ and the control graph $\mathcal{G}_{\text{control}}$ all have the same form and we show it in a simplified form in Figure 3.1. In this example, each agent has $n_i = 2$ states, $\forall i$, such that the overall system size is $n = 8$. The state matrix is given by

$$A = \left[\begin{array}{cc|cc|cc|cc} -1.2 & -0.2 & 0.6 & 0.1 & 0 & 0 & 0 & 0 \\ 0.1 & -1.3 & -0.2 & 0.1 & 0 & 0 & 0 & 0 \\ \hline 0.1 & -0.1 & -0.6 & 0.1 & 0.2 & -0.7 & 0 & 0 \\ -0.2 & 0.2 & 0.1 & -1.3 & -0.3 & -0.3 & 0 & 0 \\ \hline 0 & 0 & -0.1 & 0.1 & -1.0 & -0.1 & -0.2 & -0.2 \\ 0 & 0 & 0.4 & 0.2 & 0.1 & -2.0 & -0.1 & -0.2 \\ \hline 0 & 0 & 0 & 0 & -0.1 & -0.5 & -1.7 & -0.1 \\ 0 & 0 & 0 & 0 & -0.3 & -0.3 & -0.1 & -2.0 \end{array}\right]. \tag{3.3}$$

3.2 Related work

In the following, we provide an overview on related results on stability analysis of interconnected dynamical systems and on a few limited results on distributed stability analysis. Large-scale system stability is analyzed in [71] where the authors derive sufficient stability conditions under the assumption that every subsystem is dissipative and that both the supply rate function of each subsystem and the overall interconnection structure are centrally known. The conditions are then posed in the form of LMIs or definiteness requirements. For the special case of passive systems, the condition boils down to the diagonal stability of an associated matrix that contains information about the interconnection structure and the passivity of each subsystem [72]. Both, the vector Lyapunov method explained in Section 2.1 and the method by [71], have in common that the stability test itself is performed centrally, but they exploit the subsystem formulation of the interconnected system. In [73], the two approaches are relaxed and it is pointed out that for the class of positive linear systems, stability can be evaluated distributedly. That means that for this restricted system class no complete central model data is required, but that each subsystem knows a part of the overall system model. In a related fashion, the results in [74] show that a distributed stability test is possible for positive systems using distributed linear programming. Other results on scalable and distributed system analysis are given in [75]. The authors present scalable stability conditions for interconnected heterogeneous LTI systems represented by transfer functions that are feedback coupled through a bipartite graph. The sufficient conditions they derive are based

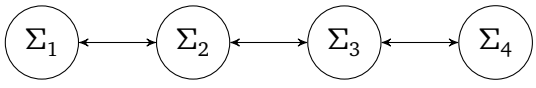


Figure 3.1: Structure of small-scale example system.

on the Nyquist stability criterion, and the conditions are posed in such a way that subsystems only need to exchange information with those subsystems that they are physically connected to. The conditions are generalized in [76] using integral quadratic constraints (IQCs). However, in both cases, the derived conditions need to be evaluated for all frequencies ω and no analysis of the degree of conservativeness of the sufficient condition is given.

The concept of α-block diagonal Lyapunov stability, a generalization of diagonal stability [31], and a notion that is adopted in this thesis, is the subject of [69] and [70], though not from a perspective where the model knowledge is distributed among subsystems. A related paper to our approach of decomposing Lyapunov inequalities is [77]. Their work is based on the same LMI decomposition technique, but their focus is not the solution using local model information but rather solving the problem more efficiently while maintaining a centralized solution.

With this overview in hand, it is clear that only very few results on distributed system analysis with limited model information are available so far and most results take a centralized perspective or make no statement on the degree of their conservativeness. Furthermore, the distributed system analysis so far is only achieved for restricted subclasses of linear systems. This restriction is reduced in this thesis in that we can treat general LTI systems, and we can analyze them with local model information, distributed among the subsystems.

3.3 Sufficient stability conditions with a sparsity structure

In this section, we lay the foundation for the two distributed stability tests developed in this chapter. We start with two different stability conditions that are reformulated into the form of optimization problems. The optimization problems have a sparsity structure which allows the distributed solution of those problems in the following sections. The principle idea is that classical stability conditions, vector Lyapunov functions and the Lyapunov inequality, are used as a constraint in an optimization problem. For the Lyapunov inequality, a structural constraint is imposed on the solution variable to maintain the desired sparsity structure.

3.3.1 Vector Lyapunov function

In this subsection, we present a test to analyze whether a system is connectively stable based on the solution of an optimization problem. The notion of connective stability and M-matrices was already briefly introduced in Section 2.1.

As we have seen there, Theorem 2.1 allows us to construct a test matrix to formulate a condition for connective stability. With Assumption 3.1, the construction of this test matrix requires only local model information because the subsystems only require their own row of the dynamics matrix. It can be seen from (2.4) that the sparsity structure of W is identical to the block sparsity structure of A as described by the system graph $\mathcal{G}_s$. However, the test if the matrix W is an M-matrix, i.e. if there exists a $d > 0$ such that $Wd > 0$, is a global problem. Hence, to enable the distributed stability test presented in the following section, we reformulate the condition of Theorem 2.1 as an optimization problem which then allows the application of distributed optimization techniques.

As a first step, we reformulate the stability condition as the following optimization problem with convexity parameters $\sigma_\delta \in \mathbb{R}_{++}, \sigma_{d_i} \in \mathbb{R}_{++}$

$$\min_{(d,\delta)\in\mathbb{R}^{N+1}} f(d,\delta) = -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{i=1}^{N} \frac{\sigma_{d_i}}{2N} d_i^2 \tag{3.4a}$$

$$\text{s.t.} - Wd + \gamma\delta 1_N \leq 0, \tag{3.4b}$$

$$-d + \delta 1_N \leq 0, \tag{3.4c}$$

where $\gamma \in \mathbb{R}_{++}$ is arbitrary. In this form, the function $f(d,\delta)$ in (3.4a) is strongly convex. The optimal function value of the problem is denoted by f^*. The choice for the convexity parameters $\sigma_\delta, \sigma_{d_i}$ has an influence on the solution, and for further insight on how to choose them please see Remark 3.3 below.

Problem 3.4 is inspired by phase-1 problems from linear programming that are solved to find a feasible solution to a given set of linear constraints [78]. This technique reformulates a problem that may have infeasible points into a problem that is always feasible. Here, the strongly convex part $-\delta + \frac{\sigma_\delta}{2}\delta^2$ in (3.4a) alone guarantees that a vector $d^* > 0$ which satisfies $Wd^* > 0$ is the candidate for an optimal solution of (3.4), as in this case there exists a $\delta^* > 0$ with $-\delta^* + \frac{\sigma_\delta^*}{2}\delta^{*2} < 0$. Finally, we add the term $\sum_{i=1}^{N} \frac{\sigma_{d_i}}{2N} d_i^2$ in (3.4a) to obtain a strongly convex objective function $f(d,\delta)$ which in turn yields a differentiable dual objective function. In the following lemma, we show that even with the addition of this strongly convex term in d, the optimal objective function value of (3.4) indicates the existence of a vector $d > 0$ such that $Wd > 0$. The conversion from a feasibility problem to an equivalent optimization problem allows us to apply distributed optimization algorithms later.

Lemma 3.1. *There exists a $d > 0$ such that $Wd > 0$ if and only if there exists a feasible point (d,δ) for (3.4) with a negative optimal objective function value $f^* < 0$.*

Proof. $\Rightarrow$: Let $Wd > 0$ hold for some $d = \tilde{d} > 0$. Set $\lambda_1 = \min_i(\tilde{d}_i)$, and $\lambda_2 = \min_i((W\tilde{d})_i)$. Then, $\tilde{\delta} := \min(\lambda_1, \lambda_2/\gamma) > 0$. Furthermore, $(d,\delta) = (0,0)$ and $(d,\delta) = (\tilde{d},\tilde{\delta})$ are feasible for problem (3.4). Hence, by convexity, for all $t \in [0,1]$, $(d,\delta) = (t\tilde{d}, t\tilde{\delta})$ is feasible for (3.4). Now consider the objective function $\Phi(t)$ at $(t\tilde{d}, t\tilde{\delta})$

$$\Phi(t) := -t\tilde{\delta} + \frac{\sigma_\delta}{2} t^2\tilde{\delta}^2 + \sum_{i=1}^{N} \frac{\sigma_{d_i}}{2N} t^2\tilde{d}_i^2.$$

Since $\Phi'(0) = -\tilde{\delta} < 0$, we see that if we choose $0 < t \leq 1$ sufficiently small, then $\Phi(t) < \Phi(0) = 0$. We then set $(d,\delta) = (t\tilde{d}, t\tilde{\delta})$ which has the desired properties.
$\Leftarrow$: We now show that if there exists a feasible point $(\tilde{d},\tilde{\delta})$ of (3.4) with negative objective function value, then $\tilde{\delta} > 0$ and $W\tilde{d} > 0$. In fact, a negative objective function value can only be achieved if $\tilde{\delta} > 0$. Now

$$\tilde{d} \geq \tilde{\delta} 1_N > 0,\ W\tilde{d} \geq \gamma\tilde{\delta} 1_N > 0.$$

Hence, setting $d = \tilde{d}$ finishes the proof. □

The consequence of Lemma 3.1 is that the optimization problem 3.4 can be regarded as a stability condition, which is summarized in the following theorem.

Theorem 3.1. *Given Assumption 2.1, the system (3.1),(3.2) is connectively stable if the optimization problem 3.4 has a negative optimal objective value, i.e.* $f^* < 0$.

Proof. The proof is straightforward and follows from Theorem (2.1) and Lemma (3.1). □

This theorem is the basis for the distributed stability test that follows in Section 3.4.1. Notice that it enables us to decide if a system is stable based purely on the sign of the optimal function value.

Example 3.2. We can apply Theorem 3.1 to the system from Example 3.1. First, we need to construct the matrix W and obtain

$$W = \begin{bmatrix} 1.28 & -1.14 & 0 & 0 \\ -0.16 & 0.59 & -0.75 & 0 \\ 0 & -0.47 & 1.01 & -0.33 \\ 0 & 0 & -0.68 & 1.67 \end{bmatrix}.$$

Notice that each multi-dimensional subsystem is summarized by one numerical value. The optimal function value with $\sigma_{d_l} = 10^{-3}, \sigma_\delta = 10^{-3}$ is computed to be $f^* = -0.0858 < 0$. Therefore, we can conclude connective stability of the system. Indeed, the corresponding vector $d = [8.02, 8.84, 4.97, 2.13]^\mathsf{T}$ leads to a positive product $Wd > 0$, which means that the original stability condition is satisfied.

3.3.2 α-block diagonal Lyapunov stability

In this subsection, we present an alternative method for testing stability of large-scale systems based on the Lyapunov linear matrix inequality (LMI). In Section 2.1, we already mentioned Lyapunov stability analysis for LTI systems and the special case of diagonal Lyapunov stability. A generalization of diagonal stability is α-block diagonal Lyapunov stability which we define as follows.

Definition 3.2. [69, 70] Given is a partition $\alpha = (\alpha_1, \ldots, \alpha_k)$ of the state indices $\{1, \ldots, n\}$ where each partition set α_i has the cardinality n_{α_i} and $\sum_{i=1}^{k} n_{\alpha_i} = n$. A system is called *$\alpha$-block-diagonally Lyapunov stable* if there exists a block matrix $P \in \mathbb{S}_{++}$ with the given block partition α such that $P = \mathrm{diag}(P^1, \ldots, P^k)$ with $P^i \in \mathbb{S}_{++}^{n_{\alpha_i}}$, $i = 1, \ldots, k$, and such that

$$A^\mathsf{T} P + PA \prec 0. \tag{3.5}$$

To illustrate the block partition, consider the following example.

Example 3.3. Given the block partition $\alpha = (\alpha_1, \alpha_2, \alpha_3)$ with $\alpha_1 = \{1\}, \alpha_2 = \{2, 3\}, \alpha_3 = \{4, 5, 6\}$ and $n_{\alpha_1} = 1, n_{\alpha_2} = 2, n_{\alpha_3} = 3$, a system with a dynamics matrix $A \in \mathbb{R}^{6\times 6}$ is α-block-diagonally Lyapunov stable if there is a matrix P with the following structure

$$P = \begin{bmatrix} P_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & P_{22} & P_{23} & 0 & 0 & 0 \\ 0 & P_{23} & P_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & P_{44} & P_{45} & P_{46} \\ 0 & 0 & 0 & P_{45} & P_{55} & P_{56} \\ 0 & 0 & 0 & P_{46} & P_{56} & P_{66} \end{bmatrix},$$

such that (3.5) is satisfied.

The notion of α-block diagonal Lyapunov stability is the basis for the second distributed stability test in the following section. If we do not have the block diagonal structural restriction on the matrix P, then the Lyapunov inequality $A^\mathsf{T}P + PA \prec 0$ has no sparsity structure, and no distributed approaches are applicable.

Assumption 3.1 has the following consequence for this section: The partition α coincides with the indices of the individual subsystems and it follows that the block sizes n_{α_i} of the blocks P^i, $i = 1, \ldots, N$, are determined by the subsystem sizes n_i in (3.1).

Example 3.4. Applying the last comment to the example system from Example 3.1, the matrix P has four blocks, each with size two. Thus, $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ with $\alpha_1 = \{1,2\}, \alpha_2 = \{3,4\}, \alpha_3 = \{5,6\}, \alpha_4 = \{7,8\}$ and $n_{\alpha_i} = 2\,\forall i$.

The overall goal of this section is to find a solution to the Lyapunov inequality with block diagonal P, i.e. to find a solution to the LMI (3.5). Therefore, in the following we consider a problem with a sparsity structure induced by the term $(A^\mathsf{T}\mathrm{diag}(P^1, \ldots, P^N) + \mathrm{diag}(P^1, \ldots, P^N)A)$ for any $P^i \in \mathbb{S}^{n_i}$. Given Assumption 3.1, the sparsity structure of this term is identical to the sparsity structure of $A^{\mathrm{sym}} = (A + A^\mathsf{T})$. As stated in Section 2.4.2, it follows that the block sparsity structure of the problem is also described by the undirected system graph $\mathcal{G}_{\mathrm{s,u}}$, while the element sparsity structure is described by the graph $\mathcal{G}_{\mathrm{LMI}} = (\mathcal{V}_{\mathrm{LMI}}, \mathcal{E}_{\mathrm{LMI}})$.

In order to find a solution to the LMI (3.5), we use the same idea as in Section 3.3.1 and formulate an optimization problem. We start with the following problem with convexity parameters $\sigma_\delta \in \mathbb{R}_{++}$ and $\sigma_{P^l} \in \mathbb{R}_{++}$,

$$\min_{\delta \in \mathbb{R}, P^l \in \mathbb{S}^{n_l}} f(\delta, P^l) = -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2N} ||P^l||_F^2 \tag{3.6a}$$

$$\text{s.t. } F(P, \delta) \succeq 0, \tag{3.6b}$$

$$P^l - \delta I_{n_l} \succeq 0 \text{ for } l = 1, \ldots, N, \tag{3.6c}$$

where

$$F(P, \delta) := -A^\mathsf{T}\mathrm{diag}(P^1, \ldots, P^N) - \mathrm{diag}(P^1, \ldots, P^N)A - \gamma\delta I_n,$$

and $\gamma \in \mathbb{R}_{++}$ is arbitrary. Again, the variable δ is used to make the problem feasible, independent of the stability of the system. The optimal function value of problem (3.6) is denoted by f^*.

Lemma 3.2. *The inequality (3.5) holds for some $P = \tilde{P} \in \mathbb{S}^n$ with $\tilde{P} = \mathrm{diag}(\tilde{P}^1, \ldots, \tilde{P}^N)$ if and only if there exists a feasible point $(P^1, \ldots, P^N, \delta)$ for (3.6) with $f^* < 0$.*

Proof. The proof is identical to the proof of Lemma 3.1 because the overall problem formulation is identical and the objective function of (3.6) is also convex. Additionally, as in the proof of Lemma 3.1, $f^* < 0$ implies that $\delta^* > 0$. □

Similarly to the vector Lyapunov test, the optimization problem (3.6) can be considered to be a stability condition based on Lemma 3.2 and we state this in the following theorem.

Theorem 3.2. *The system (3.1),(3.2) is α-block-diagonally Lyapunov stable if the optimization problem (3.6) has a negative optimal objective value, i.e.* $f^* < 0$.

Proof. The proof is straightforward and follows from Lemma (3.2) and standard Lyapunov stability theory. □

Example 3.5. When we apply Theorem 3.2 to the system dynamics from Example 3.1 with $\sigma_{P^l} = 10^{-3}, \sigma_\delta = 10^{-3}$, we obtain the optimal function value $f^* = -49.4249 < 0$, so the system is α-block-diagonally Lyapunov stable. The obtained corresponding Lyapunov matrix is given by

$$P = \text{diag}\left(\begin{bmatrix} 98.85 & 0 \\ 0 & 98.95 \end{bmatrix}, \begin{bmatrix} 131.71 & 3.61 \\ 3.61 & 99.25 \end{bmatrix}, \begin{bmatrix} 98.85 & 0 \\ 0 & 98.95 \end{bmatrix}, \begin{bmatrix} 98.85 & 0 \\ 0 & 98.95 \end{bmatrix}\right),$$

which is positive definite, and the matrix $A^T P + PA$ is negative definite.

Remark 3.2. It is clear that the approach of using an optimization based approach to solve Lyapunov inequalities, and especially doing so in a distributed fashion, is not restricted to stability analysis. In fact, we will see later on in this thesis that it can be used for model reduction and to guarantee stability of the closed loop in a control design setting. Further investigations of the applicability are considered for future work.

3.3.3 Summary

In this section, we reformulated classical stability conditions into optimization problems. Both optimization problems exhibit a sparsity structure that concurs with the interconnection structure of the original system. For the first condition, the vector Lyapunov condition, the structure stems from the original stability condition. For the second condition, the Lyapunov inequality, the structure needs to be imposed, which introduces conservativeness and we lose the necessity of the original condition. The sparsity structure will be useful in the following sections to achieve stability conditions that require only local model information.

3.4 Distributed optimization for stability analysis with local model information

This section continues on the path from the previous one with the goal of developing distributed stability tests. In the previous section, we have reformulated stability conditions into optimization problems exhibiting a sparsity structure. In this section, we build upon this basis and use distributed optimization techniques to check stability of interconnected LTI systems using local model information.

3.4.1 Vector Lyapunov function

In Section 3.3.1, an optimization problem has been derived which serves as a condition for connective stability, see Theorem 3.1. In this subsection, we employ distributed optimization techniques to solve this optimization problem such that only local model information

according to Definition 3.1 is required. The presented distributed optimization approach is based on the results in [67, 68]. The information exchange topology of the employed algorithm is determined by the structure of the constraints, which corresponds to the graph $\mathcal{G}_{\text{s,u}}$. Given Assumption 3.1, the structure also coincides with the computation graph $\mathcal{G}_{\text{comp}}$, as desired by our goal formulated in Problem 1. In the following, we describe how the optimization problem can be solved distributedly using the DCNA from [67, 68] and Section 2.5. We solve the dual problem instead of the original one because it enables the distribution of the computation. In order to derive the dual problem of (3.4), consider the corresponding Lagrangian

$$\begin{aligned}\mathcal{L}(d,\delta,\lambda,\mu) &= -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{i=1}^{N}\frac{\sigma_{d_i}}{2N}d_i^2 + \lambda^\mathsf{T}(-Wd+\gamma\delta 1_N) + \mu^\mathsf{T}(-d+\delta 1_N)\\ &= (-1+\sum_{i=1}^{N}(\gamma\lambda_i+\mu_i))\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{i=1}^{N}\left(-\sum_{j=1}^{N}\lambda_j W_{ji} - \mu_i\right)d_i + \frac{\sigma_{d_i}}{2N}d_i^2.\end{aligned}$$

The Lagrangian is clearly separable in δ and in d_i. The corresponding dual function is given by

$$\begin{aligned}\phi(\lambda,\mu) = &\min_{\delta\in\mathbb{R}}\left\{(-1+\sum_{i=1}^{N}(\gamma\lambda_i+\mu_i))\delta + \frac{\sigma_\delta}{2}\delta^2\right\}\\ &+\sum_{i=1}^{N}\min_{d_i\in\mathbb{R}}\left\{\left(-\sum_{j=1}^{N}\lambda_j W_{ji}-\mu_i\right)d_i + \frac{\sigma_{d_i}}{2N}d_i^2\right\},\end{aligned} \tag{3.7}$$

which can be evaluated in parallel.

Due to the uniqueness of the minimizer $d_i(\lambda,\mu_i)$ and $\delta(\lambda,\mu)$ in (3.7), it follows that the gradient of the concave dual function is given by [67, Theorem 3.1]

$$\begin{aligned}\nabla_{\lambda_i}\phi(\lambda,\mu) &= \nabla_{\lambda_i}\mathcal{L}(d(\lambda,\mu),\delta(\lambda,\mu),\lambda,\mu) = \gamma\delta(\lambda,\mu) - \sum_{j=1}^{N}W_{ij}d_j(\lambda,\mu_j),\\ \nabla_{\mu_i}\phi(\lambda,\mu) &= \nabla_{\mu_i}\mathcal{L}(d(\lambda,\mu),\delta(\lambda,\mu),\lambda,\mu) = \delta(\lambda,\mu) - d_i(\lambda,\mu_i).\end{aligned}$$

In addition, it can be shown that the gradient of the dual function is Lipschitz continuous with Lipschitz constant [67]

$$L = \sum_{i=1}^{N}\frac{N(\sum_{j=1}^{N}W_{ji}^2+1)}{\sigma_{d_i}} + \frac{N(1+\gamma^2)}{\sigma_\delta}. \tag{3.8}$$

Finally, the DCNA can be applied to maximize the dual function in parallel to obtain Algorithm 3. It should be mentioned that the DCNA can be implemented with event-based communication to reduce the communication effort as detailed in [68], but this aspect is outside of the scope and not a contribution of this thesis.

Starting with $(\lambda_0,\mu_0) = (0,0)$, the following convergence result for Algorithm 3 holds.

Algorithm 3 Distributed solution of optimization problem (3.4).

For $k \geq 0$ do in parallel:

1. Given λ^k and μ^k compute

$$\delta^{k+1} = \arg\min_{\delta \in \mathbb{R}} \left\{ (-1 + \sum_{i=1}^{N} (\gamma \lambda_i^k + \mu_i^k))\delta + \frac{\sigma_\delta}{2}\delta^2 \right\},$$
$$d_i^{k+1} = \arg\min_{d_i \in \mathbb{R}} \left\{ -\left(\sum_{j=1}^{N} \lambda_j^k W_{ji} + \mu_i^k \right) d_i + \frac{\sigma_{d_i}}{2N} d_i^2 \right\}.$$

2. Compute

$$\nabla_{\lambda_i} \phi(\lambda^k, \mu^k) = \gamma \delta^{k+1} - \sum_{j=1}^{N} W_{ij} d_j^{k+1},$$
$$\nabla_{\mu_i} \phi(\lambda^k, \mu^k) = -d_i^{k+1} + \delta^{k+1}.$$

3. Find

$$\tilde{\lambda}_i^k = \arg\max_{\lambda \geq 0} \left\{ \nabla_{\lambda_i} \phi(\lambda^k, \mu^k)\lambda - \frac{L}{2}(\lambda - \lambda_i^k)^2 \right\},$$
$$\tilde{\mu}_i^k = \arg\max_{\mu \geq 0} \left\{ \nabla_{\mu_i} \phi(\lambda^k, \mu^k)\mu - \frac{L}{2}(\mu - \mu_i^k)^2 \right\}.$$

4. Find

$$t_i^k = \arg\max_{t \geq 0} \left\{ -\frac{L}{2}t^2 + \left(\sum_{j=0}^{k} \frac{j+1}{2} \nabla_{\lambda_i} \phi(\lambda^j, \mu^j) \right) t \right\},$$
$$v_i^k = \arg\max_{v \geq 0} \left\{ -\frac{L}{2}v^2 + \left(\sum_{j=0}^{k} \frac{j+1}{2} \nabla_{\mu_i} \phi(\lambda^j, \mu^j) \right) v \right\}.$$

5. Set

$$\lambda_i^{k+1} = \frac{k+1}{k+3}\tilde{\lambda}_i^k + \frac{2}{k+3} t_i^k,$$
$$\mu_i^{k+1} = \frac{k+1}{k+3}\tilde{\mu}_i^k + \frac{2}{k+3} v_i^k.$$

If $k = k_{\max}$ from Theorem 3.3, stop. Otherwise increase k and return to Step 1.

Theorem 3.3. *Taking $k_{\max} = \lceil \sqrt{8L/\epsilon} \rceil - 1$ with $\epsilon > 0$ and Lipschitz constant L defined by (3.8), then after iteration $k_{\max}$ of Algorithm 3 an approximate solution to problem (3.4) is*

$$(\hat{d}, \hat{\delta}) := \sum_{j=0}^{k_{\max}} \frac{2(j+1)}{(k_{\max}+1)(k_{\max}+2)} (d^{j+1}, \delta^{j+1}),$$

which satisfies the following bounds on the primal gap

$$-\epsilon || \begin{pmatrix} \lambda^* \\ \mu^* \end{pmatrix} ||^2 \leq -\hat{\delta} + \frac{\sigma_\delta}{2} \hat{\delta}^2 + \sum_{i=1}^{N} \frac{\sigma_{d_i}}{2N} \hat{d}_i^{\,2} - f^* \leq 0, \tag{3.9}$$

as well as the following bound on the constraint violation

$$|| \begin{bmatrix} -W\hat{d} + \gamma\hat{\delta} 1_N \\ -\hat{d} + \hat{\delta} 1_N \end{bmatrix}^+ || \leq \epsilon || \begin{pmatrix} \lambda^* \\ \mu^* \end{pmatrix} ||, \tag{3.10}$$

where f^* is the optimal function value of problem (3.4) and $[\ ]^+$ is the componentwise projection onto the nonnegative real numbers. Moreover, λ^* and μ^* are optimal dual multipliers, i.e. they maximize the concave dual function $\phi(\lambda, \mu)$ in (3.7).

Proof. For ease of notation, we define

$$f(d, \delta) = -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{i=1}^{N} \frac{\sigma_{d_i}}{2} d_i^2, \quad W(d, \delta) = \begin{pmatrix} -Wd + \gamma\delta 1_N \\ -d + \delta 1_N \end{pmatrix}, \quad \Lambda = (\lambda^\mathsf{T}, \mu^\mathsf{T})^\mathsf{T}.$$

It can be shown that the following inequality holds [67, Lemma 3.3,Remark 3.8]

$$-||\Lambda^*|| ||[W(\hat{d}, \hat{\delta})]^+|| \leq f(\hat{d}, \hat{\delta}) - f^* \leq f(\hat{d}, \hat{\delta}) - \phi(\hat{\lambda}, \hat{\mu}), \tag{3.11}$$

where $(\hat{\lambda}, \hat{\mu}) := (\tilde{\lambda}^k, \tilde{\mu}^k)$. Moreover, applying Theorem 3.4 in [67] we have

$$f(\hat{d}, \hat{\delta}) - \phi(\hat{\lambda}, \hat{\mu}) \leq \min_{\Lambda \in \mathbb{R}_+^{2N}} \left\{ \frac{2L}{(k_{\max}+1)^2} ||\Lambda||^2 - ||W(\hat{d}, \hat{\delta}), \Lambda|| \right\}. \tag{3.12}$$

It is straightforward to show that the optimal solution of the right-hand side of (3.12) is obtained at $\Lambda = (k_{\max}+1)^2/(4L)[W(\hat{d}, \hat{\delta})]^+$ and it follows that

$$f(\hat{d}, \hat{\delta}) - \phi(\hat{\lambda}, \hat{\mu}) \leq -\frac{(k_{\max}+1)^2}{8L} ||[W(\hat{d}, \hat{\delta})]^+||^2 \leq 0. \tag{3.13}$$

Combining (3.11) and (3.13) yields

$$\frac{(k+1)^2}{8L} \left\| \left[W(\hat{d}, \hat{\delta}) \right]^+ \right\|^2 - \|\Lambda^*\| \left\| \left[W(\hat{d}, \hat{\delta}) \right]^+ \right\| \leq 0$$

and inequality (3.10) follows immediately as well as (3.9). □

A similar proof for this convergence rate is given in [79].

As the norm of the optimal dual multipliers λ^* and μ^* is not known beforehand, the lower bound on the primal gap in (3.9) and the upper bound on the constraint violation in (3.10) cannot be evaluated directly with the results from the distributed algorithm in order to decide if a system is connectively stable. Therefore, we propose the use of Algorithm 4.

Finally, given Algorithms 3 and 4, we can state the following theorem.

Algorithm 4 Iterative adjustment of accuracy parameter ϵ.

1. Choose a minimum accuracy parameter $\epsilon_{\min}$, an initial accuracy parameter ϵ_0 and run Algorithm 3. Set $\epsilon_1 = \theta\epsilon_0$, where $\theta \in (0,1)$.
2. Rerun Algorithm 3.
3. If $\epsilon_i > \epsilon_{\min}$, set $\epsilon_{i+1} = \theta\epsilon_i$ and go back to step 2. Otherwise stop.

The subsystems then need to evaluate if the obtained series of objective function values of all steps converge to a negative value, or if a convergence to 0 occurs.

Theorem 3.4. *Given Assumptions 2.1 and 3.1, and the parameters $\epsilon_{\min}$ and ϵ_0, the subsystems (3.1) can make a distributed decision on the connective stability of system (3.2) using Algorithms 3 and 4 using local model information according to Definition 3.1.*

Proof. Algorithm 3 can be set up to be completely distributed with respect to the computation graph $\mathcal{G}_{\text{comp}}$ which in this case corresponds to the undirected system graph $\mathcal{G}_{\text{s,u}}$. Thus, only neighboring subsystems need to communicate during the optimization to exchange their variables. This follows immediately from the sparsity structure of W for the computation of d_i^{k+1} in Step 2 of Algorithm 3. The computation of δ^{k+1} in Step 1 of Algorithm 3, which is a globally shared variable, can also be performed with local communication by using a consensus algorithm [80]: Each agent computes the term $\gamma\lambda_i + \mu_i$, and the average of all terms is determined with only local communication in a consensus phase. Knowing the size of the network N, as allowed by Definition 2.8, each agent obtains the sum over all individual terms, and it can compute δ^{k+1} by itself. In the same way, the approximate objective function value $f(\hat{d}(\epsilon), \hat{\delta}(\epsilon))$ and the Lipschitz constant L defined in (3.8) can be computed with only local communication.

With Algorithm 4, the subsystems decide on connective stability if $f(\hat{d}(\epsilon), \hat{\delta}(\epsilon)) \not\to 0$. In this case, the subsystems observe a consistent convergence to an optimal value for decreasing values of ϵ. If $f(\hat{d}(\epsilon), \hat{\delta}(\epsilon)) \to 0$ for $\epsilon \to 0$, i.e. no convergence to a value other than 0 is observed, no decision on connective stability can be made because of the known sufficiency of the condition. □

Remark 3.3. Regarding the choice of the convexity parameters σ_{d_i} and σ_δ in the cost function, we observe that Algorithm 3 has a complexity of $O(\sqrt{\frac{L}{\epsilon}})$ according to Theorem 3.3. It follows, on the one hand, that large convexity parameters σ_{d_i} and σ_δ yield a small Lipschitz constant L defined by (3.8), which reduces the number of iterations. On the other hand, in this case the accuracy ϵ has to be chosen smaller, which raises the number of iterations, as the objective function value (3.4a) and, therefore, the primal gap moves closer to zero. It follows that there is a trade-off in the choice, but that the choice has little effect on the overall computational effort.

Remark 3.4. In parallel to the work presented in this thesis, a new distributed dual gradient algorithm for linearly constrained separable problems with strongly convex objective function is published in [81] where a linear convergence rate is shown. This is a good alternative to the algorithm that we use. However, our focus is on the presentation of the feasibility of a

distributed stability test with local model information, and not primarily on the numerical algorithm. Therefore, using the DCNA is just as valid as the algorithm from [81]. Furthermore, the DCNA facilitates an event-triggered communication [68].

Concluding this section, we now have a method to test connective stability of system (3.2) using local model information according to Definition 3.1 based on Theorem 3.4.

3.4.2 α-block diagonal Lyapunov inequality

Similarly to the previous subsection, we employ a distributed optimization approach to distributedly decide on α-block diagonal Lyapunov stability according to Theorem 3.2. In order to realize this, we need to solve optimization problem (3.6), which is achieved with a decomposition method for LMIs such that we can apply distributed optimization methods. However, in order for the decomposition method to be applicable, we have to make the following assumption [82].

Assumption 3.2. $\mathcal{G}_{\text{LMI}}$ is a chordal graph.

A chordal graph is defined to be a graph where every cycle of length ≥ 4 has a chord, i.e. an edge joining non-consecutive vertices of the cycle [83]. Examples of chordal graphs are line graphs, star graphs and trees. Chordal graphs occur in practice, e.g. in vehicle platoons. Distribution systems such as power systems are typically not chordal because of their mesh-like structure. As a reminder, the graph $\mathcal{G}_{\text{LMI}}$ describes the element structure of the matrix $A^{\text{sym}} = A + A^\mathsf{T}$. As we have previously noted in relevance to Assumption 3.1, this structure coincides with the element structure of the LMI $A^\mathsf{T}P + PA \preceq 0$. If $\mathcal{G}_{\text{LMI}}$ does not satisfy this assumption, it is possible to chordalize $\mathcal{G}_{\text{LMI}}$ in polynomial time [82]. This effectively means that we need to allow additional blocks in the matrix P, which leads to additionally required connections in the computation graph $\mathcal{G}_{\text{comp}}$. Consequently, the subsystems cannot decide on α-block diagonal Lyapunov stability using local model information according to Definition 3.1, unless they decide to cooperate with additional subsystems.

Remark 3.5. The authors of [77] investigate system structures that result in a Lyapunov inequality with a graphical representation that is chordal. However, they make no connection between the chordal structure of the Lyapunov inequality and the original structure. They are merely concerned with the possibility of *any* decomposition, not a decomposition among subsystems.

To decompose the LMI (3.6b), we apply the *range-space conversion method* [82]. The idea of this decomposition method is to introduce additional slack variables that ensure that overlapping elements of the decomposed LMI are equal. This allows the application of the DNCA, which was already employed in the previous subsection.

To achieve the decomposition, we need to introduce some notations. First, let $\mathcal{N} = \{1, \dots, n\}$ and consider the *r-space sparsity pattern* [82] of constraint (3.6b), which is

$$\text{SP}_{\text{LMI}} = \{(i,j) \in \mathcal{N} \times \mathcal{N} \ : \ F_{ij}(P,\delta) \neq 0 \text{ for some } (P^1, \dots, P^N, \delta) \in \mathbb{S}^{n_1} \times \dots \times \mathbb{S}^{n_N} \times \ \mathbb{R}, \ i \neq j\}.$$

We use the following definitions [82]:

Let $F \subseteq \mathcal{N} \times \mathcal{N}$ and define

$$\begin{aligned} F^{\bullet} &= F \cup \{(i,i) \ : \ i \in \mathcal{N}\}, \\ \mathbb{S}^n(F,0) &= \left\{X \in \mathbb{S}^n \ : \ X_{ij} = 0 \ \text{ if } \ (i,j) \notin F^{\bullet}\right\}, \\ \mathbb{S}^C &= \left\{X \in \mathbb{S}^n \ : \ X_{ij} = 0 \text{ if } (i,j) \notin C \times C\right\} \ \forall \ C \subseteq \mathcal{N}, \\ \mathbb{S}^C_+ &= \left\{X \in \mathbb{S}^C \ : \ X \succeq 0\right\} \ \forall \ C \subseteq \mathcal{N}, \\ J(C) &= \{(i,j) \in C \times C \ : \ i \leq j\} \ \forall \ C \subseteq \mathcal{N}. \end{aligned}$$

One has $F(P,\delta) \subseteq \mathbb{S}^n(\mathrm{SP}_{\mathrm{LMI}},0)$ for all $(P^1,\ldots,P^N,\delta)$ and the sparsity structure of $\mathbb{S}^n(\mathrm{SP}_{\mathrm{LMI}},0)$ coincides with that of the adjacency matrix of $\mathcal{G}_{\mathrm{LMI}}$.

Consider the maximal cliques $C_1,\ldots,C_p$ of $\mathcal{G}_{\mathrm{LMI}}$ and denote by E_{ij} the $n \times n$ symmetric matrix whose components (i,j) and (j,i) are 1 and all others are 0. Thus, the set $\left\{E_{ij} \ : \ (i,j) \in \mathcal{N} \times \mathcal{N}, \ i \leq j\right\}$ is a basis of $\mathbb{S}^n$. Defining the sets [82]

$$J = \bigcup_{s=1}^{p} J(C_s) \text{ and } \Gamma(i,j) = \{s : \ i \in C_s, \ j \in C_s\} \ \ \forall (i,j) \in J,$$

the LMI (3.6b) is equivalent to

$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} W^s - E_{ij} \bullet F(P,\delta) = 0 \tag{3.14}$$

for $(i,j) \in J$ and $W^s \in \mathbb{S}^{C_s}_+$ for $s = 1,\ldots,p$. For details, we refer to Section 5.2 in [82]. It follows that problem (3.6) can be written as

$$\min_{\delta \in \mathbb{R}, P^l \in \mathbb{S}^{n_l}} -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 \tag{3.15a}$$

$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} W^s - E_{ij} \bullet F(P,\delta) = 0 \text{ for } (i,j) \in J, \tag{3.15b}$$

$$W^s \in \mathbb{S}^{C_s}_+ \text{ for } s = 1,\ldots,p, \tag{3.15c}$$

$$\delta I_{n_l} - P^l \preceq 0 \text{ for } l = 1,\ldots,N. \tag{3.15d}$$

To obtain a strongly convex objective function for (3.15) which guarantees the differentiability of the gradient of the corresponding dual objective function, we modify (3.15) to

$$\min_{\delta \in \mathbb{R}, P^l \in \mathbb{S}^{n_l}} -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 + \sum_{s=1}^{p} \frac{\sigma_{W^s}}{2p} \left\|W^s\right\|_F^2 \tag{3.16a}$$

$$E_{ij} \bullet \sum_{s \in \Gamma(i,j)} W^s - E_{ij} \bullet F(P,\delta) = 0 \text{ for } (i,j) \in J, \tag{3.16b}$$

$$W^s \in \mathbb{S}^{C_s}_+ \text{ for } s = 1,\ldots,p, \tag{3.16c}$$

$$\delta I_{n_l} - P^l \preceq 0 \text{ for } l = 1,\ldots,N. \tag{3.16d}$$

Lemma 3.3. *The inequality (3.5) holds for some $P = \tilde{P} \in \mathbb{S}^n$ with $\tilde{P} = \mathrm{diag}(\tilde{P}^1,\ldots,\tilde{P}^N)$ if and only if there exists a feasible point $(P^1,\ldots,P^N,\delta,W^1,\ldots,W^p)$ for (3.16) with $f^* < 0$, where f^* denotes the optimal function value of problem (3.16).*

Proof. The proof is similar to the proof of Lemma 3.1.

$\Rightarrow$: Let (3.5) hold for some $P = \tilde{P} \in \mathbb{S}^n$ with $\tilde{P} = \operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N)$. Set $\lambda_1 = \lambda_{\min}(\tilde{P})$ and $\lambda_2 = \lambda_{\min}\left(-A^{\mathsf{T}}\operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N) - \operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N)A\right)$. Then, $\tilde{\delta} := \min(\lambda_1, \lambda_2/\gamma) > 0$ holds. Furthermore, $(P^1, \ldots, P^N, \delta) = (\tilde{P}^1, \ldots, \tilde{P}^N, \tilde{\delta})$ and $(P^1, \ldots, P^N, \delta) = (0, \ldots, 0)$ are feasible for problem (3.6) and, due to the equivalency of (3.14) and (3.6b), there exist $\tilde{W}^s \in \mathbb{S}_+^{C_s}$ for $s = 1, \ldots, p$ such that $(P^1, \ldots, P^N, \delta, W^1, \ldots, W^p) = (\tilde{P}^1, \ldots, \tilde{P}^N, \tilde{\delta}, \tilde{W}^1, \ldots, \tilde{W}^p)$ and $(P^1, \ldots, P^N, \delta, W^1, \ldots W^p) = (0, \ldots, 0)$ are feasible for the problem (3.16). Hence, by convexity, for all $t \in [0, 1]$, $(P^1, \ldots, P^N, \delta, W^1, \ldots W^p) = (t\tilde{P}^1, \ldots, t\tilde{P}^N, t\tilde{\delta}, t\tilde{W}^1, \ldots, t\tilde{W}^p)$ is feasible for the problem (3.16). Now consider the objective function value $\Phi(t)$ at $(t\tilde{P}^1, \ldots, t\tilde{P}^N, t\tilde{\delta}, t\tilde{W}^1, \ldots, t\tilde{W}^p)$

$$-t\delta + \frac{\sigma_\delta}{2}t^2\delta^2 + \sum_{l=1}^{N}\frac{\sigma_{P^l}}{2N}t^2|||P^l||_F^2 + \sum_{s=1}^{p}\frac{\sigma_{W^s}}{2p}t^2||W^s||_F^2.$$

Since $\Phi'(0) = -\tilde{\delta} < 0$, note if we choose $0 < t \leq 1$ sufficiently small, then $\Phi(t) < \Phi(0) = 0$. We then set $(P^1, \ldots, P^N, \delta, W^1, \ldots W^p) = (t\tilde{P}^1, \ldots, t\tilde{P}^N, t\tilde{\delta}, t\tilde{W}^1, \ldots, t\tilde{W}^p)$ which has the desired properties.

$\Leftarrow$: We now show that if there exists a feasible point $(\tilde{P}^1, \ldots, \tilde{P}^N, \tilde{\delta}, \tilde{W}^1, \ldots, \tilde{W}^p)$ of (3.16) with negative objective function value, then $\tilde{\delta} > 0$ and $(\tilde{P}^1, \ldots, \tilde{P}^N)$ satisfies (3.5). As a negative objective function value can only be achieved if $\tilde{\delta} > 0$, the equivalency of (3.14) and (3.6b) yields that

$$A^{\mathsf{T}}\operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N) + \operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N)A \preceq -\gamma\tilde{\delta}I_n \prec 0$$

and

$$\tilde{P}^l \succeq \tilde{\delta}I_{n_l} \succ 0 \text{ for } l = 1, \ldots, N.$$

Setting $\operatorname{diag}(P^1, \ldots, P^N) = \operatorname{diag}(\tilde{P}^1, \ldots, \tilde{P}^N)$ finishes the proof. □

Finally, we rewrite $F(P, \delta)$ in a such way that it allows a decomposition. To this end, let $\mathscr{B}_l$ be defined as

$$\mathscr{B}_l = \left\{\sum_{i=1}^{l-1} n_i + 1, \ldots, \sum_{i=1}^{l} n_i\right\} \times \left\{\sum_{i=1}^{l-1} n_i + 1, \ldots, \sum_{i=1}^{l} n_i\right\}.$$

For $l = 1, \ldots, N$ and $(i, j) \in \mathscr{B}_l$, we define

$$F^0 = -\gamma I_n,$$
$$F_{ij}^l = \begin{cases} \frac{1}{2}\left(-A^{\mathsf{T}}E_{ij} - E_{ij}A\right) & \text{if } i < j, \\ \frac{1}{2}\left(-A^{\mathsf{T}}E_{ji} - E_{ji}A\right) & \text{if } i > j, \\ -A^{\mathsf{T}}E_{ij} - E_{ij}A & \text{if } i = j, \end{cases}$$

and with $i_l := i - \sum_{s=1}^{l-1} n_s$, we can rewrite $F(P, \delta)$ as

$$F(P, \delta) = F^0\delta + \sum_{l=1}^{N}\sum_{(i,j)\in\mathscr{B}_l} F_{ij}^l P_{i_l j_l}^l.$$

To solve problem (3.16) in parallel, we again employ the DCNA. To obtain the dual problem, consider the following Lagrangian of problem (3.16)

$$
\begin{aligned}
&\mathscr{L}(\delta, P, W, \Lambda, M) \\
&= -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 + \sum_{s=1}^{p} \frac{\sigma_{W^s}}{2p} \|W^s\|_F^2 \\
&\quad + \sum_{(i,j)\in J} \Lambda_{ij} \left(E_{ij} \bullet \sum_{s\in\Gamma(i,j)} W^s - E_{ij} \bullet F(P,\delta) \right) + \sum_{l=1}^{N} M^l \bullet \left(\delta I_{n_l} - P^l\right) \\
&= \underbrace{-\left(\sum_{(i,j)\in J} \Lambda_{ij} E_{ij} \bullet F^0 + 1 - \sum_{l=1}^{N} M^l \bullet I_{n_l} \right)}_{x_\delta} \delta + \frac{\sigma_\delta}{2}\delta^2 \\
&\quad + \sum_{l=1}^{N} \Bigg[\sum_{(i,j)\in\mathscr{B}_l} \underbrace{-\left(\sum_{(a,b)\in J} \Lambda_{ab} E_{ab} \bullet F_{ij}^l + M_{i_l j_l}^l \right)}_{=X^l_{P\,i_l j_l}} P_{i_l j_l}^l + \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 \Bigg] \\
&\quad + \sum_{s=1}^{p} \Bigg[\underbrace{- \sum_{(i,j)\in J(C_s)} -\Lambda_{ij} E_{ij}}_{X_W^s} \bullet W^s + \frac{\sigma_{W^s}}{2p} \|W^s\|_F^2 \Bigg] \\
&= -x_\delta \delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \left[-X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 \right] + \sum_{s=1}^{p} \left[-X_W^s \bullet W^s + \frac{\sigma_{W^s}}{2p} \|W^s\|_F^2 \right],
\end{aligned}
$$

which is separable in δ, $P^1, \ldots, P^N$, and $W^1, \ldots, W^p$. The corresponding dual function is

$$
\begin{aligned}
\varphi(\Lambda, M) &= \min_{\delta\in\mathbb{R}, P^l\in\mathbb{S}^{n_l}, W^s\in\mathbb{S}_+^{C_s}} \mathscr{L}(\delta, P, W, \Lambda, M) \\
&= \min_{\delta\in\mathbb{R}} \left\{ -x_\delta \delta + \frac{\sigma_\delta}{2}\delta^2 \right\} + \sum_{l=1}^{N} \min_{P^l\in\mathbb{S}^{n_l}} \left\{ -X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2N} \left\|P^l\right\|_F^2 \right\} \\
&\quad + \sum_{s=1}^{p} \min_{W^s\in\mathbb{S}_+^{C_s}} \left[-X_W^s \bullet W^s + \frac{\sigma_{W^s}}{2p} \|W^s\|_F^2 \right] \\
&= -x_\delta \delta(\Lambda, M) + \frac{\sigma_\delta}{2}\delta(\Lambda, M)^2 \quad + \sum_{l=1}^{N} \left[-X_P^l \bullet P^l(\Lambda, M) + \frac{\sigma_{P^l}}{2N} \left\|P^l(\Lambda, M)\right\|_F^2 \right] \\
&\quad + \sum_{s=1}^{p} \left[-X_W^s \bullet W^s(\Lambda, M) + \frac{\sigma_{W^s}}{2p} \|W^s(\Lambda, M)\|_F^2 \right],
\end{aligned}
$$

where $\delta(\Lambda, M)$, $P^l(\Lambda, M)$, and $W^s(\Lambda, M)$ are the unique solutions. As before, in the vector Lyapunov function case, $\varphi(\Lambda, M)$ can be evaluated in parallel and is continuously differentiable due to the uniqueness of the solutions $\delta(\Lambda, M)$, $P^l(\Lambda, M)$, and $W^s(\Lambda, M)$. Moreover,

the gradient of $\varphi(\Lambda, M)$ with

$$\nabla_{\Lambda_{ij}}\varphi(\Lambda,M) = E_{ij} \bullet \sum_{s\in\Gamma(i,j)} W^s(\Lambda,M)$$
$$-E_{ij}\bullet\left(F^0\delta(\Lambda,M)+\sum_{l=1}^{N}\sum_{(i,j)\in\mathscr{B}_l} F^l_{ij}P^l_{i_l j_l}(\Lambda,M)\right) \text{ for } (i,j)\in J$$

and

$$\nabla_{M^l}\varphi(\Lambda,M) = \delta(\Lambda,M)I_{n_l} - P^l(\Lambda,M) \text{ for } l=1,\dots,N,$$

is again Lipschitz continuous with Lipschitz constant

$$L = \sum_{s=1}^{p} p\|E_{C_s}\|^2/\sigma_{W^s} + \sum_{l=1}^{N} N\left(\left\|\hat{F}^l\right\|^2+1\right)/\sigma_{P^l} + \left(\sum_{(i,j)\in J}\left(E_{ij}\bullet F^0\right)^2 + n\right)/\sigma_\delta, \tag{3.17}$$

where $E_{C_s}\in\mathbb{R}^{|J(C_s)|\times n^2}$ is the matrix that contains the rows $\text{vec}(E_{ij})^\mathsf{T}$ for $(i,j)\in J(C_s)$ and $\hat{F}^l\in\mathbb{R}^{(|J|)\times n_l^2}$ is the matrix that contains rows $(E_{ab}\bullet F^l_{i^1 j^1},\dots,E_{ab}\bullet F^l_{i^{|\mathscr{B}_l|}j^{|\mathscr{B}_l|}})$ for $(a,b)\in J$. Finally, the DCNA can be applied to maximize the dual function in parallel to obtain Algorithm 5. Again, for ease of presentation, the aspect of event-based communication is not presented, but we refer to [68] for details.

A convergence result analogous to Theorem 3.3 can be made for Algorithm 5 and is stated in the following theorem.

Theorem 3.5. *Taking $k_{\max} = \lceil\sqrt{8L/\epsilon}\rceil - 1$ with $\epsilon > 0$ and Lipschitz constant L defined by (3.17), then after iteration $k_{\max}$ of Algorithm 5 an approximate solution to problem (3.16) is given by*

$$(\hat{P},\hat{W},\hat{\delta}) := \sum_{j=0}^{k_{\max}} \frac{2(j+1)}{(k_{\max}+1)(k_{\max}+2)}(P^{j+1},W^{j+1},\delta^{j+1})$$

and the following bounds on the primal gap are obtained

$$-\epsilon||\Omega^*||_F^2 \le -\hat{\delta}+\frac{\sigma_\delta}{2}\hat{\delta}^2 + \sum_{l=1}^{N}\frac{\sigma_{P^l}}{2N}||\hat{P}^l||_F^2 + \sum_{s=1}^{p}\frac{\sigma_{W^s}}{2p}||\hat{W}^s||_F^2 - f^* \le 0,$$

where $\Omega^* = \text{diag}(\text{diag}((\Lambda^*_{11},\dots,\Lambda^*_{nn})),M^{1*},\dots,M^{N*})$ denotes the matrix with optimal dual multipliers on its diagonal and f^* is the optimal function value of problem (3.16). Moreover, the following bound on the constraint violation is obtained

$$||\Pi(\hat{P},\hat{W},\hat{\delta})||_F \le \epsilon||\Omega^*||_F,$$

where $\Pi(P,W,\delta) = \text{diag}(\text{diag}(E(P,W,\delta)),[\delta I_{n_1}-P^1]^+,\dots,[\delta I_{n_N}-P^N]^+)$ and $E(P,W,\delta) := (E_{11}\bullet\sum_{s\in\Gamma(1,1)}W^s - E_{11}\bullet F(P,\delta),\dots,E_{nn}\bullet\sum_{s\in\Gamma(n,n)}W^s - E_{nn}\bullet F(P,\delta))^\mathsf{T}$. Here $[\ ]^+$ denotes the projection onto the space of symmetric positive semidefinite matrices.

Proof. The proof is similar to the proof of Theorem 3.3. □

Algorithm 5 Distributed solution of optimization problem (3.16).

For $k \geq 0$ do

1. Given the necessary $M^{l,k}$ and components Λ_{ij}^k, each subsystem computes

$$\delta^{k+1} = \arg\min_{\delta \in \mathbb{R}} \left\{ -x_\delta \delta + \frac{\sigma_\delta}{2} \delta^2 \right\}$$

using a consensus algorithm. Furthermore, the subsystems compute in parallel:

$$P^{l,k+1} = \arg\min_{P^l \in \mathbb{S}^{n_l}} \left\{ -X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2N} \left\| P^l \right\|_F^2 \right\},$$
$$W^{s,k+1} = \arg\min_{W^s \in S_+^{C_s}} \left\{ -X_W^s \bullet W^s + \frac{\sigma_{W^s}}{2p} \left\| W^s \right\|_F^2 \right\},$$

for $l = 1, \ldots, N$, $s = 1, \ldots, p$, and send $P^{l,k+1}$, and $W^{s,k+1}$ to their neighbors.

For $(i,j) \in J$ and $l = 1, \ldots, N$, the subsystems do in parallel:

2. Given δ^{k+1}, $P^{l,k+1}$, and $W^{s,k+1}$ compute

$$\nabla_{\Lambda_{ij}} \varphi(\Lambda^k, M^k) = E_{ij} \bullet \sum_{s \in \Gamma(i,j)} W^{s,k+1} - E_{ij} \bullet \left(F^0 \delta^{k+1} + \sum_{l=1}^{N} \sum_{(i,j) \in \mathscr{B}_l} F_{ij}^l P_{i_l j_l}^{l,k+1} \right),$$
$$\nabla_{M^l} \varphi(\Lambda^k, M^k) = \delta^{k+1} I_{n_l} - P^{l,k+1}.$$

3. Find

$$Y_{ij}^k = \arg\max_{Y_{ij} \in \mathbb{R}} \left\{ \nabla_{\Lambda_{ij}} \varphi(\Lambda^k, M^k) Y_{ij} - \frac{L}{2} \left(Y_{ij} - \Lambda_{ij}^k \right)^2 \right\},$$
$$H^{l,k} = \arg\max_{H^l \in \mathbb{S}_+^{n_l}} \left\{ \nabla_{M^l} \varphi(\Lambda^k, M^k) \bullet H^l - \frac{L}{2} \left\| H^l - M^{l,k} \right\|_F^2 \right\}.$$

4. Find

$$Z_{ij}^k = \arg\max_{Z_{ij} \in \mathbb{R}} \left\{ -\frac{L}{2} Z_{ij}^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{\Lambda_{ij}} \varphi(\Lambda^j, M^j) Z_{ij} \right\},$$
$$T^{l,k} = \arg\max_{T^l \in \mathbb{S}_+^{n_l}} \left\{ -\frac{L}{2} \left\| T^l \right\|_F^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{M^l} \varphi(\Lambda^j, M^j) \bullet T_l \right\}.$$

5. Set

$$\Lambda_{ij}^{k+1} = \frac{k+1}{k+3} Y_{ij}^k + \frac{2}{k+3} Z_{ij}^k,$$
$$M^{l,k+1} = \frac{k+1}{k+3} H^{l,k} + \frac{2}{k+3} T^{l,k}.$$

6. If $k = k_{\max}$ from Theorem 3.5, stop. Otherwise, increase k and go to Step 1.

These results yield the following theorem.

Theorem 3.6. *Given Assumptions 3.1 and 3.2, and the parameters $\epsilon_{\min}$ and ϵ_0, the subsystems (3.1) can make a distributed decision on the α-block diagonal Lyapunov stability of system (3.2) using Algorithms 5 and 4 using local model information according to Definition 3.1.*

Proof. Due to the definition of X_P^l and X_W^s in Step 1 of Algorithm 5 and the definition of $\nabla_{\Lambda_{ij}}\varphi(\Lambda^k, M^k)$ in Step 2, it follows that the required communication topology of Algorithm 5 equals the given communication topology of $\mathscr{G}_{\text{comp}}$ as we assumed $\mathscr{G}_{\text{LMI}}$ to be chordal in Assumption 3.2. Hence, Algorithm 5 uses only local model information according to Definition 3.1.

The rest of the proof is identical to the proof of Theorem 3.4. □

Note that all subproblems in the above algorithm have closed form solutions. For example, we derive the closed form solution for $W^{s,k+1}$ in Step 1: Consider the spectral decomposition of the symmetric matrix X_W^s

$$X_W^s = Q\Sigma Q^\mathsf{T} = (Q_+, Q_-)\begin{pmatrix}\Sigma_+ & 0\\ 0 & \Sigma_-\end{pmatrix}\begin{pmatrix}Q_+^\mathsf{T}\\ Q_-^\mathsf{T}\end{pmatrix},$$

where Σ_+ contains the non-negative eigenvalues of X_W^s. It follows that the optimal solution $W^{s,k+1}$ is the projection on the positive semidefinite part of X_W^s

$$W^{s,k+1} = \frac{pQ_+\Sigma_+Q_+^\mathsf{T}}{\sigma_{W^s}}.$$

Identically, solutions for $H^{l,k}$ and $T^{l,k}$ can be obtained.

3.4.3 Summary

In this section, we derived two different methods to check if an interconnected system is stable. Both tests are based on the stability conditions from the previous section, namely two optimization problems. The first condition is based on the vector Lyapunov function approach, the second on α-block diagonal Lyapunov stability. Using modern distributed optimization techniques in the form of the DCNA, two algorithms were obtained that allow the subsystems to decide on stability using only local model information. In particular, the subsystems only need to share model data and their optimization variables with neighboring systems. The illustration and validation of these results based on numerical experiments follow in the next section.

3.5 Numerical illustration and validation of stability tests

In this section, the two presented approaches for testing stability are evaluated. In the first subsection, both distributed algorithms are applied to test systems that satisfy both conditions, while in the second subsection only one condition is satisfied to show the reduced conservativeness. Finally, the tests are applied to a power system example.

3.5.1 Systems satisfying the vector Lyapunov condition

For the vector Lyapunov test from Section 3.4.1, 100 asymptotically stable systems with $N = 25$ subsystems are randomly created, with $n_i = 2\,\forall i$, and with connection probability between subsystems of 0.1. The created systems have the following properties: The minimal real parts of the eigenvalues of A are between -50.6 and -46.6, and the maximal ones are between -7.4 and -0.001. There are 82 to 190 directed edges (on average 133) between the subsystems, and there is never the case that a subsystem is connected to every other subsystem. They are also created in such a way that they satisfy Assumption 3.2 necessary to apply the α-block diagonal Lyapunov stability test.

To validate the results of the distributed Algorithm 3, we compare its results with a centralized implementation in Yalmip [84]. The comparison with Yalmip is not in terms of efficiency but only serves as validation with regards to the final obtained function values, because the goals of our distributed algorithm and a centralized implementation are different in terms of privacy vs. efficiency. The convexity parameters are chosen as $\sigma_\delta = 10^{-3}$, $\sigma_{d_i} = 10^{-3}$. The values of the cost function obtained with Yalmip lie between -18 and $-4.78 \cdot 10^{-4}$. The approximate result from the distributed algorithm is always smaller than the Yalmip result, as expected by Eq. (3.9). Using Algorithm 4 with $\epsilon_0 = 10^{-1}, \epsilon_{\min} = 10^{-5}$ and $\theta = 0.1$, we obtain the objective function differences between the values obtained with Yalmip and the values obtained with Algorithm 3 that are summarized in Table 3.1 with minimum, maximum and mean values. They show that the approximated solution given by Algorithm 3 improves with decreasing ϵ, as expected. Furthermore, the numerical computations show that, with decreasing ϵ, convergence to negative objective function values is observed in all 100 cases. Hence, all systems are correctly identified to be connectively stable by Theorem 3.4.

The systems are also analyzed with the distributed α-block diagonal Lyapunov stability test. We first compare the results of optimization problem (3.16) solved with Yalmip with the results from Algorithm 5. The convexity parameters are set to $\sigma_\delta = 10^{-3}, \sigma_{P^l} = 10^{-3}, \sigma_{W^s} = 10^{-5}$. Using Algorithm 4 with $\epsilon_0 = 10^{-1}, \epsilon_{\min} = 10^{-3}$, and $\theta = 0.1$ the differences in Table 3.2 are obtained, indicating that Algorithm 5 approximates the optimal solution well. Also, convergence to negative cost function values is observed for all 100 systems and hence, we correctly identify all systems to be α-block diagonally Lyapunov stable by Theorem 3.6. In conclusion, both methods identify all 100 stable systems correctly.

Furthermore, the numerical effort of both tests is compared. For the same accuracy $\epsilon = 10^{-3}$, the vector Lyapunov test requires between 173956 and 283095 iterations

Table 3.1: Difference in objective function value between Yalmip (f^*_{Yalmip}) and Algorithm 3 (f^*_ϵ) for different values of accuracy parameter ϵ for Section 3.5.1.

ϵ	$\min(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\max(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\text{mean}(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$
10^{-1}	0.0041	0.090	0.012
10^{-2}	$4.08 \cdot 10^{-4}$	0.0089	0.0012
10^{-3}	$4.08 \cdot 10^{-5}$	$8.91 \cdot 10^{-4}$	$1.16 \cdot 10^{-4}$
10^{-4}	$4.13 \cdot 10^{-6}$	$8.89 \cdot 10^{-5}$	$1.17 \cdot 10^{-5}$
10^{-5}	$4.5 \cdot 10^{-7}$	$8.90 \cdot 10^{-6}$	$1.19 \cdot 10^{-6}$

Table 3.2: Difference in objective function value between Yalmip (f^*_{Yalmip}) and Algorithm 5 (f^*_ϵ) for different values of accuracy parameter ϵ for Section 3.5.1.

ϵ	$\min(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\max(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\text{mean}(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$
10^{-1}	0.0019	0.041	0.0033
10^{-2}	$1.86 \cdot 10^{-4}$	0.0048	$3.89 \cdot 10^{-4}$
10^{-3}	$1.92 \cdot 10^{-5}$	$7.58 \cdot 10^{-4}$	$8.57 \cdot 10^{-5}$

while the α-block diagonal Lyapunov test requires between 1736306 and 2385591. In addition to the higher number of iterations, the individual iterations of the α-block diagonal Lyapunov test are more costly than the vector Lyapunov test because they involve matrix operations while the vector Lyapunov test involves only scalars on a subsystem level. The solution of LMIs that is part of the α-block diagonal Lyapunov test is inherently more costly than the linear program structure of the vector Lyapunov test. The overall size of the problem is also clearly smaller for the vector Lyapunov test because it uses a scalar approximation for every subsystem while the α-block diagonal Lyapunov test works with the complete model.

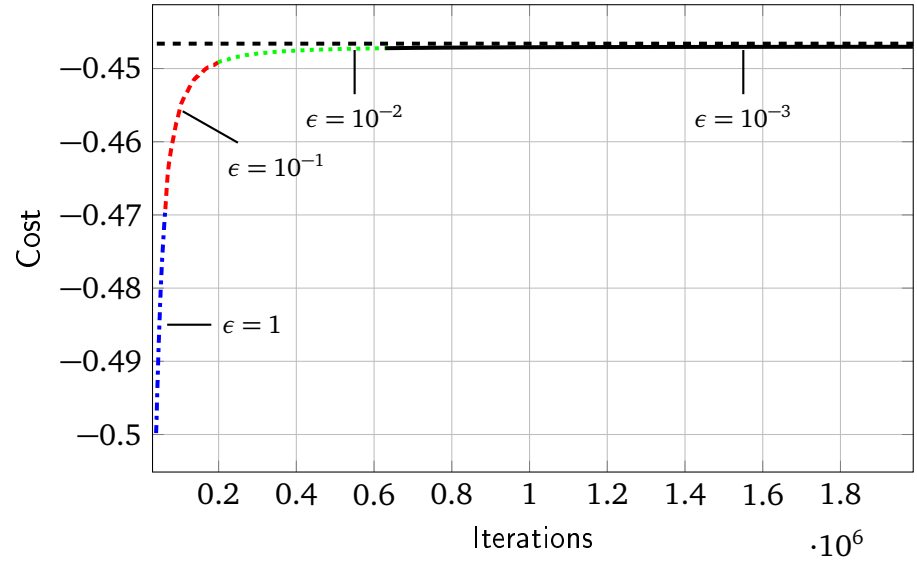
To visualize the principle behavior of the optimization algorithm and the influence of ϵ, the cost evolution for one example system is shown in Figure 3.2a where the α-block diagonal Lyapunov stability test is applied. The benefit of the additional number of iterations caused by a smaller value of ϵ can be observed. The evolution starts with a negative value and then approaches the optimal value from below. In Figure 3.2b, we see the decrease in distance from the actual optimal value which is always smaller than the respective ϵ.

3.5.2 Systems violating the vector Lyapunov condition

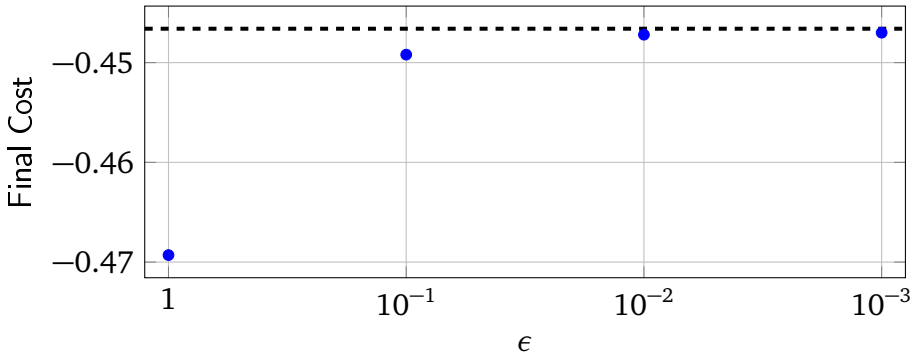
In Section 3.5.1, it is shown that the α-block diagonal Lyapunov stability test also identifies all systems to be asymptotically stable that satisfy the vector Lyapunov condition. In this subsection, again 100 systems are randomly created, each with $N = 25$ subsystems and $n_i = 2\,\forall i$, but in such a way that they fail the vector Lyapunov condition while satisfying the α-block diagonal Lyapunov condition. The minimal real parts of the eigenvalues of A lie between -22.03 and -18.6, the maximal ones between -2.6 and -0.03, and there are 78 to 197 directed interconnections. They are also ensured to satisfy Assumption 3.2.

Employing Algorithm 4 with $\epsilon_0 = 10^{-1}, \epsilon_{\min} = 10^{-3}$ and $\theta = 0.1$, the cost difference between Yalmip (values between -3.3 and -0.068) and Algorithm 5 are given in Table 3.3. This procedure indicates convergence to negative cost function values in all 100 cases. Hence, we conclude α-block diagonal stability by Theorem 3.6.

When the vector Lyapunov test is applied to the systems using Yalmip, the objective value is between $1.4 \cdot 10^{-13}$ and $1.76 \cdot 10^{-11}$, i.e. the systems do not satisfy the condition for connective stability. When applying Algorithm 3 with Algorithm 4, the obtained objective functions become closer and closer to zero with decreasing ϵ ($10^{-3} \to 10^{-6}$). While one cannot guarantee that the condition may be satisfied for even smaller values of ϵ, the subsystems have to choose an appropriate $\epsilon_{\min}$ to make their decision. Summarizing the two subsections, the α-block diagonal Lyapunov stability test identifies a larger class of systems at a higher numerical effort. This trade-off is summarized in Table 3.4. In addition, it can be observed that



(a) Cost evolution for different ϵ



(b) Final cost for different ϵ

Figure 3.2: Algorithm behavior for different values of ϵ. Final optimal value is -0.4466 in black (dashed).

the distributed solution can be very close to the actual, centralized solution, which illustrates the applicability of the distributed approach.

Naturally, there are systems that violate both the vector Lyapunov condition and the α-block diagonal Lyapunov stability condition. However, the α-block diagonal Lyapunov stability condition is clearly less conservative than the vector Lyapunov condition at the cost of an increased numerical cost. On the other hand, the vector Lyapunov condition establishes connective stability, i.e. a form of robust stability with respect to the interconnections. In practice, it is recommended to first check the simple vector Lyapunov condition and only check the α-block diagonal Lyapunov condition if the vector Lyapunov condition fails.

Table 3.3: Difference in objective function value between Yalmip (f^*_{Yalmip}) and Algorithm 5 (f^*_ϵ) for different values of accuracy parameter ϵ for Section 3.5.2.

ϵ	$\min(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\max(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$	$\text{mean}(\lvert f^*_\epsilon - f^*_{\text{Yalmip}}\rvert)$
10^{-1}	0.0017	0.0376	0.0076
10^{-2}	$1.89 \cdot 10^{-4}$	0.0038	0.0011
10^{-3}	$2.22 \cdot 10^{-5}$	0.0035	$4.86 \cdot 10^{-4}$

Table 3.4: Comparison between the vector Lyapunov test and the α-block diagonal Lyapunov test. The α-block diagonal Lyapunov test is clearly less conservative at a larger computational effort.

	vector Lyapunov test	Lyapunov Test
# identified stable systems out of 200	100	200
Average # iterations for $\epsilon = 10^{-3}$	156468	1751734

3.5.3 Application to 30 bus power system

In this subsection, we apply the presented tests to a power system model. The dynamics of each subsystem follow [85]

$$\begin{aligned}\dot{\delta}_i(t) &= 2\pi f_i(t),\\ \dot{f}_i(t) &= -\frac{f_i(t)}{T_{P_i}} - \frac{K_{P_i}}{2\pi T_{P_i}}\left(\sum_{j\in\mathcal{N}_{\text{in},i}} K_{S_{ij}}[\delta_i - \delta_j]\right) + \frac{K_{P_i}P_{g_i}}{T_{P_i}},\end{aligned}$$

where δ_i is the phase angle, f_i the frequency, P_{g_i} the generator output (input to the system), T_{P_i} the system model time constant, K_{P_i} the system gain and $K_{S_{ij}}$ the synchronizing coefficient of the tie-line between the ith and the jth area. The parameters for all areas are identical with $T_{P_i} = 25, K_{P_i} = 100, K_{S_{ij}} = 0.5$. As the original system is only marginally stable, we use a structured feedback for stabilization, and refer to Chapter 4 for details on how to achieve this.

The interconnection topology is taken from the IEEE 30 bus test case [86] and is also visualized in Figure 3.3. The system graph is not a chordal graph. Thus, for the α-block diagonal Lyapunov stability test, the computation graph needs to be at least the chordal extension of the system graph to allow a distributed test. The chordal extension is also shown in the figure.

Eventually, stability cannot be shown using the vector Lyapunov condition because of its inherent conservativeness, as the obtained function value is essentially 0. Applying the α-block diagonal Lyapunov stability test, the value of the cost obtained with Yalmip is -1.7004. Using the distributed Algorithm 5 with Algorithm 4 with accuracy values $\epsilon = \{1, 10^{-1}, 10^{-2}, 10^{-3}\}$, the following corresponding objective function values are obtained: $-1.7164, -1.7036, -1.7023, -1.7022$. One can see that, with decreasing ϵ, convergence to a negative cost function value is observed. Thus, the distributed α-block diagonal Lyapunov test indicates asymptotic stability of this practical example of a large-scale dynamical system

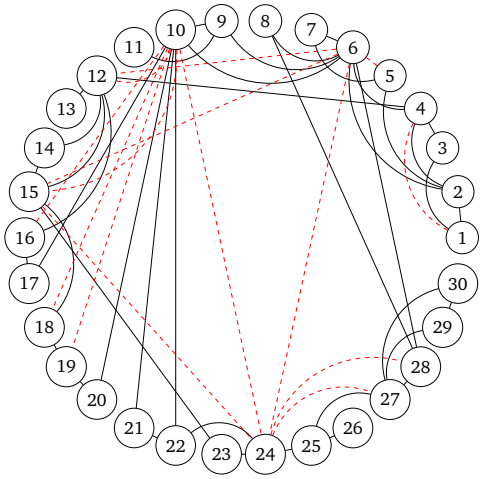


Figure 3.3: Topology of 30 bus power system, in red/dashed the additional edges of its chordal extension.

based on Theorem 3.6.

Remark 3.6. If one simply wishes to check stability of this system, there are methods, of course, which require a smaller computational effort. However, the purpose of the method is mainly to achieve a stability result without centralized model knowledge and this example illustrates the feasibility of the approach.

3.5.4 Summary

In this section, we illustrated the results from Section 3.4 using various numerical examples. We observed that the algorithms exhibit different conservativeness properties. While the vector Lyapunov test is far more conservative, its numerical effort is much smaller, which means that both approaches have their respective advantages. The conservativeness is the subject of the next section. Furthermore, we showed that the distributed algorithms lead to accurate results in comparison to the centralized solution

3.6 Analysis of the conservativeness of the stability conditions

In this section, we take a closer look at the conservativeness of the two stability conditions which stems from their respective sufficiency. We start by comparing the two conditions in terms of their specific forms to illustrate their differences. We then present a necessary condition for α-block diagonal Lyapunov stability. Subsequently, this result is analyzed and illustrated more deeply in a numerical experiment, which also gives an insight in the overall conservativeness of the α-block diagonal Lyapunov stability condition.

3.6.1 Comparison of the two conditions

In this subsection, we discuss the two tests with respect to differences in terms of conservativeness, and respective advantages and disadvantages. Both tests have in common that they use a block diagonal quadratic Lyapunov function. The vector Lyapunov function has the form $V_{\mathrm{M}} = \sum_{i=1}^{N} d_i x_i^{\mathsf{T}} (T_i^{\mathsf{T}})^{-1} T_i^{-1} x_i$ while the α-block diagonal Lyapunov stability approach has the Lyapunov function $V_{\mathrm{L}} = \sum_{i=1}^{N} x_i^{\mathsf{T}} P_i x_i$. Since both are restricted to a block diagonal form, both constitute only sufficient conditions. They also have in common that the summands need to represent Lyapunov functions of the individual subsystems. This means that both require the individual isolated subsystems to be stable. However, by comparing the structure of the Lyapunov functions V_{M} and V_{L}, one can see that V_{M} has only N degrees of freedom since the transformation matrices T_i are fixed. On the other hand, V_{L} has $\sum_{i=1}^{N} \frac{n_i(n_i+1)}{2}$ degrees of freedom, which reduces conservativeness except for the special case that all subsystems are scalar. Thus, the vector Lyapunov condition is more conservative because it evaluates the stability of the overall system using scalar (worst case) approximations of the individual subsystems. In fact, the vector Lyapunov test can be regarded as a special case of the α-block diagonal Lyapunov condition. It requires diagonal stability of the transformed system $\tilde{A}$ that is comprised of the blocks $\tilde{A}_{ij}$ given in Eq. (2.4), but using only N parameters on the diagonal instead of n. Clearly, the vector Lyapunov test is more conservative than the α-block diagonal Lyapunov test, but numerical investigations in Section 3.5 have shown that it has computational advantages. However, a difference that should be noted is that the conservativeness of the vector Lyapunov test is not introduced for the sake of the distributed solution but is inherent to the approach of vector Lyapunov functions. For the Lyapunov inequality test, the conservativeness stems from the block diagonal restriction necessary for the distributed solution. Furthermore, the α-block diagonal Lyapunov test has the disadvantage that the communication topology needs to be a chordal graph, so there may be a need for additional communication links instead of only the ones that are allowed by the computation graph $\mathcal{G}_{\mathrm{comp}}$. This is visualized in Figure 3.4 for a small example.

The conservativeness is illustrated with the following example.

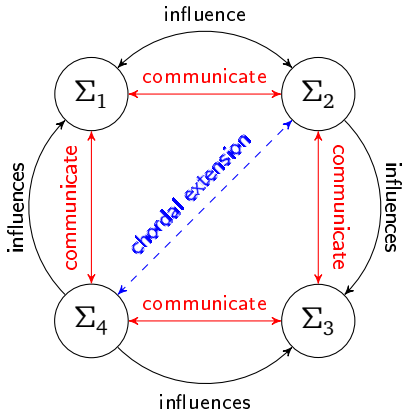


Figure 3.4: Communication topology for the two stability tests. M-Matrix test (red, solid), additional links for Lyapunov test (blue, dashed) because of chordal extension.

Example 3.6. The A-matrix of a system with two subsystems is given by

$$A = \left[\begin{array}{cc|cc} -1 & 0 & \frac{1}{2} & \frac{1}{10} \\ 0 & -\frac{1}{10} & -1 & -\frac{1}{2} \\ \hline 1 & 1 & -\frac{1}{2} & 0 \\ -\frac{3}{2} & \frac{1}{3} & 0 & -1 \end{array}\right]$$

The real parts of the eigenvalues of A are $(-1.37, -0.59, -0.32, -0.32)$ so all eigenvalues of A have negative real parts and, therefore, the system is asymptotically stable, and the interconnection topology is chordal. The test matrix W, according to (2.4), is $W = \left[\begin{smallmatrix} 0.16 & -1.15 \\ -1.92 & 0.54 \end{smallmatrix}\right]$ with eigenvalues $(-1.15, 1.85)$. One eigenvalue of W has negative real part so the matrix is not an M-matrix. However, it is possible to determine a block diagonal Lyapunov matrix P as

$$P = \begin{bmatrix} 49.23 & 12.16 & 0 & 0 \\ 12.16 & 40.76 & 0 & 0 \\ 0 & 0 & 35.36 & 9.58 \\ 0 & 0 & 9.58 & 17.47 \end{bmatrix},$$

which is positive definite and satisfies the Lyapunov inequality (3.19) showing α-block diagonal stability.

3.6.2 Necessary condition for α-block diagonal Lyapunov stability

In this subsection, we identify a necessary condition for the case that a system is α-block-diagonally Lyapunov stable.

Before we begin, we introduce the following notation. Given a matrix A and an index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$, the block submatrix $A_{(\alpha_{i_1}, \ldots, \alpha_{i_k})}$ for a set $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}$ is obtained by removing the blocks corresponding to the indices in the set $(\alpha_{j_1}, \ldots, \alpha_{j_l})$ with $\{j_1, \ldots, j_l\} = \{1, \ldots, N\} \setminus \{i_1, \ldots, i_k\}$.

Example 3.7. Given is the matrix

$$A = \left[\begin{array}{cc|cc|cc} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ \hline A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} \\ \hline A_{51} & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ A_{61} & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{array}\right]$$

with index partition $\alpha = (\alpha_1, \alpha_2, \alpha_3)$ with $\alpha_1 = \{1, 2\}, \alpha_2 = \{3, 4\}, \alpha_3 = \{5, 6\}$. Then, we have, for example,

$$A_{(\alpha_1, \alpha_2)} = \left[\begin{array}{cc|cc} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ \hline A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{array}\right]$$

and

$$A_{(\alpha_1,\alpha_3)} = \left[\begin{array}{cc|cc} A_{11} & A_{12} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{25} & A_{26} \\ \hline A_{51} & A_{52} & A_{55} & A_{56} \\ A_{61} & A_{62} & A_{65} & A_{66} \end{array}\right].$$

In order to obtain the necessary condition, we first need to introduce another notion of stability in the following definition.

Definition 3.3. Given a block index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$, a matrix $A \in \mathbb{R}^{n\times n}$ is called *totally BD-stable* if AB_D and all its block submatrices $AB_{D_{(\alpha_{i_1},\ldots,\alpha_{i_k})}}$ for all possible subsets $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}$ are Hurwitz stable for any positive diagonal matrix B_D.

A necessary condition for a matrix A to be totally BD-stable is given in the following lemma.

Lemma 3.4. *Given a matrix $A \in \mathbb{R}^{n\times n}$ and an index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$, the matrix A is totally BD-stable only if*

$$(-1)^s \det(A_{(\alpha_{i_1},\ldots,\alpha_{i_k})}) > 0, \tag{3.18}$$

for all possible subsets $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}$ and where s is the size of the block submatrix $A_{(\alpha_{i_1},\ldots,\alpha_{i_k})}$, i.e. $s = \sum_{l=1}^{k} n_{\alpha_{i_l}}$.

Proof. A necessary condition for Hurwitz stability of a matrix $A \in \mathbb{R}^{n\times n}$ is

$$(-1)^n \det(A) > 0,$$

because only then it is possible that all eigenvalues are negative. For A to be totally BD-stable, this needs to hold for all block submatrices of AB_D. Because $\det(B_D) > 0$ and because $\det(AB_D) = \det(A)\det(B_D)$, this requirement is equivalent to (3.18). □

We furthermore have the following lemma regarding α-block diagonal Lyapunov stability.

Lemma 3.5. *Given are a matrix $A \in \mathbb{R}^{n\times n}$ and an index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$. If the matrix A is α-block-diagonally Lyapunov stable, all its block submatrices $A_{(\alpha_{i_1},\ldots,\alpha_{i_k})}$ for all possible subsets $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}$ are also α-block-diagonally Lyapunov stable.*

Proof. This follows directly from the specialization of the quadratic form: Negative definiteness of (3.19) means that $x^\mathsf{T}(A^\mathsf{T}P + PA)x < 0$ for all $x \in \mathbb{R}^n$. That means that it must also hold for vectors x where elements corresponding to particular index subsets α_i are set to zero. □

The next lemma gives a relation between the two forms of stability, total BD stability and α-block diagonal Lyapunov stability.

Lemma 3.6. *Given are a matrix $A \in \mathbb{R}^{n\times n}$ and an index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$. If the matrix A is α-block-diagonally Lyapunov stable, it is also totally BD-stable.*

Proof. Assume that A is α-block-diagonally Lyapunov stable. Then there is a block diagonal P with block partition α such that

$$A^\mathsf{T}P + PA = Q \prec 0.$$

Pre- and post-multiplying by the diagonal matrix $B_D \succ 0$ gives

$$B_D A^\mathsf{T} P B_D + B_D P A B_D = B_D Q B_D \prec 0.$$

By defining $P_D := PB_D = B_D P$ we get

$$B_D A^\mathsf{T} P_D + P_D A B_D = B_D Q B_d \prec 0.$$

From this, it follows that AB_D is also α-block-diagonally Lyapunov stable with the Lyapunov matrix P_D. Using Lemma 3.5, this also holds for all submatrices with partition α. As α-block-diagonally Lyapunov stable (sub)matrices are also Hurwitz stable, it follows that A is totally BD-stable. □

The consequence of Lemma 3.6 is that the class of α-block-diagonally Lyapunov stable matrices is a subclass of the set of totally BD-stable matrices. Therefore, a necessary condition for totally BD-stable matrices is also necessary for α-block-diagonally Lyapunov stable matrices. To summarize this, we combine the result of Lemma 3.6 with Lemma 3.4 to arrive at the following Theorem, which is the main result of this subsection.

Theorem 3.7. *Given are a matrix $A \in \mathbb{R}^{n\times n}$ and an index partition $\alpha = (\alpha_1, \ldots, \alpha_N)$. The matrix A is α-block-diagonally Lyapunov stable only if*

$$(-1)^s \det(A_{(\alpha_{i_1},\ldots,\alpha_{i_k})}) > 0, \tag{3.19}$$

for all possible subsets $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, N\}$ and where s is the size of the block submatrix $A_{(\alpha_{i_1},\ldots,\alpha_{i_k})}$, i.e. $s = \sum_{l=1}^{k} n_{\alpha_{i_l}}$.

Proof. This follows directly from Lemmas 3.4, 3.5 and 3.6. □

The concept of a matrix being totally BD-stable extends the known notion of total D-stability [31] and the line of reasoning in this subsection is similar to the diagonal case, but the result is more general. In the next subsection, we analyze the implications of this theorem with a numerical experiment.

3.6.3 Numerical analysis of Theorem 3.7

In order to gain an intuition into when systems satisfy the necessary condition for α-block diagonal Lyapunov stability given by Theorem 3.7, we conduct an in-depth numerical experiment. To achieve that, we first consider the case of a network with $N = 6$ agents. In total, there are 112 possible different connected interconnection topologies. For each topology, we create 100 random systems in the following way: All diagonal blocks $A_{ii} \in \mathbb{R}^{2\times 2}$ are identical and all coupling matrices $A_{ij} \in \mathbb{R}^{2\times 2}$ are identical to each other, but A_{ii} and A_{ij} are not identical. At first, all entries of A_{ii} and A_{ij} are selected from a normal distribution. Afterwards, to tip the scale towards stability of the resulting matrix, the diagonal entries of A_{ii} are set to $-0.5 \times x$, where x is a number selected from a uniform distribution between 0 and 1. This gives us a total of 11200 different stable test systems.

With these systems, we check whether the block diagonal Lyapunov inequality (3.5) is satisfied where the block sizes of the six blocks are 2×2 and correspond to the subsystem sizes. Out of the 11200 stable systems, there are 1738 (15.51%) of them for which the block diagonal Lyapunov inequality is not satisfied. When condition (3.19) from Theorem 3.7 is applied to the 11200 test systems, it is observed that all 1168 systems that fail the condition also fail the α-block diagonal Lyapunov stability condition. However, there remain 570 systems that satisfy the condition but are still not α-block-diagonally Lyapunov stable, hinting that the condition is not sufficient. The condition is, however, a first test whether it is possible for a system to be α-block-diagonally Lyapunov stable – the system is not if it fails (3.19) – and in this numerical experiment, the condition covers about two thirds of the systems that are not α-block-diagonally Lyapunov stable.

We repeat the same analysis where we construct the systems in the same fashion, but the diagonal offset factor is set to the values 0.2 and 1.0, respectively, instead of 0.5. There is an additional fourth case where the blocks are not all identical. In this fourth case, the block sizes of the subsystems are either 2 or 3, and the block entries are created randomly where the diagonal is offset by the factor 1 as described earlier. This case is chosen to make sure that we are not only looking at the special case with all identical block sizes and block entries. The results of all four cases are summarized in Table 3.5. From the first three cases, it can be observed that the number of systems that are not α-block-diagonally Lyapunov stable increases with decreasing diagonal offset factor. This is to be expected as the dominance of the diagonal entry is decreased and with that the margin of stability. On the other hand, the percentage of systems identified by condition (3.19) is not correlated to the diagonal factor, but it is always roughly two thirds of the systems. What is remarkable about the fourth case is that more than 94% of the considered systems are not α-block-diagonally Lyapunov stable. It is not entirely clear why this is the case, but we assume that the difference in block entries plays a larger role than the difference in block sizes, and it is attributed to the way the test systems are created. Nevertheless, even in case 4, condition (3.19) identifies 70% of the systems that are not α-block-diagonally Lyapunov stable.

Next, we do the same analysis for a network with $N = 8$ agents. For eight agents, there are 11117 different possible connected topologies. The parameters are selected in the same fashion as in the case with $N = 6$ agents, and we obtain 1111700 different stable test systems. This means that there are about 100 times more test systems in this system class than in

Table 3.5: Analysis of α-block diagonal Lyapunov stability for all possible graphs with 6 vertices (112 graphs, each with 100 different A-matrices) with different diagonal entries. Cases 1-3: all subsystems have dimension 2, the diagonal entries are offset by the factors 0.5, 0.2 and 1, respectively. Case 4: subsystems have dimension 2 or 3, diagonal entries are offset by the factor 1.

Case	1	2	3	4
systems not α-block-diagonally Lyapunov stable	1738	2859	1377	10629
systems not α-block-diagonally Lyapunov stable [%]	15.5	25.5	12.3	94.9
systems identified by Theorem (3.7)	1168	1726	883	7512
systems identified by Theorem (3.7) [%]	67.2	60.4	64.1	70.7

the case of six subsystems. The systems are constructed in the same way as before with the diagonal offset of 0.5. Out of the 1111700 systems, 115268 (10%) are not α-block-diagonally Lyapunov stable. Condition (3.19) identifies 96343 (83.6%) of those systems. These results indicate that condition (3.19), while only necessary, can identify a large portion of the systems that are not α-block-diagonally Lyapunov stable.

To see if there is a correlation to properties of the connection graph, different graph measures are analyzed in addition to (3.19), namely algebraic connectivity, node degree, betweenness centrality, clustering coefficients, closeness centrality, number of cycles and eigenvalue centrality. For details on these graph measures, see A.2. However, no correlation at all could be observed between these purely structural properties and the results of the Lyapunov inequality.

Remark 3.7. Note that a similar condition to (3.19) exists for diagonal stability and it requires that this sign condition holds for all actual principal minors, instead of just the block minors as in our case in Theorem 3.7. Checking this condition, however, reveals that there are numerous systems that violate this requirement for diagonal stability while being α-block-diagonally Lyapunov stable. This is, of course clear, as this would, for example, require that the entry A_{11} is negative. The negativeness of this entry is required for diagonal stability, but not for α-block diagonal Lyapunov stability. Therefore, the introduction of the block-wise condition in the form of (3.19) makes sense.

3.6.4 Further comments on α-block diagonal Lyapunov stability

The numerical experiment from the previous subsection indicates that the conservativeness of the α-block diagonal Lyapunov stability condition is highly dependent on the system parameters, their interplay and the variance between them. This can be seen very well in the second row of Table 3.5 where all four cases exhibit the same interconnection topologies but vary dramatically in the portion of systems that are not α-block-diagonally Lyapunov stable. Unfortunately, as also seen in the previous subsection, we cannot state a necessary and sufficient condition for general LTI systems for the case that they are α-block-diagonally Lyapunov stable.

An issue that is related to the previous point is the following: For a full matrix P, Lyapunov stability is equivalent to Hurwitz stability which in turn is equivalent to the spectral property of the matrix A that all eigenvalues have negative real part. Neither diagonal [87] nor α-block diagonal Lyapunov stability is directly equivalent to a spectral property of the matrix A.

Another related difficulty is that diagonal stability and with that α-block diagonal Lyapunov stability is not a coordinate-free property [31] meaning that if the matrix A is not α-block-diagonally Lyapunov stable, there may be a coordinate transformation matrix T such that $T^{-1}AT$ is α-block-diagonally Lyapunov stable. In fact, for diagonal Lyapunov stability, it is known that such a transformation exists for any nonderogatory Hurwitz matrix [31] which means that the matrix has exactly one eigenvector per eigenvalue. Because diagonal Lyapunov stability is a subclass of α-block diagonal Lyapunov stable systems, this statement naturally extends to the block diagonal case. However, a coordinate transformation of this form is in general not block diagonal and, therefore, removes the original structure of the system, which makes the application of the presented distributed approach impossible.

3.6.5 Summary

In this section, we investigated the conservativeness of the two stability tests with an emphasis on the α-block diagonal Lyapunov stability case. First, we compared the two conditions and illustrated that the vector Lyapunov case is a special case of the α-block diagonal Lyapunov case. Then, we introduced a necessary condition for α-block diagonal Lyapunov stability and, using a large numerical experiment, we investigated how strict this condition is. We observed that the condition covers the majority of the systems that are not α-block-diagonally Lyapunov stable, but no sufficient condition was found.

3.7 Distributed model reduction using balanced truncation

In the previous sections, distributed optimization is used to investigate α-block diagonal Lyapunov stability of an interconnected system by distributedly solving a Lyapunov inequality. It is well known, however, that Lyapunov inequalities also play other roles in the field of control and system analysis. One of these roles is model reduction. Therefore, in this section, we adjust the Lyapunov inequality considered in the distributed optimization problem in order to apply the overall approach used in the previous sections to the different task of model reduction.

3.7.1 Introduction to model reduction using generalized gramians

In this subsection, we briefly recapitulate a common method for model reduction. Model reduction is an important issue, especially when dealing with large-scale systems. The purpose of model reduction is to find a model of smaller order than the original model that exhibits a behavior that is as close to the original model as possible. A well-known and standard tool to achieve this is balanced truncation [88]. For a short introduction on balanced truncation, please see Section A.3. The key idea is to use the controllability and observability gramians, or their generalized form, to determine so-called Hankel values and to use these Hankel values to choose which states to remove from the model. Essentially, the Hankel values are a measure for how large the influence of a state is in the input-output-behavior of the overall system. At the same time, the Hankel values provide an error bound of the reduced model in comparison to the actual one. In order to maintain the interconnection structure, the authors in [89] propose the use of structured gramians that contain a block diagonal structure. With this, the main steps to perform a distributed model truncation are the following.

1. Given a block index partition $\alpha = (\alpha_1, \dots, \alpha_N)$, solve the Lyapunov inequalities

$$AX_{\mathrm{c,g}} + X_{\mathrm{c,g}}A^{\mathsf{T}} + BB^{\mathsf{T}} \preceq 0, \tag{3.20a}$$
$$A^{\mathsf{T}}Y_{\mathrm{o,g}} + Y_{\mathrm{o,g}}A + C^{\mathsf{T}}C \preceq 0, \tag{3.20b}$$

   where the generalized controllability gramian $X_{\mathrm{c,g}}$ and the generalized observability gramian $Y_{\mathrm{o,g}}$ are restricted to a block diagonal structure with the block partition α.

2. Balance the gramians using a block diagonal coordinate transformation T such that $TX_{\mathrm{c,g}}T^{\mathsf{T}} = (T^{\mathsf{T}})^{-1}Y_{\mathrm{o,g}}T^{-1} = \Sigma$, where Σ is a diagonal, positive definite matrix. This

can be done distributedly for each individual subsystem using singular value decomposition. The result of this step is the set of the generalized Hankel singular values γ_i which are the diagonal entries of Σ.

3. Distributedly choose a desired order $r < n$ and truncate the overall system by the transformed states with Hankel singular values smaller than γ_r.

The most challenging step in a distributed setting is the first one. Nevertheless, as we have seen in Section 3.4.2, it is possible to distributedly solve a Lyapunov LMI of a similar form using distributed optimization. With that basis, we will see in the next subsection that a similar approach can be used to find generalized gramians.

3.7.2 Distributed optimization techniques for model reduction

In this subsection, we address the problem of distributed model reduction. The problem that we need to solve is to find solutions to the two LMIs (3.20). In order to achieve this using only local model information, we follow the same approach that was used in Sections 3.3.2 and 3.4.2. Instead of solving the feasibility problem of finding just any solutions to the two LMIs (3.20) we formulate two different optimization problems in order to apply the DCNA. The goal is to obtain two optimization problems that have the same form as problem (3.16). In order to avoid the repetition of the results from Section 3.4.2, we only state the main steps of our general approach which transforms the basic LMI conditions (3.20) into optimization problems:

1. Introduce a cost function of the form

$$f_{\mathrm{c,g}} = -\delta + \frac{\sigma_\delta}{2}\delta_{\mathrm{c,g}}^2 + \sum_{l=1}^{N} \frac{\sigma_X^l}{2N} ||X_{\mathrm{c,g}}^l||_F^2$$

   and the corresponding cost function for the observability problem.

2. Introduce new slack variables W_{c}^s and W_{o}^s with $s = 1, \ldots, p$ and augment the cost function with the new variables, as in the problem (3.16), to obtain

$$f_{\mathrm{c,g}} = -\delta + \frac{\sigma_\delta}{2}\delta_{\mathrm{c,g}}^2 + \sum_{l=1}^{N} \frac{\sigma_X^l}{2N} ||X_{\mathrm{c,g}}^l||_F^2 + \sum_{s=1}^{p} \frac{\sigma_{W^s}}{2p} ||W_{\mathrm{c}}^s||_F^2$$

   and the corresponding cost function for the observability problem.

3. Replace the LMI condition $F(P, \delta) \succeq 0$ with the respective controllability and observability LMI resulting in the form stated below.

This yields the following optimization problems.

$$\min_{\delta \in \mathbb{R}, X_{\mathrm{c,g}}^l \in \mathbb{S}^{n_l}} f(\delta_{\mathrm{c,g}}, X_{\mathrm{c,g}}^l) = -\delta_{\mathrm{c,g}} + \frac{\sigma_\delta}{2}\delta_{\mathrm{c,g}}^2 + \sum_{l=1}^{N} \frac{\sigma_X^l}{2N} ||X_{\mathrm{c,g}}^l||_F^2 + \sum_{s=1}^{p} \frac{\sigma_{W^s}}{2p} ||W_{\mathrm{c}}^s||_F^2 \tag{3.21a}$$

$$\text{s.t. } F_{\mathrm{c}}(X_{\mathrm{c,g}}, \delta_{\mathrm{c,g}}) \succeq 0, \tag{3.21b}$$

$$X_{\mathrm{c,g}}^l - \delta_{\mathrm{c,g}} I_{n_l} \succeq 0 \text{ for } l = 1, \ldots, N, \tag{3.21c}$$

and

$$\min_{\delta\in\mathbb{R},Y^l_{\mathrm{o,g}}\in\mathbb{S}^{n_l}} f(\delta_{\mathrm{o,g}},Y^l_{\mathrm{o,g}}) = -\delta_{\mathrm{o,g}} + \frac{\sigma_\delta}{2}\delta^2_{\mathrm{o,g}} + \sum_{l=1}^{N}\frac{\sigma_{Y^l}}{2N}||Y^l_{\mathrm{o,g}}||_F^2 + \sum_{s=1}^{p}\frac{\sigma_{W^s}}{2p}||W^s_{\mathrm{o}}||_F^2 \tag{3.22a}$$

$$\text{s.t. } F_{\mathrm{o}}(Y_{\mathrm{o,g}},\delta_{\mathrm{o,g}}) \succeq 0, \tag{3.22b}$$

$$Y^l_{\mathrm{o,g}} - \delta_{\mathrm{o,g}} I_{n_l} \succeq 0 \text{ for } l = 1,\dots,N, \tag{3.22c}$$

where

$$F_{\mathrm{c}}(X_{\mathrm{c,g}},\delta_{\mathrm{c,g}}) := -A\mathrm{diag}(X^1_{\mathrm{c,g}},\dots,X^N_{\mathrm{c,g}}) - \mathrm{diag}(X^1_{\mathrm{c,g}},\dots,X^N_{\mathrm{c,g}})A^{\mathsf{T}} - \gamma\delta_{\mathrm{c,g}}BB^{\mathsf{T}}, \tag{3.23}$$

and

$$F_{\mathrm{o}}(Y_{\mathrm{o,g}},\delta_{\mathrm{o,g}}) := -A^{\mathsf{T}}\mathrm{diag}(Y^1_{\mathrm{o,g}},\dots,Y^N_{\mathrm{o,g}}) - \mathrm{diag}(Y^1_{\mathrm{o,g}},\dots,Y^N_{\mathrm{o,g}})A - \gamma\delta_{\mathrm{o,g}}C^{\mathsf{T}}C, \tag{3.24}$$

and $\gamma \in \mathbb{R}_{++}$ is arbitrary. Based on these optimization problems we can state the following theorem, which gives a connection between them and the original LMI conditions (3.20).

Theorem 3.8. *The inequalities (3.20) hold for some block diagonal $X_{\mathrm{c,g}} = \tilde{X}_{\mathrm{c,g}}$ and $Y_{\mathrm{o,g}} = \tilde{Y}_{\mathrm{o,g}}$ with a given block partition α if and only if there exists a feasible point $(X_{\mathrm{c,g}},\delta_{\mathrm{c,g}})$ and $(Y_{\mathrm{o,g}},\delta_{\mathrm{o,g}})$ for (3.21) and (3.22) respectively, such that their optimal function values are negative, i.e. $f^*_{\mathrm{c,g}} < 0$ and $f^*_{\mathrm{o,g}} < 0$.*

Proof. The first part of the proof is identical to the proof of Lemma 3.1 because the overall problem formulation is identical and the objective function of (3.6) is also convex. This first part of the proof establishes that there exist solutions to the inequalities

$$AX_{\mathrm{c,g}} + X_{\mathrm{c,g}}A^{\mathsf{T}} + \gamma\delta_{\mathrm{c,g}}BB^{\mathsf{T}} \preceq 0,$$
$$A^{\mathsf{T}}Y_{\mathrm{o,g}} + Y_{\mathrm{o,g}}A + \gamma\delta_{\mathrm{o,g}}C^{\mathsf{T}}C \preceq 0,$$

when the optimal function values are negative. Furthermore, by scaling the two inequalities above, we obtain

$$A\frac{X_{\mathrm{c,g}}}{\gamma\delta_{\mathrm{c,g}}} + \frac{X_{\mathrm{c,g}}}{\gamma\delta_{\mathrm{c,g}}}A^{\mathsf{T}} + BB^{\mathsf{T}} \preceq 0,$$
$$A^{\mathsf{T}}\frac{Y_{\mathrm{o,g}}}{\gamma\delta_{\mathrm{o,g}}} + \frac{Y_{\mathrm{o,g}}}{\gamma\delta_{\mathrm{o,g}}}A + C^{\mathsf{T}}C \preceq 0,$$

and with that

$$A\tilde{X}_{\mathrm{c,g}} + \tilde{X}_{\mathrm{c,g}}A^{\mathsf{T}} + BB^{\mathsf{T}} \preceq 0,$$
$$A^{\mathsf{T}}\tilde{Y}_{\mathrm{o,g}} + \tilde{Y}_{\mathrm{o,g}}A + C^{\mathsf{T}}C \preceq 0,$$

where $\tilde{X}_{\mathrm{c,g}} = \frac{X_{\mathrm{c,g}}}{\gamma\delta_{\mathrm{c,g}}}$ and $\tilde{Y}_{\mathrm{o,g}} = \frac{Y_{\mathrm{o,g}}}{\gamma\delta_{\mathrm{o,g}}}$. Hence, $\tilde{X}_{\mathrm{c,g}}$ and $\tilde{Y}_{\mathrm{o,g}}$ are solutions to the inequalities (3.20) which concludes the proof. □

Given this result, we can apply the same approach as in Section 3.4.2 and use the DCNA to distributedly find the solutions to the optimization problems (3.21) and (3.22). The overall distributed model reduction approach is summarized in Algorithm 6.

We see that Algorithm 6 and Theorem 3.8 are another interesting application of distributed optimization techniques to solve classical problems from the control and system analysis field.

Algorithm 6 Distributed balanced truncation model reduction using DCNA.

1. Solve problem (3.21) using Algorithms 5 and 4 adapted to this problem.
2. Solve problem (3.22) using Algorithms 5 and 4 adapted to this problem.
3. Each agent locally balances the solution as follows: First, the agent performs a singular value decomposition of the matrix $X_{\mathrm{c,g}}^{i\frac{1}{2}} Y_{\mathrm{o,g}}^{i} X_{\mathrm{c,g}}^{i\frac{1}{2}}$, which gives

$$X_{\mathrm{c,g}}^{i\frac{1}{2}} Y_{\mathrm{o,g}} X_{\mathrm{c,g}}^{i\frac{1}{2}} = U^i \Sigma^i U^{i^*},$$

   where U^i is unitary and $\Sigma^i \succ 0$ is diagonal. Then, set $T_i^{-1} = X_{\mathrm{c,g}}^{i\frac{1}{2}} U^i$, which gives the state transformation matrix T_i and the generalized Hankel singular values from the entries of Σ^i.
4. The agents distributedly decide which Hankel values, and with that, which states are truncated by repeatedly using a minimum consensus approach [90]. This is repeated until a specific desired accuracy of the reduced model, or a desired system order is achieved.

3.7.3 Numerical examples

In this subsection, we illustrate the model reduction approach from this section using numerical examples. First, we give a small scale illustrating example and then apply the approach to the power system model introduced for the stability test example in Section 3.5.3.

3.7.3.1 Illustrating example

To illustrate the overall approach, a small-scale example is used. We consider $N = 4$ subsystems in a line-graph topology where each subsystem has $n_i = 4$ states, giving us a total system order of $n = 16$. The A-matrix of the system is created randomly where the entries are picked from a normal distribution. The diagonal entries are replaced by larger negative entries to obtain a stable system. The resulting system matrices are

$$A = \begin{bmatrix}
-3.1 & -0.7 & -1.7 & -1.1 & -1.0 & -0.8 & 0.2 & -1.6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1.4 & -7.6 & 0.9 & 0.2 & 0.3 & -0.7 & 1.4 & -1.2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1.6 & -0.4 & -9.6 & 0.5 & -1.9 & 0.1 & 1.3 & -1.7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-0.1 & -0.6 & 0.4 & -9.2 & 1.1 & 1.5 & 0.4 & 0.7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1.8 & 0.3 & 0.5 & 1.3 & -3.3 & -1.6 & -0.3 & 0.2 & 0.3 & 0.1 & 0.7 & 1.1 & 0 & 0 & 0 & 0 \\
0.6 & -1.0 & -0.6 & 0.5 & 0.2 & -8.3 & 0.2 & 0.8 & -0.1 & 0.1 & -1.3 & 0.2 & 0 & 0 & 0 & 0 \\
-0.7 & -0.4 & -0.3 & 0.1 & -0.3 & -1.2 & -6 & -0.9 & 0.2 & -1.0 & 0.3 & -0.3 & 0 & 0 & 0 & 0 \\
-1.1 & -1.0 & -0.6 & 0.4 & 0.6 & 0.1 & -1.1 & -7.1 & 0 & -1.7 & -0.7 & -1.2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.5 & 0.4 & 2.6 & 1.0 & -7.3 & 1.1 & -1.2 & 2.7 & 1.2 & 0.6 & 0.9 & -0.1 \\
0 & 0 & 0 & 0 & 1.0 & -0.3 & -0.2 & 0 & -0.6 & -3.5 & -0.7 & 0.1 & 0 & -1.5 & -0.1 & -0.3 \\
0 & 0 & 0 & 0 & -1.7 & 0.3 & 0.8 & 0 & 1.3 & 0.5 & -6.3 & -0.3 & -0.4 & 0.7 & 1.3 & -0.2 \\
0 & 0 & 0 & 0 & 0 & -0.3 & -0.9 & -0.2 & 0.2 & -0.7 & -0.2 & -3.2 & 0.7 & -0.9 & -0.4 & 1.4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & -1.3 & 1.7 & 1.9 & -0.2 & -0.4 & 0.2 & 0.2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.3 & 1.1 & -0.4 & 1.1 & 0.8 & -1.3 & 0.9 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.2 & 0.1 & 0.8 & 0.3 & -0.1 & 0.1 & -2.1 & 0.1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.9 & -1.3 & -0.5 & -1.0 & 0 & -2.4 & -0.3 & -0.4
\end{bmatrix},$$

$$B = \operatorname{diag}\left(\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T, \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T, \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T, \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T\right),$$
$$C = \operatorname{diag}\left(\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}, \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}, \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}, \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}\right).$$

In the following, we apply Algorithm 6 to the above system. Furthermore, we do the same model reduction based on the optimization problems (3.21) and (3.22) but with the standard LMI software Yalmip [84] as a benchmark for our distributed solution. Additionally, we look at the model reduction based directly on the Lyapunov inequalities (3.20) both with block diagonal restriction and with full gramian matrices. In all four cases, we choose to reduce the system order by four, such that only 12 states remain. It has to be noted that, in general, the resulting behaviors of the four cases cannot be compared directly because the solutions of LMIs are not necessarily unique leading to different model reductions.

The first observation is that the distributed solution using Algorithm 6 gives practically identical results to the solution based on Yalmip. The largest relative difference in resulting generalized Hankel singular values is 0.353%. This indicates again that the distributed algorithm performs well. Next, we need to evaluate the model reduction. The reduction leads to the removal of two states each of the first two subsystems. In Figure 3.5, we see four of the 16 step responses of the reduced system. The four selected responses are those from input i to output i, $i = 1, \ldots, 4$. For the second subsystem, a deviation can be observed stemming from the removed states. In absolute terms the deviation is, however, still quite small. The effect on the first subsystem is negligible and the third and fourth subsystems behave completely identical to the full system model. Similarly, in Figure 3.6, we see the Bode magnitude plot of the same input-output-combinations and we can make the same observations. The second subsystem stands out in that the transfer functions do not coincide. For the first subsystem they do at least up to a certain frequency. The third and fourth subsystem have identical transfer functions.

Next, we look at the model reduction based on the Lyapunov inequalities (3.20) with block diagonal restriction. In this case, removing the four smallest generalized Hankel singular values leads to the removal of three states of the first subsystem and one of the second. This is reflected in the step responses in Figure 3.7, where now the second subsystem of the reduced model behaves quite similarly to the full model while there is a larger discrepancy for the first one. Again, the third and fourth subsystems are unaffected.

Last, the case of the Lyapunov inequalities (3.20) with full gramian matrices is considered. The model reduction procedure removes one state from subsystem three, and three states from subsystem four. This is in accordance with the differences observed in the step-responses in Figure 3.8.

In conclusion, all four variants perform satisfactorily and there is no clear advantage visible. Notably, however, the proposed distributed approach performs reasonably well and does not need a complete system model, which opens up new possibilities for distributed model reduction.

3.7.3.2 Application to 30 bus power system

In the following, we apply Algorithm 6 to the 30 bus power system example system from Section 3.5.3. For details on the system dynamics, we refer to that section. The original system order is $n = 60$.

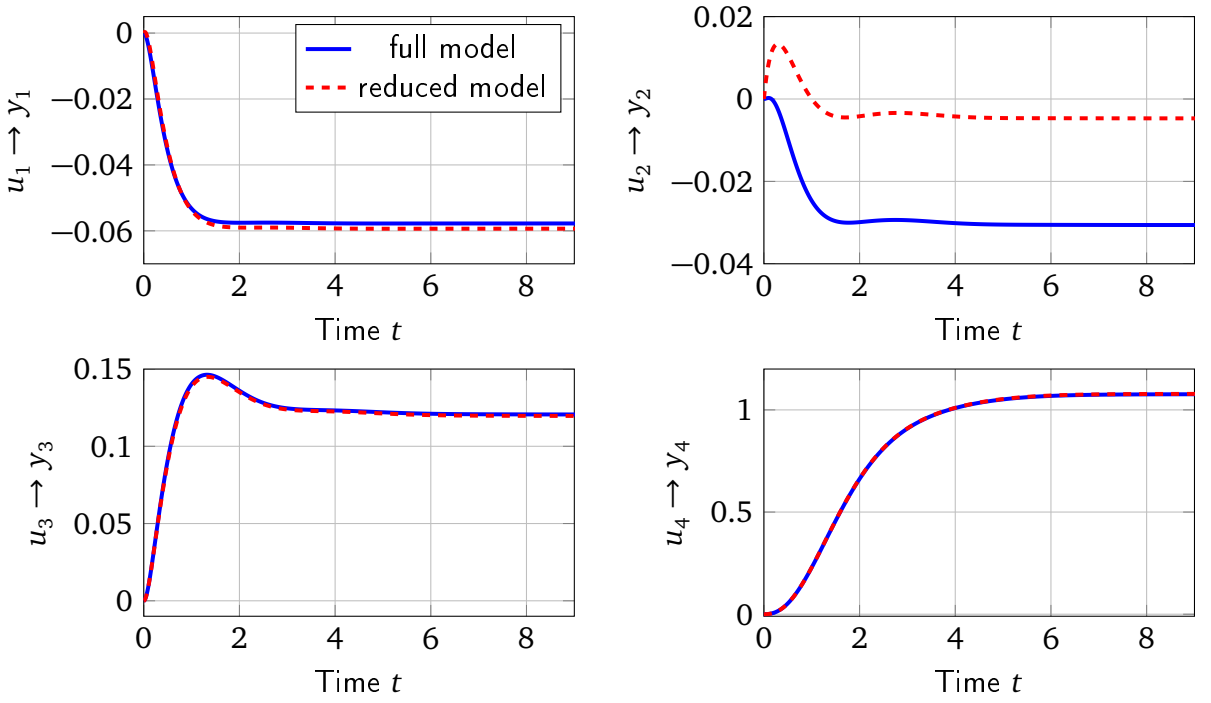


Figure 3.5: Comparison of step responses of system with $N = 4$ agents. Full model (blue, solid) $n = 16$, reduced model (red, dashed) $n = 12$. Balanced truncation with Algorithm 6.

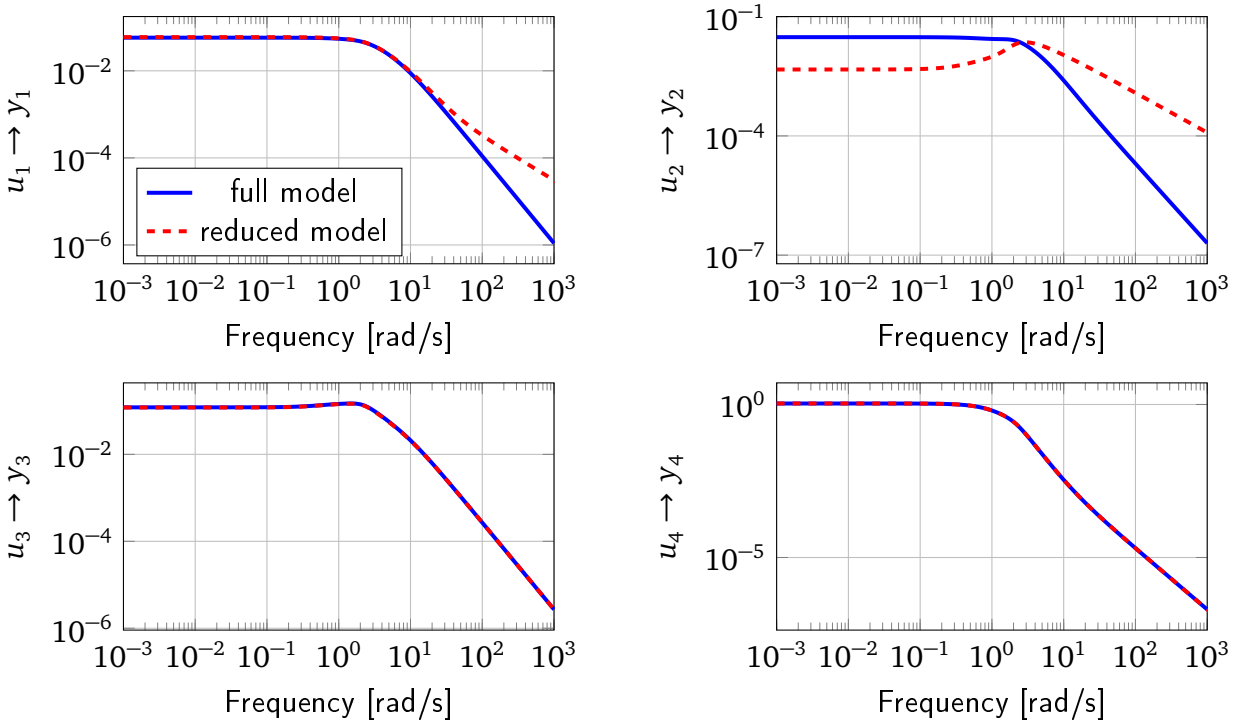


Figure 3.6: Comparison of transfer functions of system with $N = 4$ agents. Full model (blue, solid) $n = 16$, reduced model (red, dashed) $n = 12$. Balanced truncation with Algorithm 6.

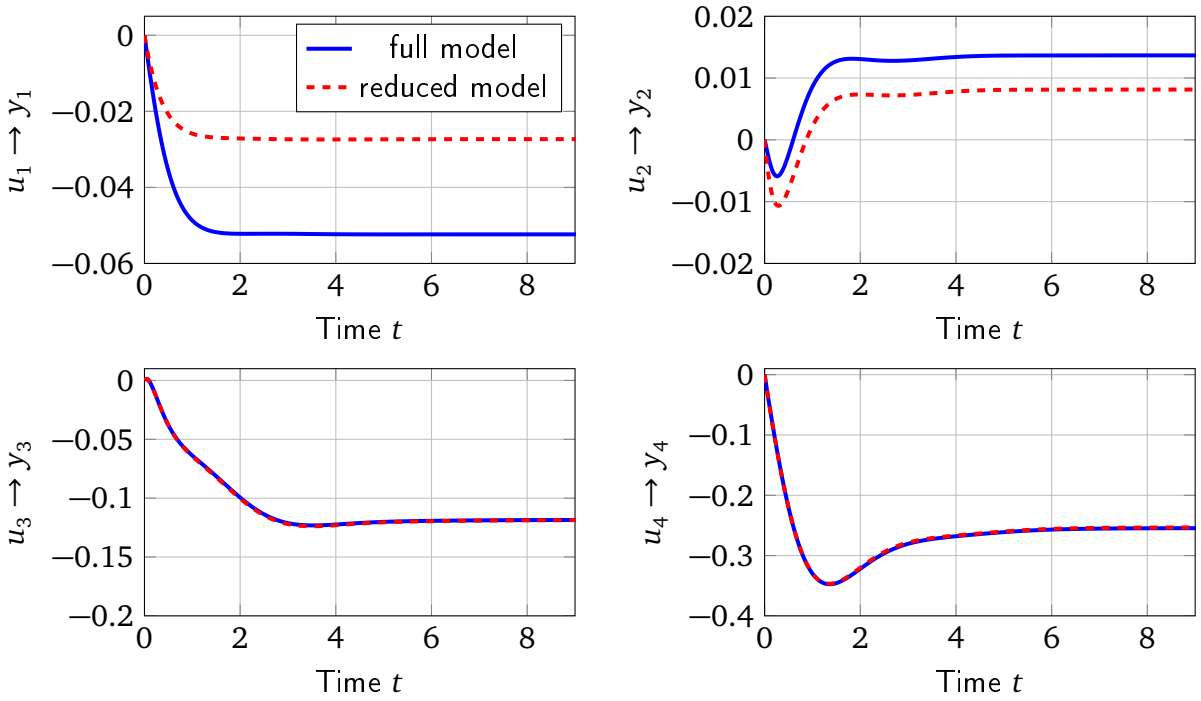


Figure 3.7: Comparison of step responses of system with $N = 4$ agents. Full model (blue, solid) $n = 16$, reduced model (red, dashed) $n = 12$. Balanced truncation with Lyapunov inequalities (3.20) with block diagonal restriction.

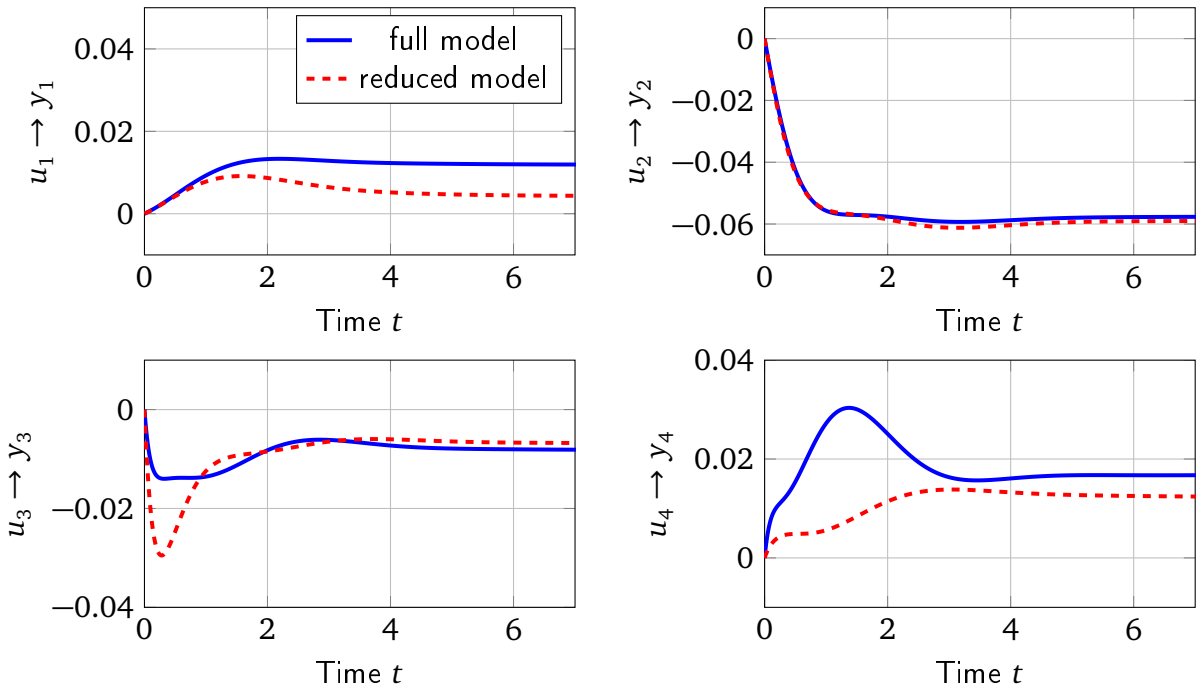


Figure 3.8: Comparison of step responses of system with $N = 4$ agents. Full model (blue, solid) $n = 16$, reduced model (red, dashed) $n = 12$. Balanced truncation with Lyapunov inequalities (3.20) with full gramian matrices.

As for the previous example, Algorithm 6 gives results that are practically identical to the solution based on Yalmip. We then decide to remove 15 states which reduces the system order by 25%. Because the presentation of the analysis of all step and frequency responses for every input-output combination would be too exhaustive we pick four as examples, namely those from input i to output i for $i = 1,\ldots,4$. The resulting step responses and Bode magnitude plots are shown in Figure 3.9 and 3.10. In the step responses, we can see that for subsystems one, two and four, there is practically no difference in the behavior. The third subsystem is affected by the changes in that the steady state changes by 19%. This can be explained by the fact that out of the four systems, only subsystem three has a state removed. However, note that subsystems one and four are neighbors of subsystem three, but there is no visible effect of the model reduction on them. A similar observation can be made for the frequency response. Only subsystem three is obviously affected by the model reduction. The responses are similar up to a certain frequency at about 10rad/s but then diverge.

Similar observations can be made for the other 26 subsystems where those affected by the reduced order show slightly different behavior in the reduced case. This example illustrates the applicability of the approach to larger systems. It also shows that the reduced model is not in perfect alignment with the full model, but of course this is expected.

3.7.4 Summary

In this section, a new idea was presented that allows to reduce a dynamic model in a distributed fashion. The overall approach follows the same line of thinking as the distributed

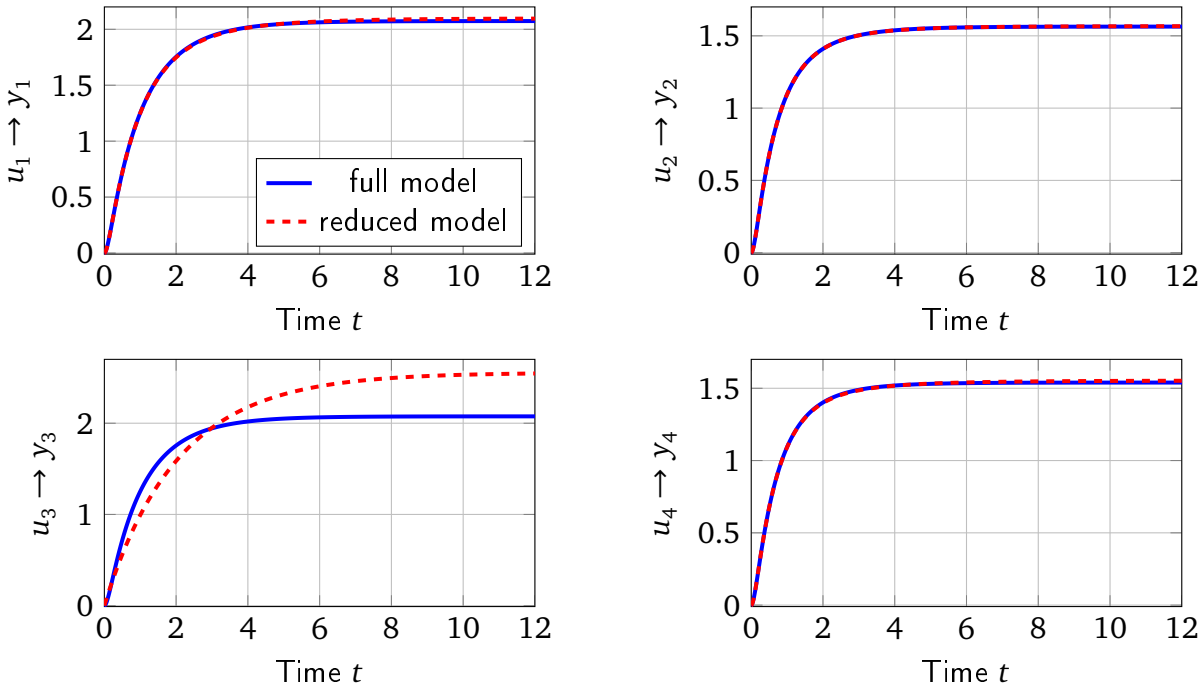


Figure 3.9: Comparison of step responses of 30 bus power system system for inputs/outputs 1-4. Full model (blue, solid) $n = 60$, reduced model (red, dashed) $n = 45$. Balanced truncation with Algorithm 6.

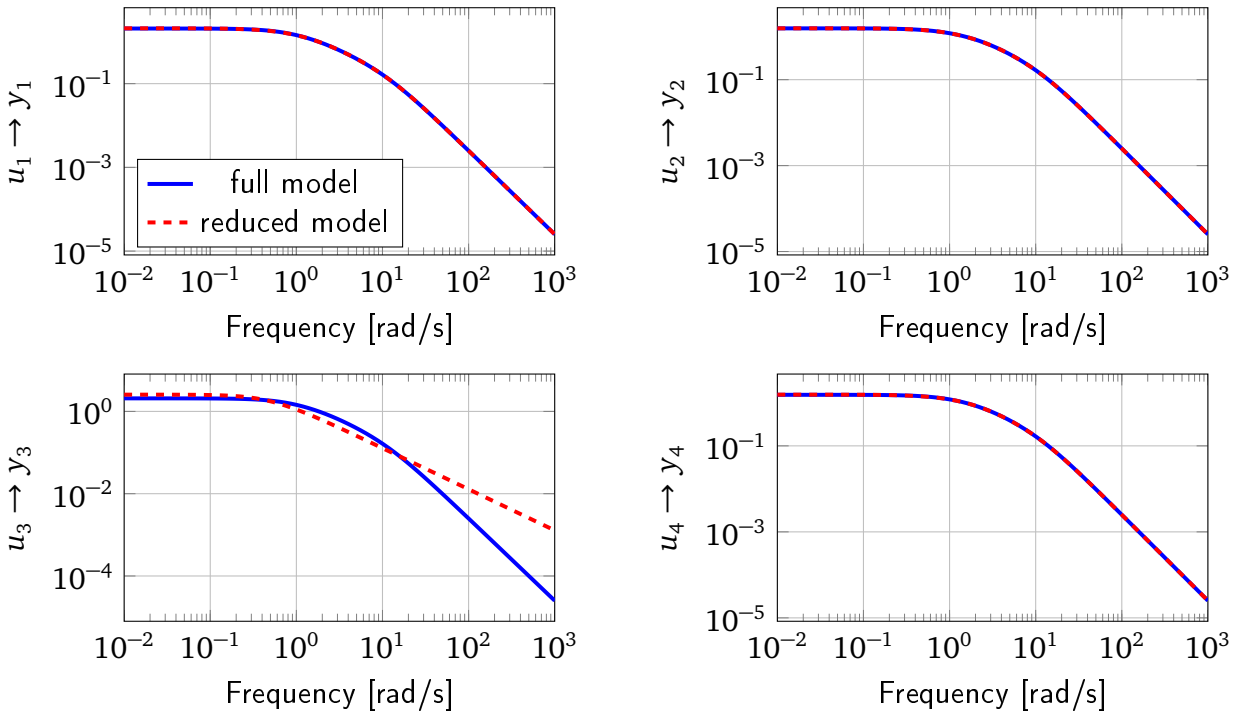


Figure 3.10: Comparison of transfer functions of 30 bus power system system for inputs/outputs 1-4. Full model (blue, solid) $n = 60$, reduced model (red, dashed) $n = 45$. Balanced truncation with Algorithm 6.

stability tests of the previous section. Lyapunov LMIs are solved distributedly based on the formulation of a distributed optimization problem that contains the LMIs as constraints. Then, the DCNA can be applied to find the solution. This allows a model reduction based on balanced truncation that takes the overall system model into account and the approach is based on well-known and established model reduction tools and results. The results were illustrated using a small-scale numerical experiment, and the approach was applied to a 30 bus power system model.

3.8 Chapter summary

In this chapter, we developed two different distributed stability tests. These two tests enable us to make a distributed decision on the stability of an interconnected dynamical system using only local model information. The first test is based on the concept of vector Lyapunov functions and consists of the test of a matrix to see if it is an M-matrix. The second test is based on the block diagonal form of the Lyapunov inequality. Both stability conditions are first reformulated into optimization problems. The structure of these optimization problems allow a distributed solution based on the application of the distributed constrained Nesterov algorithm (DCNA).

In addition to the development of the conditions, we also analyzed them in terms of conservativeness and in terms of their numerical behavior. For the α-block diagonal Lyapunov stability case, a necessary condition was found, and numerical investigations show that a

large portion of systems that are not α-block-diagonally Lyapunov stable can be identified using it.

Finally, we saw that the same distributed optimization approach used in the stability analysis can also be employed to achieve distributed model reduction based on Lyapunov inequalities and balanced truncation.

All results of this chapter were corroborated using numerical investigations and illustrations.

Note that the results of this chapter are partly based on the work in [20].

4

Stabilizing distributed optimal control design with local model information

This chapter addresses the second aspect of this thesis, namely the distributed optimal control design with local model information. The control law exhibits a prescribed control law structure and is optimal with respect to a linear quadratic (LQ) cost functional. The solution of the optimal control problem is obtained iteratively using a gradient descent approach. The employed gradient descent method is described thoroughly including details on the distributed step size selection, how to guarantee convergence, and extensions to a conjugate gradient method. The control design guarantees stability of the closed loop. This is achieved by making use of a terminal cost term that can be computed in a distributed fashion by adapting Algorithm 5 from Chapter 3.

As the distributed approach is based on simulated trajectories, which leads to a large requirement of communication, we also present a method to reduce this communication effort by using event-based information exchange. Furthermore, the control design method is extended to singular, differential-algebraic systems. Then, the approach is applied to the problem of optimal formation control in a multi-robot setting. Finally, we present two generalized forms of the control law.

The remainder of this chapter is organized as follows. Section 4.1 contains the problem statement. This is followed by a literature overview on related work in Section 4.2. In Section 4.3, we provide the solution to the considered distributed control problem. Results on distributed step size selection are presented in Section 4.4. We address event-based trajectory simulation in Section 4.5, and show the two extensions of the approach to different problem classes in Sections 4.6 and 4.7. Last, two generalizations to time-varying and nonlinear control laws are presented in Section 4.8. We conclude with a summary in Section 4.9.

4.1 Problem formulation

In this section, we state the system dynamics considered for the majority of this chapter, formulate the considered control design problem, and refine the definition of local model information.

The system dynamics we consider in this chapter are the ones already introduced in Section 2.4 in Equations (2.8),(2.9). For ease of reference, we restate them here. Each subsystem has the dynamics described by

$$\dot{x}_i(t) = A_{ii}x_i(t) + \sum_{\substack{i=1 \\ i\neq j}}^{N} A_{ij}x_j(t) + B_i u_i(t), \tag{4.1}$$

with $x_i \in \mathbb{R}^{n_i}$, $u_i \in \mathbb{R}^{m_i}$, $A_{ij} \in \mathbb{R}^{n_i \times n_j}$, and $B_i \in \mathbb{R}^{n_i \times m_i}$. The overall system is compactly written as

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{4.2}$$

where $x = [x_1^\mathsf{T}, ..., x_N^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^n$, $u = [u_1^\mathsf{T}, ..., u_N^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^m$, $\sum_{i=1}^N n_i = n$ and $\sum_{i=1}^N m_i = m$.

To describe the problem in a concise way, we state the following definition which constitutes what we mean by *local model information* in this chapter.

Definition 4.1. The set $\mathscr{K}$ consisting of *feasible control laws under local model information* are those control laws $u = -Kx$ that satisfy the following two conditions:

1. The design process of the control law is restricted to the computation graph $\mathscr{G}_{\text{comp}}$, i.e. during the control design information is exchanged only along the edges of that graph. Additionally, the subsystems know the overall system size N and the number of states n.

2. The structure of the feedback law is in accordance with the control graph $\mathscr{G}_{\text{control}}$, i.e. $K_{ij} \neq 0$ iff $(j,i) \in \mathscr{E}_{\text{control}}$.

A design method satisfying these conditions requires only local model information in accordance with Definition 2.8. We emphasize here that we consider the word *distributed* to apply in two phases, both during the control design itself, and also in the online use of the control law. The first part is often neglected. Therefore, the goal of this part of the thesis is to design a stabilizing distributed feedback matrix K_{dist} that satisfies Definition 4.1 and optimizes an LQ cost functional. This goal is summarized concisely in the following problem.

Problem 2. Given system (4.1),(4.2), find the solution to the following optimization problem using only local model information in accordance with Definition 4.1:

$$\min_{x,u} J_\infty(t,x,u) = \int_t^\infty x^\mathsf{T}(\tau)Qx(\tau) + u^\mathsf{T}(\tau)Ru(\tau)\mathrm{d}\tau, \tag{4.3a}$$

$$\text{s.t. } \dot{x}(\tau) = Ax(\tau) + Bu(\tau),\ x(t) = x_t \tag{4.3b}$$

$$u(\tau) = -K_{\text{dist}}x(\tau), \tag{4.3c}$$

$$K_{\text{dist}} \text{ is stabilizing}, \tag{4.3d}$$

$$K_{\text{dist}} \in \mathscr{K}, \tag{4.3e}$$

where $Q \in \mathbb{S}_+^n$ and $R \in \mathbb{S}_{++}^m$

Note that the constraint (4.3d) is already implicitly included in (4.3a)-(4.3c) because (4.3a) only has a finite value for a stable closed-loop system, but we state it for clarity.

4.2 Related work

In Section 2.2.2, we already presented an overview on results on distributed control. All the results mentioned there have the unifying property that the control design is performed from a centralized perspective. This means that while the implementation of the control law, that is the computation of the actual control inputs, is in accordance with a sparse interconnection structure the design requires a central and complete system model. The presented methods in this chapter circumvent this requirement. In addition, they offer some degrees of freedom in terms of the structure of the control law, and like many of the results mentioned in Section 2.2.2 they make use of optimization techniques such as LMIs, dual decomposition and descent methods. Other work has tried to achieve similar goals, which is described in the following.

One direction that tries to minimize central coordination with the use of optimization tools is distributed model predictive control (MPC) [85, 91, 92, 93, 94]. In [91], a dual decomposition approach is used to distributedly find an optimal input trajectory. However, to show stability, the method requires global knowledge about the system behavior, and a solution of the distributed optimization problem may be too slow for a real-time MPC application. The convergence speed is improved in [92] using the Nesterov algorithm. For a broader overview on this research direction, there are also several survey articles [85, 93, 94].

The control design with limited, local model information has been subject to some research efforts in the last few years [95, 96, 97, 98, 99, 100]. The authors of [95] consider scalar, discrete-time subsystems and they investigate the best achievable performance when the subsystems only know their own row of the system dynamics matrix during the control design, and show that a deadbeat strategy is the optimal one. These results are extended to multidimensional systems in [96] where, in addition, other design topologies are consid-

ered. The systems, however, need to be fully actuated. For stochastically varying systems, the value of model information is investigated in [97] where it is assumed that the subsystems know the mean and variance of the overall system dynamics. This latter aspect, however, has a negative influence on the scalability of the approach. A similar approach to the one presented in this chapter is derived in [98]. There, a distributed gradient method is used to iteratively improve a feedback matrix and the individual subsystems exchange information only with direct neighbors. However, there is no distributed stopping criterion for the distributed algorithm, no convergence guarantee and no stability guarantee. The authors of [99] consider the finite horizon LQ problem and investigate conditions such that an optimal input trajectory can be determined using information according to a given optimization graph. However, their result is not a feedback law and stability is not considered. For the special case of vehicle systems on a line graph, a distributed design method for LQ optimized feedback laws is presented in [100].

The mentioned results make limiting assumptions on the dynamics or the interconnection structure, or they do not guarantee convergence or stability. Therefore, the methods in this chapter solve these problems and guarantee stability, convergence and are valid for general LTI systems. The work on distributed MPC relies on the repeated distributed online solution of large optimization problems to obtain an open-loop input trajectory. This may not be possible in real-time. For this reason, the design method presented in this chapter yields a feedback law, which gives an additional degree of robustness with regards to computational effort, possible communication problems and model uncertainties.

4.3 Distributed linear quadratic optimal control design with guaranteed stability

In this section, we present a solution to problem 4.1, that is we aim to find a structured LQ-optimal feedback law which is stabilizing and in accordance with Definition 4.1.

Before we proceed, we introduce the finite horizon version of the optimization problem (4.3):

$$\min_{x,u} J(t,x,u) = x^{\mathsf{T}}(t+t_{\mathrm{f}})Sx(t+t_{\mathrm{f}}) + \int_{t}^{t+t_{\mathrm{f}}} x^{\mathsf{T}}(\tau)Qx(\tau) + u^{\mathsf{T}}(\tau)Ru(\tau)\mathrm{d}\tau, \tag{4.4a}$$

$$\text{s.t. } \dot{x}(\tau) = Ax(\tau) + Bu(\tau),\ x(t) = x_t \tag{4.4b}$$

$$u(\tau) = -K_{\mathrm{dist}}x(\tau), \tag{4.4c}$$

$$K_{\mathrm{dist}} \text{ is stabilizing}, \tag{4.4d}$$

$$K_{\mathrm{dist}} \in \mathscr{K}. \tag{4.4e}$$

We will later see that through an appropriate choice of the matrix $S \in \mathbb{S}_{+}^{n}$, the finite horizon cost (4.4a) serves as an upper bound to the infinite horizon cost (4.3a). Therefore, solving problem (4.4) simultaneously leads to an optimized – but admittedly possibly suboptimal – solution to the original problem (4.3). However, because the presented approach is only applicable to finite horizon cost functionals, as will be seen later, we have to accept this possible suboptimality. Naturally, with increasing horizon, the result converges to the infinite

horizon result. Clearly, considering the finite horizon leads to the question of how stability can be guaranteed, which will also be addressed. Note that since we do not consider any constraints on the input or state, the optimization problems (4.3) and (4.4) are always feasible, but because K_{dist} is the optimization variable, both problems are non-convex.

To provide an overview of the approach, the stabilizing optimal control design involves the following main steps:

1. Consider the finite horizon problem (4.4) instead of the infinite horizon problem (4.3).

2. Determine a suitable terminal cost term $x^{\mathsf{T}}(t+t_{\text{f}})Sx(t+t_{\text{f}})$ that guarantees stability. This is addressed in Section 4.3.3.

3. Find the distributed control law K_{dist} that minimizes (4.3) given the stabilizing terminal cost matrix S. This is described in detail in Section 4.3.1.

As will be seen, the presented distributed control design method makes use of simulated trajectories. This is the reason why the original optimization problem (4.3) needs to be reformulated to the finite horizon equivalent in (4.4), which corresponds to Step 1 because trajectories can only be simulated for a finite horizon. Step 2 is required because optimality with respect to a finite horizon does not itself guarantee stability.

The presentation of the overall solution proceeds in the subsequent subsections as follows: First, we present an iterative approach of solving (4.4) while ignoring the two last constraints (4.4d) and (4.4e). Then, we introduce additional assumptions that allow the solution satisfying the constraint (4.4e). Last, we present a distributed approach that allows us to guarantee stability and with that satisfy condition (4.4d).

Remark 4.1. In the centralized case, we know that the optimal control law for the finite horizon is time-varying while we assume a time-invariant control law in the present chapter. For more insight to this and a performance comparison, see Section 4.8.

4.3.1 Distributed control design using gradient descent

In this section, we look at the distributed solution of problem (4.4) using only information exchange with neighbors while ignoring the stability constraint (4.4d). In the first step, we also ignore the constraint (4.4e). The resulting algorithm of this section is the basis for the design of the control law with guaranteed stability.

Remark 4.2. For the complete distributed stabilizing design, we make several additional assumptions along the way in this section. We structure the following text in such a way that all the results up to each assumption are valid but may not allow a completely distributed computation and still contain centralized aspects. After we state the final Assumption 4.5, the design is completely distributed. In other words, we reduce the centralization of the computation step by step with every additional assumption.

By considering the Lagrangian of problem (4.4) and using optimality conditions, the following proposition is derived.

Proposition 4.1. *Given optimization problem (4.4), the gradient of the cost functional (4.4a) with respect to the control law blocks $K_{\text{dist},ij}$ is given by*

$$(\nabla_{K_{\text{dist}}} J)_{ij} = \int_t^{t+t_\text{f}} -2R_i u_i x_j^\mathsf{T} + B_i^\mathsf{T} \lambda_i x_j^\mathsf{T} \,\mathrm{d}\tau, \tag{4.5}$$

where

$$\dot{\lambda}(\tau) = -A_{K_{dist}}^\mathsf{T} \lambda(\tau) + 2Q_{K_{\text{dist}}} x(\tau), \tag{4.6a}$$
$$\lambda(t+t_f) = -2Sx(t+t_\text{f}),\ \tau \in [t, t+t_\text{f}], \tag{4.6b}$$

with $A_{K_{\text{dist}}} = A - BK_{\text{dist}}$ and $Q_{K_{\text{dist}}} = (Q + K_{\text{dist}}^\mathsf{T} R K_{\text{dist}})$.

Proof. The Lagrangian of the optimization problem is given by

$$L = x^\mathsf{T}(t+t_\text{f}) S x(t+t_\text{f}) + \int_t^{t+t_\text{f}} \Big[x^\mathsf{T}(\tau)(Q + K_{\text{dist}}^\mathsf{T} R K_{\text{dist}}) x(\tau) + \lambda^\mathsf{T}(\tau)(\dot{x}(\tau) - A_{K_{\text{dist}}} x(\tau)) \Big] \mathrm{d}\tau + \mu(x(t) - x_t).$$

Partial integration gives

$$L = x^\mathsf{T}(t+t_\text{f}) S x(t+t_\text{f}) + \int_t^{t+t_\text{f}} \Big[x^\mathsf{T}(\tau)(Q + K_{\text{dist}}^\mathsf{T} R K_{\text{dist}}) x(\tau) + x^\mathsf{T}(\tau)(-\dot{\lambda}(\tau) - A_{K_{\text{dist}}}^\mathsf{T} \lambda(\tau)) \Big] \mathrm{d}\tau + \mu(x(t) - x_t) + (\lambda^\mathsf{T}(\tau) x(\tau)|_t^{t+t_\text{f}}.$$

The optimality condition $\frac{\partial L}{\partial x} = 0$ gives the dynamics of the adjoint state λ as stated in (4.6). Similarly, the gradient is given by computing the derivative $\frac{\partial L}{\partial K_{\text{dist}}}$ and leads to the formula stated in (4.5). □

Using the search direction from Proposition 4.1, Algorithm 7 is used to find a finite horizon optimal control law according to the cost functional (4.4a).

The only requirement for the initializing feedback $K_{\text{dist}}^{(0)}$ is that it satisfies the allowed structure according to Definition 4.1 (2). An obvious choice would be the zero matrix of appropriate size.

Note that, in principle, Algorithm 7 can be used for a centralized solution to problem 4.4 when the constraints (4.4d) and (4.4e) are ignored to obtain a control law with a desired sparsity structure. In order to incorporate constraint (4.4e) and enable a solution in accordance with Definition 4.1, however, we make the following assumptions.

Assumption 4.1. The computation graph $\mathcal{G}_{\text{comp}}$ contains the undirected system graph $\mathcal{G}_{\text{s,u}}$ as a subgraph, and the computation graph $\mathcal{G}_{\text{comp}}$ contains the control graph $\mathcal{G}_{\text{control}}$ as a – possibly different – subgraph, i.e. $\mathcal{G}_{\text{s,u}}(\mathcal{V}_\text{s}, \mathcal{E}_{\text{s,u}}) \subseteq \mathcal{G}_{\text{comp}}(\mathcal{V}_\text{s}, \mathcal{E}_{\text{comp}})$ and $\mathcal{G}_{\text{control}}(\mathcal{V}_\text{s}, \mathcal{E}_{\text{control}}) \subseteq \mathcal{G}_{\text{comp}}(\mathcal{V}_\text{s}, \mathcal{E}_{\text{comp}})$.

Algorithm 7 Gradient descent algorithm for the solution of (4.4).

Given K_{dist}^0 and $x(t)$, do the following steps.

1. Simulate the states $x_i(\cdot)$ of system (4.2) for the horizon t_f with the given initial condition $x(t)$.
2. Simulate the adjoint states $\lambda_i(\cdot)$ from (4.6) for the same horizon t_f.
3. Every agent calculates its respective entries of the gradient given by
$$(\nabla_{K_{\text{dist}}} J)_{ij} = \sum_{l=1}^{n} \int_t^{t+t_\text{f}} -2R_i u_{i,l} x_{j,l}^\mathsf{T} + B_i^\mathsf{T} \lambda_{i,l} x_{j,l}^\mathsf{T} \mathrm{d}\tau.$$
4. For each neighboring agent j, update
$$K_{\text{dist},ij}^{(k+1)} = K_{\text{dist},ij}^{(k)} - \gamma_k (\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}$$
with a step length γ_k.
5. If a stopping criterion, e.g. $||(\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}|| < \epsilon$, is satisfied, stop. Otherwise, increase k and go back to 1.

This assumption is reasonable and not very restrictive. It follows from the assumption that subsystems that are physically interconnected, described by the given system graph $\mathcal{G}_\text{s}$, regardless of the direction, can exchange information during the control design. However, systems that are not connected may also exchange information. Furthermore, systems communicating during the design phase may communicate online to compute their input signal through the control law, but they do not have to.

Assumption 4.2. The weighting matrix $Q \in \mathbb{S}_+^n$ has at most the block sparsity structure according to the computation graph $\mathcal{G}_\text{comp}$, i.e. $Q_{ij} = 0$ if $(i,j) \notin \mathcal{E}_\text{comp}$. The weighting matrix $R \in \mathbb{S}_{++}^n$ is block diagonal. The block sizes result from the subsystem dimensions.

The assumption is not very restrictive because in most cases in practice, diagonal matrices are used. The structure allows cost terms that are completely local to each subsystem, using the main diagonal of Q, as well as coupled costs with neighbors according to the computation graph. In other words, subsystems need to be able to communicate during the design phase when they share a cost. Note that this contains the case that subsystems are not physically connected at all, e.g. in a formation control problem of unconnected vehicles. Then, the system graph is unconnected, but the computation graph structure can be used to express the formation goals. This assumption is necessary in order to enable the distribution of the control design because the weighting matrix structure plays a role in the neighborhood structure of the adjoint states, as can be seen in (4.6).

This leads to the following proposition.

Proposition 4.2. *Given system (4.2) and Assumptions 4.1 and 4.2, Algorithm 7 results in a distributed feedback law satisfying Definition 4.1, that is* $K_{\text{dist}} \in \mathscr{K}$.

Proof. In order to simulate the closed-loop state trajectories in Step 1 and the adjoint state trajectories in Step 2, agent i only needs to exchange information with neighbors according to the computation graph $\mathcal{G}_{\text{comp}}$ by Assumption 4.2. Furthermore, to compute the gradient in Step 3, each agent again needs those trajectories of neighbors j that it plans to use with a non-zero entry $K_{\text{dist},ij}$ in its feedback law. By Definition 4.1 and Assumption 4.1, these are exactly neighbors according to the control graph $\mathcal{G}_{\text{control}} \subseteq \mathcal{G}_{\text{comp}}$. □

For an illustration of the idea of the approach so far, see Figure 4.1.

In the next subsection, an improvement of Algorithm 7 is presented, and in the subsection one after the next, we will see how we can determine a terminal cost term that guarantees stability, and how we can obtain it in a distributed fashion according to Definition 4.1.

4.3.2 Averaging approach to eliminate dependence on initial condition

In the previous subsection, we presented an iterative and distributed control design approach that is optimal with respect to a finite horizon LQ cost functional. The key idea behind our approach to enable the distribution is to use simulated trajectories to compute a gradient.

One inherent problem of Algorithm 7 is that it depends on two independent initial conditions. The first one is $K_{\text{dist}}^{(0)}$, i.e. the feedback matrix to start the iterative process which is the actual decision variable. Second, in order to start the state and co-state simulations, an initial condition of the state $x(t)$ for the starting time t needs to be chosen. Thus, the result of the control law design depends on this choice of the state initial condition which is unrelated to the actual decision variables, namely the entries of the feedback matrix. This second initial condition is unwanted and we aim to gain independence of it for two reasons. First, we would like to find the optimal feedback matrix independent of a specific state initial

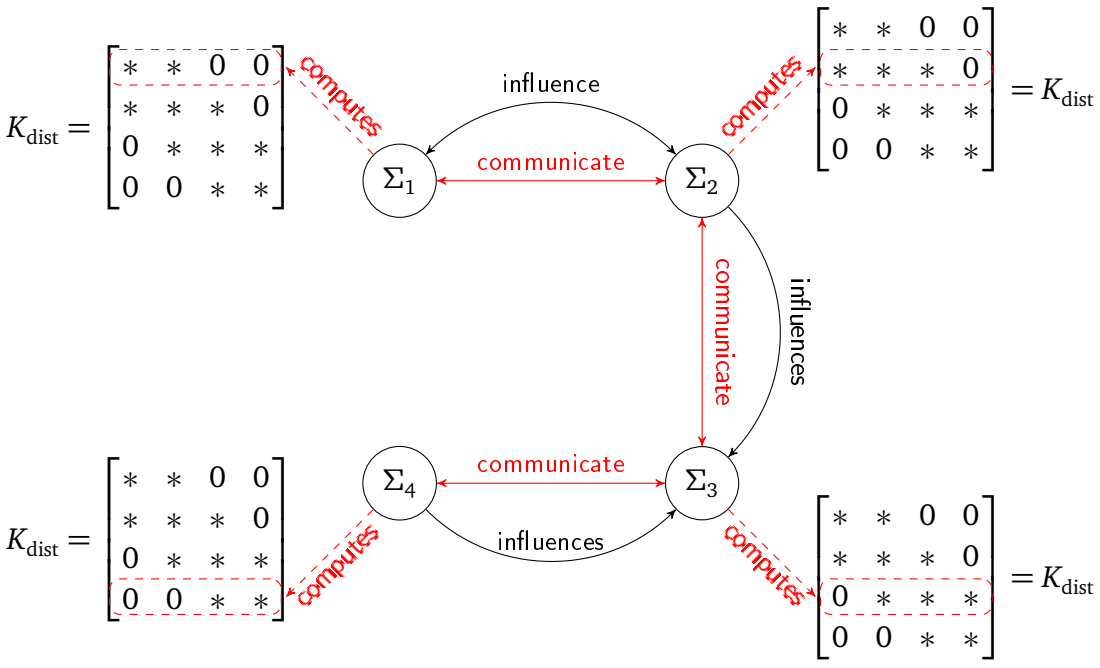


Figure 4.1: Illustration of the distributed control design approach.

condition because we generally do not know the actual initial condition of the process in advance during the controller design, and because it might be quite different from the one used in the design algorithm. Second, a systematic approach to pick this one specific initial condition is not obvious.

Another, more model related point can be made. Since the agents are confined to models of their own dynamics and have no global model, all the information necessary for the optimal controller design has to be extracted from the simulation data. This makes it important that all states are sufficiently excited using the initial condition of the state trajectory. By using just one specific, fixed initial condition $x(t)$, as in the previous subsection, only a limited direction of the system behavior could be excited, or the coupling structure of the system could prohibit the spreading of the signal. Imagine that $x(t)$ happens to be only a unit base vector, such that only one state of one agent is excited by the initial condition. If the system is large, nodes that do not have a direct coupling to the excited node and are relatively far away will be likely to get very little information about the system dynamics. Thus, it will not be able to determine an appropriate controller for every possible excitation of the system. Naturally, the resulting controller will work well for the specific initial condition, but this is usually not desired when designing a feedback controller.

In order to overcome this dependence, and to ensure sufficient dynamical excitation of the system, we propose an averaging approach. Therefore, we make the following assumption about the initial condition of the state $x(t)$. Note that this assumption is only used for the initial condition of the simulated trajectories in the design process. This is not related to the actual initial condition of the online process and, therefore, does not change the original problem.

Assumption 4.3. The initial condition $x(t)$ is a random variable, uniformly distributed on the surface of the n-dimensional unit sphere with expected value $\mathrm{E}[x(t)x^\mathsf{T}(t)] = \frac{1}{n} I_n$.

The cost functional (4.4a) is adjusted to the following

$$J(x,u) = \mathrm{E}\left[x^\mathsf{T}(t+t_\mathrm{f}) S x(t+t_\mathrm{f}) + \int_t^{t+t_\mathrm{f}} x^\mathsf{T}(\tau) Q x(\tau) + u^\mathsf{T}(\tau) R u(\tau) \mathrm{d}\tau \right],$$

where E represents the expected value with respect to the initial condition $x(t)$. Thus, the cost becomes independent of $x(t)$. Using the solutions for $x(\cdot), \lambda(\cdot)$, it can be shown that all terms in the gradients (4.5) are linear in the term $x(t)x^\mathsf{T}(t)$. To achieve this, we need to show that $\lambda(\cdot)$ depends linearly on $x(t)$. The solution for $\lambda(\cdot)$ can be given as

$$\lambda(t') = -2 e^{A_{K_\mathrm{dist}}^\mathsf{T}(t_\mathrm{f}+t-t')} S x(t+t_\mathrm{f}) + 2 \int_{t'}^{t+t_\mathrm{f}} e^{A_{K_\mathrm{dist}}^\mathsf{T}(\tau-t')} Q_{K_\mathrm{dist}} x(\tau) \mathrm{d}\tau,$$

where $t' \in (t, t+t_\mathrm{f})$. Clearly, $x(\tau)$ and $x(t+t_\mathrm{f})$ depend linearly on $x(t)$, and hence so does $\lambda(t')$. Therefore, all products in the gradient are linear in $x(t)x^\mathsf{T}(t)$.

Because of the linearity and in order to average over unit initial conditions, we define $x_l(t), \lambda_l(t)$ as the simulated trajectories based on the initial condition e_l, where e_l is the lth unit base vector. With this, we can adjust Algorithm 7 to this averaging approach and arrive at Algorithm 8.

Algorithm 8 Gradient descent algorithm for the solution of (4.4) including initial condition averaging.

Given K_{dist}^0, do the following steps.

1. Simulate the states $x_{i,l}(\cdot)$ of system (4.2) n times for the horizon t_f where the state initial condition of the lth simulation is the unit base vector e_l with $l = 1, ..., n$.
2. Simulate the corresponding adjoint states $\lambda_{i,l}(\cdot)$ from (4.6) for the same horizon t_f.
3. Every agent calculates its respective entries of the gradient given by
$$(\nabla_{K_{\text{dist}}} J)_{ij} = \frac{1}{n}\left(\sum_{l=1}^{n} \int_{t}^{t+t_\text{f}} -2R_i u_{i,l} x_{j,l}^\mathsf{T} + B_i^\mathsf{T} \lambda_{i,l} x_{j,l}^\mathsf{T} \mathrm{d}\tau \right).$$
4. For each neighboring agent j, update
$$K_{\text{dist},ij}^{(k+1)} = K_{\text{dist},ij}^{(k)} - \gamma_k (\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}$$
with a step length γ_k.
5. If a stopping criterion, e.g. $||(\nabla_{K_{\text{dist}}} J)_{ij}^{(k)}|| < \epsilon$, is satisfied, stop. Otherwise, increase k and go back to 1.

Using Algorithm 8, the resulting controller is independent of the initial condition of the state. The agents need to have knowledge about the total number of states n in the system in order to know how many simulations they have to run, but this is still in accordance with Definition 4.1. Furthermore, a protocol is required that determines which unit base vector is used at what time as the initial condition.

We see that making use of Assumption 4.3 ensures the point we made earlier because it leads to the simulation of the system with every unit base vector as an initial condition in the algorithm, thus exciting every state. Thus, the resulting controller is optimal given the maximum possible amount of information about the system dynamics without actually knowing the system model. It is important to note that this does not help to overcome the non-convexity of the problem with respect to the controller parameters.

In the next section, we address the problem of guaranteeing stability in this distributed setting with local model information.

4.3.3 Distributed computation of stability guaranteeing terminal cost term

In this subsection, we describe how a terminal cost weighting matrix is used to guarantee stability. Next, it is shown how this matrix can be determined using an optimization problem in a similar fashion to the stability analysis problem in Section 3.4.2. Last, a distributed approach is presented to solve the optimization problem that requires only local information exchange among neighboring subsystems according to the computation graph $\mathcal{G}_{\text{comp}}$.

4.3.3.1 Stability from terminal cost term

The idea of using a terminal cost term to guarantee stability stems from model predictive control (MPC). We shortly recapitulate the main principles of MPC.

The core idea of MPC is to repeatedly solve an optimal control problem over a finite time horizon t_f based on current state measurements to obtain an optimal open-loop input trajectory $u^*(\cdot)$. Only the first part of the input trajectory is applied to the system for the duration of the sampling period Δt and then the optimization problem is solved again with new state measurements. The problem that finite horizon optimality does not inherently guarantee stability can be circumvented by using a terminal cost term which represents a bound on the infinite horizon cost functional [101]. In this thesis, we adapt this approach of using a terminal cost term to a distributed feedback control law instead of an open-loop trajectory as is used in MPC. More importantly, we place our focus on the distributed computation of the desired terminal term in the subsequent subsections which, in contrast, is typically handled by a centralized solution of a Lyapunov equation.

The starting point to obtain the terminal cost term is a stabilizing, though not necessarily optimized, auxiliary control law K_aux. Given K_aux, we compute a solution (P,δ) to the linear matrix inequality (LMI)

$$F_{K_\mathrm{aux}}(P,\delta) := A_{K_\mathrm{aux}}^\mathsf{T} P + P A_{K_\mathrm{aux}} + \gamma\delta \underbrace{(Q + K_\mathrm{aux}^\mathsf{T} R K_\mathrm{aux})}_{Q_{K_\mathrm{aux}}} \preceq 0, \tag{4.7}$$

where $A_{K_\mathrm{aux}} = A - BK_\mathrm{aux}$, $\gamma > 0$ is a pre-specified constant, and the solution variables are $P \in \mathbb{S}^n$ and $\delta \in \mathbb{R}$. The reason for using a Lyapunov inequality is that it links stability with the cost functional, as will be seen in the following.

Observe that K_aux is stabilizing if the resulting $P \in \mathbb{S}^n_{++}$ and if $\delta > 0$ [20]. If not, K_aux needs to be re-designed. Note that the auxiliary feedback K_aux is never actually applied but merely required for the stability proof. The only feedback law that is actually applied is the distributed control law K_dist. Given a valid solution (P,δ), we state the following lemma.

Lemma 4.1. *Given a solution (P,δ) to the LMI (4.7) and a stabilizing feedback law $u(\tau) = -K_\mathrm{aux}x(\tau)$, the infinite horizon cost functional $J_\infty(t,x,u)$ is bounded from above by the terminal cost term as*

$$x(t)^\mathsf{T} S x(t) \geq \int_t^\infty x^\mathsf{T}(\tau) Q x(\tau) + u^\mathsf{T}(\tau) R u(\tau) d\tau, \tag{4.8}$$

where $S = \frac{P}{\gamma\delta}$.

Proof. Differentiation of $x^\mathsf{T} S x$ along a trajectory of system (4.2) and using (4.7) gives

$$\dot{V} = \frac{\mathrm{d}}{\mathrm{d}t} x^\mathsf{T} S x = x^\mathsf{T}(A_{K_\mathrm{aux}}^\mathsf{T} S + S A_{K_\mathrm{aux}}) x \leq -x^\mathsf{T} Q_{K_\mathrm{aux}} x < 0. \tag{4.9}$$

Integrating both sides from t to ∞, and knowing that K_aux is stabilizing gives the result from (4.8). □

Next, it is shown that the distributed control law K_dist from optimization problem (4.4) with the terminal cost term with $S = \frac{P}{\delta\gamma}$ yields a value function that is non-increasing. This is stated in the following lemma.

Lemma 4.2. *For $\tau \in (t, t+\Delta t]$ the optimal value function satisfies*

$$J^*(x(\tau), \tau, \tau + t_\mathrm{f}) \leq J^*(x(t), t, t + t_\mathrm{f}) - \int_t^\tau x^\mathsf{T}(s) Q x(s) + u^\mathsf{T}(s) R u(s) ds.$$

Proof. The following notation is used: $\bar{x}(\tau; x(t), t)$ is the prediction of the state trajectory at time τ using state information from time t, and $\bar{x}^*(\tau; x(t), t, t + t_\mathrm{f})$ is the optimal predicted state trajectory at time τ using state information from time t with optimization horizon t_f.

At time t, the optimal feedback input $\bar{u}^*(\cdot; x(t), t, t + t_\mathrm{f})$ is computed as $\bar{u}^* = -K_\mathrm{dist}\bar{x}^*$ and we additionally have the optimal predicted state trajectory $\bar{x}^*(\cdot; x(t), t, t + t_\mathrm{f})$ on $[t, t + t_\mathrm{f}]$. Then, the value of the optimal value function is

$$\begin{aligned} J^*(x(t), t, t + t_\mathrm{f}) = \int_t^{t+t_\mathrm{f}} & \bar{x}^{*^\mathsf{T}}(s; x(t), t, t + t_\mathrm{f}) Q \bar{x}^*(s; x(t), t, t + t_\mathrm{f}) \\ & + \bar{u}^{*^\mathsf{T}}(s; x(t), t, t + t_\mathrm{f}) R \bar{u}^*(s; x(t), t, t + t_\mathrm{f}) \mathrm{d}s \\ & + \bar{x}^{*^\mathsf{T}}(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f}) S \bar{x}^*(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f}). \end{aligned} \tag{4.10}$$

For τ in the current interval $(t, t + \Delta t]$, the control input is determined by K_dist and the state trajectory is identical to the predicted trajectory, i.e. $x(s) = \bar{x}^*(s; x(t), t, t + t_\mathrm{f})$ for any $s \in [t, \tau]$. Assuming the suboptimal feedback

$$u(\tau) = \begin{cases} -K_\mathrm{dist} x(\tau), & \text{if } t \leq \tau \leq t + t_\mathrm{f} \\ -K_\mathrm{aux} x(\tau), & \text{if } \tau > t + t_\mathrm{f}, \end{cases} \tag{4.11}$$

is applied to the system, the resulting trajectories are identical to the ones from optimization time t, except for the shifted part in the time interval $[t + t_\mathrm{f}, \tau + t_\mathrm{f}]$. Thus, we have that $\bar{x}(s; x(\tau), \tau) = \bar{x}^*(s; x(t), t, t + t_\mathrm{f})$ for all $s \in [\tau, t + t_\mathrm{f}]$. To determine the value of the cost function for $\tau \in (t, t+\Delta t]$, we compute the cost generated in the new part of the optimization interval, namely $[t + t_\mathrm{f}, \tau + t_\mathrm{f}]$. With the chosen input (4.11), (4.9) is satisfied within that interval. Integration of (4.9) in the interval of interest $[t + t_\mathrm{f}, \tau + t_\mathrm{f}]$ gives

$$\begin{aligned} & \bar{x}(\tau + t_\mathrm{f}; x(\tau), \tau)^\mathsf{T} S \bar{x}(\tau + t_\mathrm{f}; x(\tau), \tau) + \int_{t+t_\mathrm{f}}^{\tau+t_\mathrm{f}} \bar{x}(s; x(\tau), \tau)^\mathsf{T} Q_{K_\mathrm{aux}} \bar{x}(s; x(\tau), \tau) \mathrm{d}s \\ & \leq \bar{x}^*(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f})^\mathsf{T} S \bar{x}^*(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f}). \end{aligned}$$

With this, the value of the cost functional for $\tau \in (t, t + \Delta t]$ can be bounded as

$$\begin{aligned} \bar{J}(x(\tau), \tau, \tau + t_\mathrm{f}) \leq \int_\tau^{t+t_\mathrm{f}} & \bar{x}^*(s; x(t), t, t + t_\mathrm{f})^\mathsf{T} Q_{K_\mathrm{dist}} \bar{x}^*(s; x(t), t, t + t_\mathrm{f}) \mathrm{d}s \\ & + \bar{x}^*(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f})^\mathsf{T} S \bar{x}^*(t + t_\mathrm{f}; x(t), t, t + t_\mathrm{f}), \end{aligned}$$

where $Q_{K_\mathrm{dist}} = Q + K_\mathrm{dist}^\mathsf{T} R K_\mathrm{dist}$. Combining this with (4.10), and knowing that J^* is optimal, we obtain for $\tau \in (t, t + \Delta t]$

$$\begin{aligned} J^*(x(\tau), \tau, \tau + t_\mathrm{f}) & \leq \bar{J}(x(\tau), \tau, \tau + t_\mathrm{f}) \\ & \leq J^*(x(t), t, t + t_\mathrm{f}) - \int_t^\tau x(s)^\mathsf{T} Q_{K_\mathrm{dist}} x(s) \mathrm{d}s. \end{aligned}$$

With $Q \succeq 0$ and $R \succ 0$, we conclude that the value function is always non-increasing. □

With the help of Lemmas 4.1 and 4.2, we can state the following lemma about the stability of an MPC implementation of the input $u(t) = -K_{\text{dist}}x(t)$ resulting from optimization problem (4.4). By MPC implementation, we understand that the control law is applied for the sampling period Δt after which the optimization problem (4.4) is solved again and this process is repeated indefinitely.

Lemma 4.3. *The closed loop of system (4.1),(4.2) with an MPC implementation of the input $u(t) = -K_{\text{dist}}x(t)$ resulting from optimization problem (4.4) with $S = \frac{P}{\gamma\delta}$ from (4.7) is asymptotically stable.*

Proof. We define $V(x) = J^*(x, t, t + t_f)$. This function has the following properties:

- $V(0) = 0$, $V(x) > 0$ for $x \neq 0$,
- along the trajectory of the closed-loop system, there is for $0 \leq t_1 < t_2 \leq \infty$

$$V(x(t_2)) - V(x(t_1)) \leq -\int_{t_1}^{t_2} x^{\mathsf{T}}(t)Qx(t)\mathrm{d}t, \tag{4.12}$$

where Lemma 4.2 is used. We now show asymptotic stability, i.e. for every $\epsilon > 0$, there is $\eta(\epsilon) > 0$ such that $||x(0)|| < \eta(\epsilon)$ implies $||x(t)|| < \epsilon$ for all $t \geq 0$ [30]. From (4.12), we obtain

$$V(x(\infty)) \leq V(x(0)) - \int_0^{\infty} x^{\mathsf{T}}(t)Qx(t)\mathrm{d}t.$$

Because $V(x(\infty)) \geq 0$ and $V(x(0)) \leq \beta > 0$, the integral must exist and it is bounded. Clearly, the closed-loop system given by (2.9) with feedback $u = -K_{\text{dist}}x$ is uniformly continuous. Hence, we can apply Barbalat's lemma and conclude that $||x(t)|| \to 0$ as $t \to \infty$, which implies asymptotic stability. □

Remark 4.3. Note that the proofs of the Lemmas in this subsection follow [101] but are adapted to our needs and to the case of a distributed feedback law instead of an open loop input trajectory.

Remark 4.4. The choice of K_{aux} is not unique and it has an influence on the solution of the LMI (4.7) and effectively on the resulting terminal cost term. Thus, it also has an effect on the resulting K_{dist}. Unfortunately, no general guidelines can be stated, but using an optimized control law, e.g. based on LQR, is a reasonable approach.

We have now shown that the control law resulting from Algorithm 8 is stabilizing, if a suitable terminal cost term according to the LMI (4.7) is used. However, in order for the design method to satisfy Definition 4.1, the computation of the matrix S also needs to be distributed. This is addressed in the next subsection.

4.3.3.2 Distributed computation of terminal cost matrix using distributed optimization

Given the terminal cost term from the previous subsection, we know that the solution to problem (4.4) is stabilizing. In this subsection, we determine this terminal cost term $S = \frac{P}{\gamma\delta}$ in accordance with Definition 4.1 using distributed optimization. The goal of this subsection is to find a solution to the LMI (4.7). The approach to achieve this is based on the distributed stability test in Section 3.4.2 with slight adaptation.

To facilitate a distributed solution of the LMI (4.7) we make the following assumption concerning its block sparsity structure.

Assumption 4.4. The LMI (4.7) has a block sparsity structure in accordance with the computation graph $\mathcal{G}_{\text{comp}}$, that is the block $(F_{K_{\text{aux}}}(P,\delta))_{ij} = 0$ iff $(i,j) \notin \mathcal{E}_{\text{comp}}$.

The critical terms of (4.7) to satisfy the assumption are $A^\mathsf{T}P + PA$ and $K_{\text{aux}}^\mathsf{T} R_{\text{aux}} K$ as Q automatically satisfies the required property by the previous Assumption 4.2. Let us first consider the simplest case where $\mathcal{G}_{\text{comp}} = \mathcal{G}_{\text{s,u}}$. Then, P is a block diagonal matrix where the block sizes correspond to the subsystem sizes and K_{aux} needs to have a form such that $K_{\text{aux}}^\mathsf{T} R K_{\text{aux}}$ satisfies the assumption. One way to achieve this is that K_{aux} is a completely decentralized control law, hence block diagonal. However, there are other possible structures for K_{aux} to satisfy the requirement depending on the allowed graphical structure. In this first case, where $\mathcal{G}_{\text{comp}} = \mathcal{G}_{\text{s,u}}$, the block structure of $F_{K_{\text{aux}}}(P,\delta)$ corresponds to the structure of $A^{\text{sym}} = A + A^\mathsf{T}$, which is exactly represented by $\mathcal{G}_{\text{s,u}} = \mathcal{G}_{\text{comp}}$. However, in the case in which the computation graph contains more edges than the system graph, i.e. $\mathcal{E}_{\text{comp}} \supset \mathcal{E}_{\text{s,u}}$, P and K_{aux} contain larger blocks corresponding to the merging of several subsystems in accordance with the additional edges. The block size of a larger block then corresponds to the sum of the individual block sizes. To make the presentation of the results simpler, we focus on the first case in the following. With the block diagonal restriction, the LMI considered in the following becomes

$$A_{K_{\text{aux}}}^\mathsf{T} \underbrace{\text{diag}(P^1,\dots,P^N)}_{P} + \text{diag}(P^1,\dots,P^N) A_{K_{\text{aux}}} + \gamma\delta \underbrace{(Q + K_{\text{aux}}^\mathsf{T} R K_{\text{aux}})}_{Q_{K_{\text{aux}}}} \preceq 0. \tag{4.13}$$

The sparsity structure of this LMI is now identical to the original system structure, which is the basis for the distributed solution. In other words, this means that the block sparsity structure of the LMI is described by $G_{\text{s,u}}$ while the element sparsity structure is described by the graph $\mathcal{G}_{\text{LMI}} = (\mathcal{V}_{\text{LMI}}, \mathcal{E}_{\text{LMI}})$ where $(j,i) \in \mathcal{E}_{\text{LMI}}$ iff the element $A_{ij}^{\text{sym}} \neq 0$. Note that in this graph, an edge corresponds to an individual matrix entry, and not to a matrix block. Similarly, throughout this subsection, the notation X_{ij} refers to the (i,j)-element of the matrix X, and not to the block.

Remark 4.5. With the structural restriction from Assumption 4.4, it becomes clear that S also automatically satisfies the same sparsity structure as Q from Assumption 4.2, i.e. $S_{ij} = 0$ if $(i,j) \notin \mathcal{E}_{\text{comp}}$.

To distribute the solution of the LMI, we proceed in the same fashion as in Section 3.4.2 and decompose it by applying the *range-space conversion method* [82]. Therefore, as in the referred section, the following assumption is required.

Assumption 4.5. $\mathcal{G}_{\text{LMI}}$ is a chordal graph.

If G_{LMI} is not a chordal graph we cannot decompose the LMI to solve it distributedly. In that case, the structure of (4.13) needs to be extended by additional edges to its chordal extension, and, with that, effectively enlarging the original $\mathcal{G}_{\text{comp}}$ to satisfy Assumption 4.5. This is achieved by allowing more entries in P, which is equivalent to adding edges to $\mathcal{E}_{\text{comp}}$ that go beyond $\mathcal{E}_{\text{s,u}}$. This extension, however, can be found in polynomial time [82]. In practice, if the chordal extension is required that means that the affected subsystems need to collaborate with additional subsystems than originally expressed in $\mathcal{G}_{\text{comp}}$. Note that Assumption 4.5 cannot be expressed in terms of $\mathcal{G}_{\text{s,u}}$ or $\mathcal{G}_{\text{comp}}$ because the element structure is relevant.

Again, the overall idea is to obtain a result with only local information by decomposing the LMI (4.13), and then to formulate an optimization problem that contains the decomposed condition as a constraint. Afterwards, we apply the distributed constrained Nesterov algorithm (DCNA).

We use the same definitions as in Section 3.4.2. With them, we define the following matrices for $l = 1, \dots, N$ and $(i, j) \in \mathcal{B}_l$

$$
\begin{aligned}
F^0 &= -\gamma Q_{K_{\text{aux}}}, \\
F_{ij}^l &= \begin{cases} \frac{1}{2}\left(-A_{K_{\text{aux}}}^{\mathsf{T}} E_{ij} - E_{ij} A_{K_{\text{aux}}}\right) & \text{if } i < j, \\ \frac{1}{2}\left(-A_{K_{\text{aux}}}^{\mathsf{T}} E_{ji} - E_{ji} A_{K_{\text{aux}}}\right) & \text{if } i > j, \\ -A_{K_{\text{aux}}}^{\mathsf{T}} E_{ij} - E_{ij} A_{K_{\text{aux}}} & \text{if } i = j. \end{cases}
\end{aligned}
$$

Essentially, the main difference to Section 3.4.2 is that we have replaced the identity matrix with the matrix $Q_{K_{\text{aux}}}$ for F^0. Similarly, in F_{ij}^l, the matrix A is replaced by the closed-loop form $A_{K_{\text{aux}}}$ which includes the pre-determined auxiliary feedback law K_{aux}.

With this and with $i_l = i - \sum_{s=1}^{l-1} n_s$, the LMI (4.13) can be rewritten as

$$
F(P, \delta) := F^0 \delta + \sum_{l=1}^{N} \sum_{(i,j) \in \mathcal{B}_l} F_{ij}^l P_{i_l j_l}^l \succeq 0.
$$

Then, following the same steps taken in Section 3.4.2, we eventually arrive at the distributed optimization problem given by

$$
\begin{aligned}
\min_{\delta \in \mathbb{R}, P^l \in \mathbb{S}^{n_l}} \quad & -\delta + \frac{\sigma_\delta}{2}\delta^2 + \sum_{l=1}^{N} \frac{\sigma_{P^l}}{2} \|P^l\|_F^2 + \sum_{s=1}^{p} \frac{\sigma_{Y^s}}{2} \|Y^s\|_F^2 \\
\text{s.t.} \quad & \delta I_{n_l} - P^l \preceq 0 \text{ for } l = 1, \dots, N, && \text{(4.14a)} \\
& E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^s - E_{ij} \bullet F(P, \delta) = 0 \text{ for } (i, j) \in J, && \text{(4.14b)} \\
& Y^s \in \mathbb{S}_+^{C_s} \text{ for } s = 1, \dots, p, && \text{(4.14c)}
\end{aligned}
$$

which can be solved using Algorithm 9.

Remark 4.6. We refer to Section 3.4.2 for more details on the algorithm's convergence guarantees, closed form solutions to subproblems, its accuracy and the possibility to employ event-based communication during the optimization.

Algorithm 9 Distributed solution of (4.14).

For $k \geq 0$ do

1. Set $M^{l,0} = 0_{n_l}$ and $\Lambda_{ij}^0 = 0_n$.
2. Given $M^{l,k}$ and components Λ_{ij}^k, the agents compute

$$\begin{aligned}
\delta^{k+1} &= \arg\min_{\delta \in \mathbb{R}} \left\{ -x_\delta \delta + \frac{\sigma_\delta}{2} \delta^2 \right\}, \\
P^{l,k+1} &= \arg\min_{P^l \in \mathbb{S}^{n_l}} \left\{ -X_P^l \bullet P^l + \frac{\sigma_{P^l}}{2} \|P^l\|_F^2 \right\}, \\
Y^{s,k+1} &= \arg\min_{Y^s \in \mathbb{S}_+^{C_s}} \left\{ -X_Y^s \bullet Y^s + \frac{\sigma_{Y^s}}{2} \|Y^s\|_F^2 \right\},
\end{aligned}$$

   for $l = 1, \ldots, N$ and $s = 1, \ldots, p$.

For $(i,j) \in J$ and $l = 1, \ldots, N$, the agents do in parallel

2. Given δ^{k+1}, $P^{l,k+1}$, and $Y^{s,k+1}$ compute

$$\begin{aligned}
\nabla_{\Lambda_{ij}} f(\Lambda^k, M^k) &= E_{ij} \bullet \sum_{s \in \Gamma(i,j)} Y^{s,k+1} - E_{ij} \bullet \left(F^0 \delta^{k+1} + \sum_{l=1}^{N} \sum_{(i,j) \in \mathscr{I}_l} F_{ij}^l P_{i_l j_l}^{l,k+1} \right), \\
\nabla_{M^l} f(\Lambda^k, M^k) &= \delta^{k+1} I_{n_l} - P^{l,k+1}.
\end{aligned}$$

3. Find

$$\begin{aligned}
Y_{ij}^k &= \arg\max_{Y_{ij} \in \mathbb{R}} \left\{ -\frac{L}{2} \left(Y_{ij} - \Lambda_{ij}^k \right)^2 + \nabla_{\Lambda_{ij}} f(\Lambda^k, M^k) Y_{ij} \right\}, \\
H^{l,k} &= \arg\max_{H^l \in \mathbb{S}_+^{n_l}} \left\{ -\frac{L}{2} \|H^l - M^{l,k}\|_F^2 + \nabla_{M^l} f(\Lambda^k, M^k) \bullet H^l \right\}.
\end{aligned}$$

4. Find

$$\begin{aligned}
Z_{ij}^k &= \arg\max_{Z_{ij} \in \mathbb{R}} \left\{ -\frac{L}{2} Z_{ij}^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{\Lambda_{ij}} f(\Lambda^j, M^j) Z_{ij} \right\}, \\
T^{l,k} &= \arg\max_{T^l \in \mathbb{S}_+^{n_l}} \left\{ -\frac{L}{2} \|T^l\|_F^2 + \sum_{j=0}^{k} \frac{j+1}{2} \nabla_{M^l} f(\Lambda^j, M^j) \bullet T_l \right\}.
\end{aligned}$$

5. Set

$$\begin{aligned}
\Lambda_{ij}^{k+1} &= \frac{k+1}{k+3} Y_{ij}^k + \frac{2}{k+3} Z_{ij}^k, \\
M^{l,k+1} &= \frac{k+1}{k+3} H^{l,k} + \frac{2}{k+3} T^{l,k}.
\end{aligned}$$

6. Increase k and go back to step 2.

With this algorithm, we can state the following theorem and the main result of this section.

Theorem 4.1. *Given Assumptions 4.1-4.5, the control design method consisting of Algorithms 8 and 9 give an LQ-optimized static feedback law K_{dist} solving problem (4.4) which is stabilizing for system (4.1),(4.2) and which is a feasible control law according to Definition 4.1.*

Proof. All steps of Algorithm 9 can be implemented using only information from direct neighbors in $\mathcal{G}_{\text{comp}}$. The critical, non-obvious step is the computation of δ^{k+1}. However, because the subsystems know the overall system size N, this can be realized using a consensus algorithm. This leads to a stabilizing cost term S using Lemma 4.3 which results in stability of the MPC implementation of K_{dist}. However, Algorithm 8 is independent of any current state information such that a repeated solution of optimization problem (4.4) always gives the same solution. Hence, an MPC implementation is identical to a static feedback law that can be continuously applied. □

Note that the resulting control law is technically not an MPC method because no repeated optimization is used but a static feedback law. MPC arguments are, however, used in order to guarantee stability for a finite horizon cost functional.

In conclusion, the combination of Algorithms 8 and 9 allow the design of a distributed control law using only local model information from neighbors according to the computation graph, and the control structure satisfies the structure of the control graph.

Remark 4.7. We have stated that Algorithm 9 gives only an approximate result. In order to guarantee stability from a theoretical standpoint, however, we do in fact need the exact result. Nonetheless, as is shown in Section 3.4.2, we can obtain a value that is arbitrarily close to the actual result. Future work will explore whether it is possible to guarantee stability given a certain approximation error.

4.3.4 Numerical results

In this subsection, we illustrate and validate the results from this section. We start with a simple example illustrating the structure and main steps of the solution of the problem. Afterwards, we show in a numerical fashion that the presented control design approach is indeed stabilizing. Last, the approach is applied to an academic power system example model.

4.3.4.1 Illustrating example

In this subsection, we use a small-scale example to illustrate the main overall steps of the two algorithms. The topology that we consider is the one in Figure 4.1, which corresponds to a line graph with $N = 4$ agents. This is in fact the smallest system where it makes sense to use the presented method. For three agents, there is always one subsystem that knows the overall system. For illustrative purposes, we assume that each subsystem has $n_i = 1$ states, and the system dynamics are created randomly. The considered state matrix is then

$$A = \begin{bmatrix} 1.35 & 0.86 & 0 & 0 \\ -1.76 & 0.12 & -0.18 & 0 \\ 0 & 0.74 & -1.21 & 1.24 \\ 0 & 0 & -0.25 & -1.06 \end{bmatrix}.$$

The input matrix is $B = \text{diag}(-1.07, 1.44, 2.91, -0.47)$. The weighting matrices Q and R are set to I_4. Two of the eigenvalues have positive real part so the system is unstable. Because of the scalar subsystems, the graph $\mathcal{G}_{\text{LMI}}$ coincides with the computation graph $\mathcal{G}_{\text{comp}} = \mathcal{G}_{\text{control}}$ and they are chordal. The resulting graph has $p = 3$ maximal cliques with $C_1 = \{1,2\}, C_2 = \{2,3\}$ and $C_3 = \{3,4\}$. This leads to the sets $J(C_1) = \{(1,1),(1,2),(2,2)\}, J(C_2) = \{(2,2),(2,3),(3,3)\}, J(C_3) = \{(3,3),(3,4),(4,4)\}$. The set J is their union with seven elements, and we need $p = 3$ additional slack matrix variables Y^s. The auxiliary feedback matrix K_{aux} is a decentralized LQR control law. Furthermore, the terminal cost matrix S is simply a diagonal matrix. To compute S, Algorithm 9 stops after 32348 iterations for an accuracy of $\epsilon = 10^{-2}$. The cost evolution of Algorithm 9 is shown in Figure 4.2a. As we have seen before in Section 3.5.1, the algorithm approaches the optimal value from below. The resulting value of δ is 23.32 and the terminal cost matrix S is given by $S = \text{diag}(5.41, 30.6, 1, 1.26)$. The optimization horizon is $t_f = 5$. In Figures 4.2b and 4.2c, the subsequent behavior of Algorithm 8 is shown. First, it can be seen that the cost converges quickly after 12 iterations. This is achieved with the Barzilai-Borwein step size that is treated in the following Section 4.4.2. In Figure 4.2c, the norm of the difference between the feedback law of each iteration and the resulting optimal one is shown. The resulting feedback law is given by

$$K_{\text{dist}} = \begin{bmatrix} -2.62 & -0.21 & 0 & 0 \\ 0.28 & 1.23 & 0.0071 & 0 \\ 0 & 0.012 & 0.66 & 0.11 \\ 0 & 0 & -0.019 & -0.23 \end{bmatrix}.$$

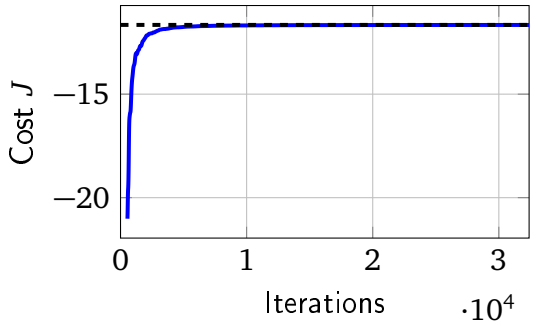
All eigenvalues of the closed loop with the resulting feedback have negative real part so the system is successfully stabilized by the resulting LQ-optimized feedback law. In addition, an interesting comparison can be made when we look at the centralized, complete infinite horizon LQ optimal feedback law resulting from the Riccati equation (2.7) which is given by

$$K_{\text{LQR}} = \begin{bmatrix} -2.62 & -0.21 & 0.0085 & 0.010 \\ 0.28 & 1.23 & 0.0027 & 0.012 \\ -0.023 & 0.0055 & 0.66 & 0.11 \\ 0.0045 & -0.0039 & -0.018 & -0.23 \end{bmatrix}.$$

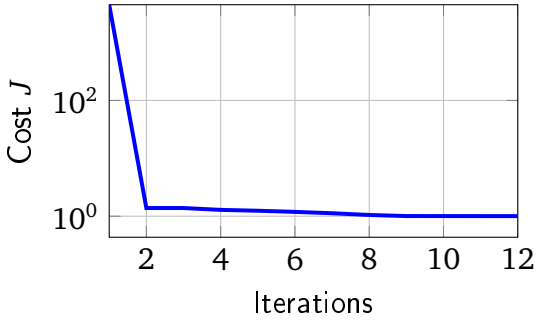
It becomes clear that the entries of the distributed control law are very close to the complete, centralized control law, especially the diagonal terms, where the largest relative difference is 0.23%. The available, nonzero off-diagonal terms of K_{dist} differ slightly, but this is to be expected as they are required to balance the unavailable entries. The resulting cost of both control laws for an infinite horizon also only differs by 0.021%.

4.3.4.2 Numerical illustration of stability

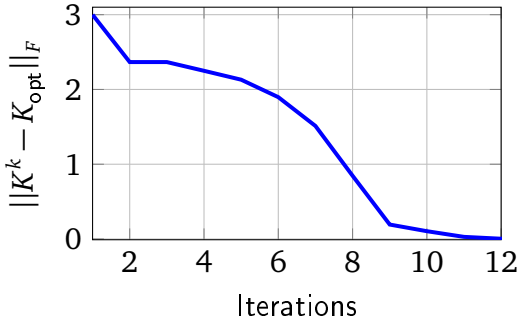
This subsection contains a numerical demonstration that the derived feedback law from the presented Algorithms 8 and 9 is indeed stabilizing. As benchmark systems, 100 different unstable test systems are randomly created. Each system has $N = 40$ subsystems, and every subsystem has $n_i = 2$ states. The systems are created in such a way that the subsystems can stabilize the system with decentralized LQR control laws designed with only their own decoupled model ($Q = I_{n_i \times n_i}, R = 100 I_{m_i \times m_i}$) which leads to a decentralized control law K_{aux}.



(a) Cost evolution of Algorithm 9. Optimal value in black (dashed) is −11.6567.



(b) Cost evolution for distributed gradient descent in Algorithm 8.



(c) Difference between $K^{(k)}$ and final K_{opt} in Algorithm 8.

Figure 4.2: Illustration of behaviors of Algorithms 8 and 9.

As a first step, solutions to (4.14) are obtained. To evaluate the performance of the distributed Algorithm 9 (S_{dist}), we compare the results with a centralized solution from Yalmip [84] (S_{Y}). Afterwards, we apply Algorithm 8 to compute the distributed control law K_{dist} for the following scenarios: (1) $t_{\text{f}} = 1$ and $S = S_{\text{Y}}$ from (4.14) solved with Yalmip, (2) $t_{\text{f}} = 1$ and $S = S_{\text{dist}}$ from (4.14) solved with Algorithm 9, (3) $t_{\text{f}} = 1$ and $S = 0_n$, (4) $t_{\text{f}} = 2$ and $S = 0_n$, (5) $t_{\text{f}} = 5$ and $S = 0_n$, (6) $t_{\text{f}} = 10$ and $S = 0_n$. The scenarios are chosen to compare two things: First, we compare the results from Yalmip with the distributed algorithm. Second, the presented approach of computing a terminal cost term for stability (Scenarios (1) and (2)) is compared with the different approach of tuning the optimization horizon to achieve stability.

The results are summarized in Table 4.1. It shows that the terminal cost term guarantees stability as expected. When the terminal cost term is not used and instead tuning of the horizon is used, we notice that it is not clear which horizon should be chosen as it is highly dependent on the given system dynamics which are not globally known. Only the last scenario without the terminal cost term gives stability for all 100 systems. Tuning of the horizon to achieve stability can then turn into a “playing of games” [102] which makes clear why the presented approach involving the terminal cost term is preferable.

Additionally, as a comparison of the results of Yalmip with the distributed Algorithm 9, we compare the costs of the resulting control laws obtained with S_Y and S_{dist} and the average relative cost difference is less than 0.3%. This indicates that the distributed algorithm gives results that are very close to the centralized results despite the fact that only neighborhood

Table 4.1: Number of stabilized systems out of 100 for the six scenarios: (1): $t_\mathrm{f} = 1$, $S = S_\mathrm{Y}$ from (4.14) with Yalmip; (2): $t_\mathrm{f} = 1$, $S = S_\mathrm{dist}$ from (4.14) with Algorithm 9; (3) $t_\mathrm{f} = 1$, $S = 0_n$; (4) $t_\mathrm{f} = 2$, $S = 0_n$; (5) $t_\mathrm{f} = 5$, $S = 0_n$; (6) $t_\mathrm{f} = 10$, $S = 0_n$.

Scenario	(1)	(2)	(3)	(4)	(5)	(6)
# stabilized systems	100	100	58	80	97	100

information is used and it demonstrates the applicability of the distributed approach.

4.3.4.3 Application to electrical power system model

In this subsection, we apply the presented algorithm to a power system model. The considered subsystem dynamics are [103, 104]

$$\begin{aligned}
\dot{\delta}_i &= \omega_i, \\
\dot{\omega}_i &= \frac{1}{H_i} P_{M_i} - \frac{D_i}{H_i}\omega_i - \frac{1}{H_i}\sum_{j\in\mathcal{N}_{\mathrm{in},i}} \Gamma_{ij}\delta_j + \frac{1}{H_i}\sum_{j=1}^{N+M} \Psi_{ij} P_{L_j}, \\
\dot{P}_{M_i} &= \frac{1}{\tau_{T_i}}(P_{G_i} - P_{M_i}), \\
\dot{P}_{G_i} &= \frac{1}{\tau_{G_i}}(P_{\mathrm{ref}_i} - P_{G_i} - \frac{1}{r_i}\omega_i).
\end{aligned}$$

Here, $i \in \{1,\dots,N\}$, δ_i is the phase angle, ω_i the frequency, P_{M_i} the turbine error state and P_{G_i} the governor error state. The system parameters are the inertia H_i, the damping coefficient D_i, the turbine and governor time constants τ_{T_i} and τ_{G_i}, as well as the droop regulator constant r_i. Furthermore, there are N nodes that have attached generators, and M additional nodes that only have loads. In addition, we have the matrices $\Gamma \in \mathbb{R}^{N\times N}$ and $\Psi \in \mathbb{R}^{N\times(N+M)}$ which are derived as follows. Starting with the adjacency matrix $\mathscr{A}$ with $\mathscr{A}_{ij} = -b_{ij}$ where b_{ij} is the susceptance of the line between the nodes i and j for $i,j \in \{1,\dots,N+M\}$, we can define the matrix $\mathscr{B} := \mathscr{A} - \mathrm{diag}(\mathscr{A}1_{N+M})$. We can then partition the matrix $\mathscr{B}$ into blocks $\left[\begin{smallmatrix} \mathscr{B}_{11} & \mathscr{B}_{12} \\ \mathscr{B}_{12} & \mathscr{B}_{22} \end{smallmatrix}\right]$ and then obtain $\Gamma = (\mathscr{B}_{11} - \mathscr{B}_{12}\mathscr{B}_{22}^{-1}\mathscr{B}_{21})$ and $\Psi = \left[-I_N \ \mathscr{B}_{12}\mathscr{B}_{22}^{-1}\right]$. This process of deriving Γ and Ψ essentially removes the load nodes from the system description and includes them into the generator nodes, which is called Kron reduction. The control input of subsystem i is P_{ref_i} and there may be exogenous disturbances in the form of deviations in power demand represented by P_{L_i}.

The considered cost functional is

$$J = \sum_{i=1}^{N} \int_0^{t_\mathrm{f}} \left[\sum_{j\in\mathcal{N}_{\mathrm{in},i}} p_{ij}(\delta_i(\tau) - \delta_j(\tau))^2 + q_i\omega_i^2(\tau) + r_i P_{\mathrm{ref}_i}^2(\tau) \right] \mathrm{d}\tau, \tag{4.15}$$

with $p_{ij} = 10, q_i = 1000, r_i = 1$ and $t_\mathrm{f} = 100$. This cost functional penalizes deviations in frequency, the control input and the differences in neighboring phases. The phase difference is proportional to deviations from the scheduled power flow between neighboring areas, which is undesired.

Concretely, we consider a modified version of the IEEE 39 bus test case. We have modified the original system for two reasons. First, we increase the number of generators from the typical 10 to 16 to reflect the trend of having more distributed power generation. Second, the original IEEE test case has no sparsity structure after the Kron reduction, i.e. the real power of the proposed approach cannot be demonstrated. The original topology that we consider is shown in Figure 4.3. The reduced topology is shown in Figure 4.4, and this is also the structure of the state matrix A, and, therefore, it represents the graphs $\mathcal{G}_{\mathrm{s}} = \mathcal{G}_{\mathrm{s,u}} = \mathcal{G}_{\mathrm{control}} = \mathcal{G}_{\mathrm{comp}}$.

To evaluate the performance of the control law, we assume that the load at node 22 increases after 0.1 seconds, i.e. $P_{L_{22}}(t) = 0.1$ for $t > 0.1$. This directly influences generators 3, 11 and 16, which is expressed in the structure of the matrix Ψ. The resulting frequency behaviors of said nodes and of their neighbors are shown in Figure 4.5. It shows that the distributed control law stabilizes the frequencies back to the desired ones. In the open loop a permanent frequency deviation remains after the disturbance.

4.3.5 Summary

In this section, we presented our results on LQ optimal and stabilizing distributed control design using only local model information. With advanced and modern distributed optimization techniques, a terminal cost term is obtained which guarantees stability of the closed loop even though a finite horizon cost functional is considered. Afterwards, a gradient descent method iteratively leads to an optimal feedback law. Numerical examples illustrated and corroborated the numerical results. In the next section, we delve into more details of the

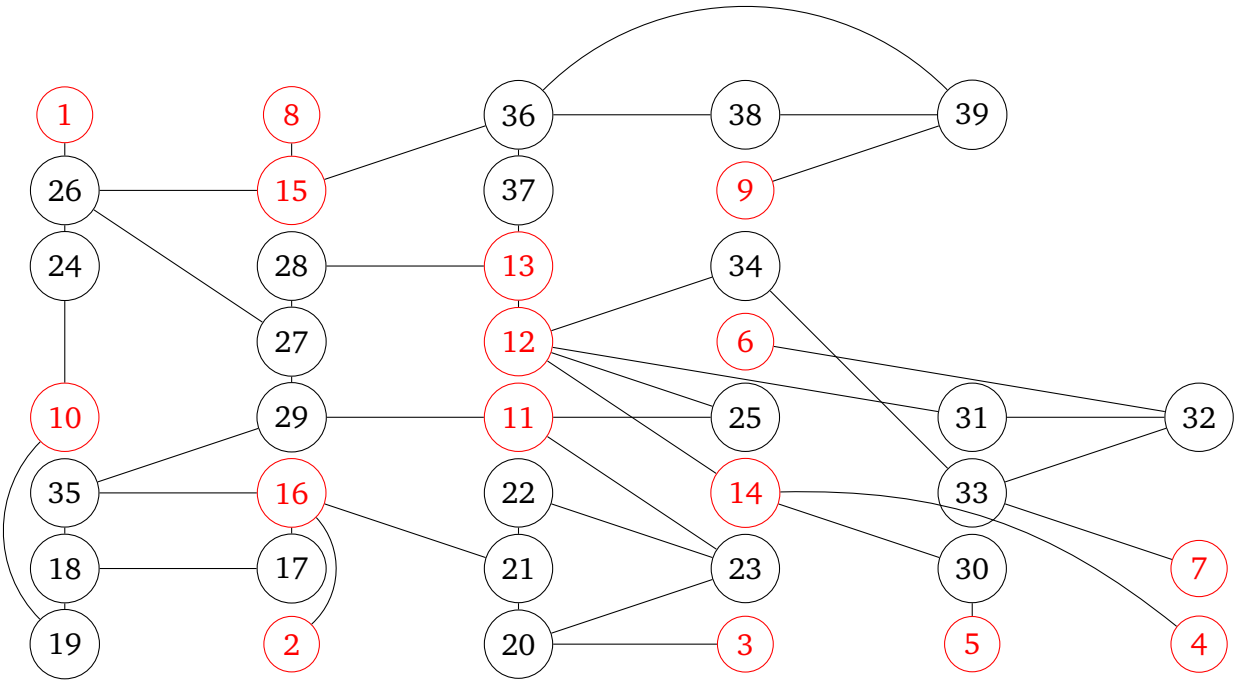


Figure 4.3: Modified IEEE 39 bus test system with 16 generators (6 additional ones). Generator nodes in red, load nodes in black.

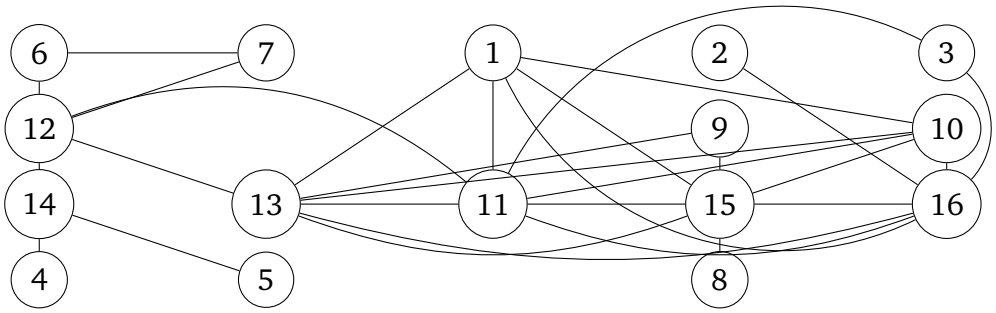


Figure 4.4: Reduced topology of modified IEEE 39 bus test system with 16 generators (6 additional ones).

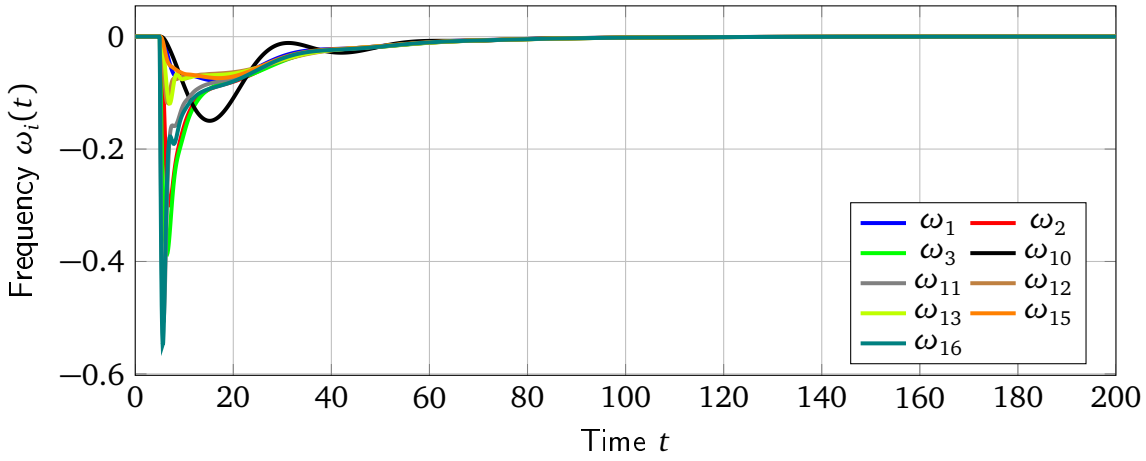


Figure 4.5: Frequency behavior after disturbance.

gradient algorithm by presenting results on the choice of the step size.

4.4 Distributed step size selection

This section addresses the selection of the step size γ_k for the distributed gradient descent method from Algorithm 8 in Section 4.3.2. Several different step sizes are compared and an extension of the gradient method to a conjugate gradient method is also considered to evaluate the performance.

4.4.1 Overview

An important property of Algorithm 8 is that it can be implemented in a distributed fashion using only local information, i.e. there is no central entity with global knowledge. However, a critical issue when using a gradient descent method is the choice of the step size denoted by γ_k. In the following, we compare two different step size methods in terms of their convergence speed and computational effort. The easiest choice for a step size is using a constant step size. The main advantage is an easy implementation and it requires no additional computations but convergence may be slow. An alternative is the so-called Barzilai-Borwein (BB)

step size [105]. This step size selection method has been shown to yield good convergence speeds, and for nonquadratic problems it might even outperform conjugate gradient methods [106]. We refer to A.1.1 for an introduction on step lengths in gradient methods. Note that even when using a descent direction, both step size methods, constant and BB, are not necessarily monotonous. In this case, the step might lead to a cost increase. To prohibit this undesired behavior, it is important to test the Powell-Wolfe conditions to guarantee convergence. The effort of this test is also taken into account in our comparison. In addition, a comparison is made with a conjugate gradient method. The challenge is to distributedly determine the Barzilai-Borwein step size, and also to compute the conjugate gradient search direction. The main concept that we employ is to use a consensus algorithm.

4.4.2 Distributed computation of Barzilai-Borwein step size

The distributed setting of the control design problem makes finding a good step size γ_k for Algorithm 8 presented in Section 4.3.2 non-trivial. The straightforward approach of performing a line search to find the optimal γ_k is not applicable in this distributed environment. In addition, the exact line search involves the solution of an optimization problem in every iteration step and can be computationally expensive. A different method for the selection of the step size is the aforementioned BB method. Applied to the presented problem, the BB method gives the step size as

$$\gamma_k = \frac{\Delta\mathrm{vec}(K_{\mathrm{dist}})^{\mathsf{T}} \Delta\mathrm{vec}(K_{\mathrm{dist}})}{\Delta\mathrm{vec}(K_{\mathrm{dist}})^{\mathsf{T}} \Delta\mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J)}, \tag{4.16}$$

where $\Delta\mathrm{vec}(K_{\mathrm{dist}}) = \mathrm{vec}(K_{\mathrm{dist}}^{(k)}) - \mathrm{vec}(K_{\mathrm{dist}}^{(k-1)})$ and $\Delta\mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J) = \mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J^{(k)}) - \mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J^{(k-1)})$. This computation requires additional storage because the feedback matrix and the gradient for the current and the last iterations are necessary.

This step size, however, cannot be directly computed in a distributed fashion using the formula from (4.16) because the scalar products in numerator and denominator contain the whole feedback law in a vectorized form. However, exactly because there are scalar products in both numerator and denominator, one can also write (4.16) as

$$\gamma_k = \frac{\frac{1}{N}\sum_{i=1}^{N} \Delta\mathrm{vec}(K_{\mathrm{dist},i})^{\mathsf{T}} \Delta\mathrm{vec}(K_{\mathrm{dist},i})}{\frac{1}{N}\sum_{i=1}^{N} \Delta\mathrm{vec}(K_{\mathrm{dist},i})^{\mathsf{T}} \Delta\mathrm{vec}(\nabla_{K_{\mathrm{dist},i}} J)},$$

which means that in both numerator and denominator, we have the sum of all the scalar products of information available to each particular agent, and then average them over the number of agents. The averaging factor $\frac{1}{N}$ can clearly be canceled to obtain (4.16). Because distributed averaging is related to the consensus problem, we can follow an idea from [107] to present a method to compute the BB step size distributedly in two steps. In the first step, each agent uses its own entries of the feedback and gradient matrices to determine an estimate of the numerator and denominator of the BB step size γ_k. In the second step, a distributed consensus algorithm gives the value of (4.16). To begin, each node i, with $i = 1, \ldots, N$, initializes the two scalar values

$$\rho_i(k(0)) = \Delta\mathrm{vec}(K_{\mathrm{dist},i})^{\mathsf{T}} \Delta\mathrm{vec}(K_{\mathrm{dist},i}),$$

and

$$\psi_i(k(0)) = \Delta\mathrm{vec}(K_{\mathrm{dist},i})^\mathsf{T} \Delta\mathrm{vec}(\nabla_{K_{\mathrm{dist},i}} J).$$

Thus, every agent uses its own respective row(s) of the feedback and gradient matrices to compute the local parameters $\rho_i(k(0))$ and $\psi_i(k(0))$ corresponding to the iteration k. Then, the agents start the following consensus iterations during which information exchange is necessary

$$\rho_i(k(t+1)) = W_{ii}\rho_i(k(t)) + \sum_{j\in\mathcal{N}_i} W_{ij}\rho_j(k(t)),$$
$$\psi_i(k(t+1)) = W_{ii}\psi_i(k(t)) + \sum_{j\in\mathcal{N}_i} W_{ij}\psi_j(k(t)).$$

Here, W is a symmetric, non-negative, doubly stochastic matrix with strictly positive diagonal entries. It is important to choose the entries of the matrix W such that its sparsity structure is in accordance with the computation graph $\mathcal{G}_{\mathrm{comp}}$. A common choice for W is according to the so-called Metropolis rule [107, 108]. This leads to the following proposition [107].

Proposition 4.3. *If the graph $\mathcal{G}_{\mathrm{comp}}$ is connected, then*

$$\lim_{t\to\infty} \frac{\rho_i(k(t))}{\psi_i(k(t))} = \gamma_k, \text{ for all } i = 1, ..., n.$$

Proof. The result follows from the doubly stochastic property of the matrix W which establishes the fact that the iteration leads to the respective averages of the ρ_i and ψ_i [80, 108]. For details on the proof, see [107]. □

The consensus phase stops when the relative difference of $\gamma_k := \frac{\rho_i(k(t))}{\psi_i(k(t))}$ between consecutive iterations t and $(t-1)$ falls below a pre-specified threshold.

Because of the prescribed structure of the matrix W, the computation of the BB step size using the consensus algorithm from Proposition 4.3 is in accordance with our requirements for local information exchange without a central entity.

4.4.3 Convergence guarantees

Because neither the BB step size nor a constant step size guarantee convergence, a test needs to be performed to check if the step size satisfies two requirements, namely the Powell-Wolfe conditions which are

$$J(K_{\mathrm{dist}}^{(k)} + \gamma_k d_k) \leq J(K_{\mathrm{dist}}^{(k)}) + c_1\gamma_k \mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J(K_{\mathrm{dist}}^{(k)}))^\mathsf{T} d_k,$$
$$\mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J(K_{\mathrm{dist}}^{(k)} + \gamma_k d_k))^\mathsf{T} d_k \geq c_2 \mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J(K_{\mathrm{dist}}^{(k)}))^\mathsf{T} d_k,$$

where c_1 and c_2 are constants to be chosen, and γ_k is initially either the BB step size or a constant step size. Furthermore, for the gradient descent method, the search direction is $d_k = -\mathrm{vec}(\nabla_{K_{\mathrm{dist}}} J(K_{\mathrm{dist}}^{(k)}))$. It can be shown that this condition is always satisfied for sufficiently small γ_k.

As both the left hand and the right hand sides of both inequalities are separable for each agent by Assumption 4.2, all agents can compute their respective summands in the terms

$$\Phi^{(k)} := \sum_{i=1}^{N} J(K_{\text{dist},i}^{(k)} - \gamma_k \nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)})) - J(K_{\text{dist},i}^{(k)}) + c_1 \gamma_k ||\text{vec}(\nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)}))||_2^2,$$

$$\Psi^{(k)} := \sum_{i=1}^{N} \text{vec}(\nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)} - \gamma_k \nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)})))^{\mathsf{T}} \text{vec}(\nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)})) + c_2 \text{vec}(\nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)}))^{\mathsf{T}} \text{vec}(\nabla_{K_{\text{dist},i}} J(K_{\text{dist},i}^{(k)})).$$

Then, a consensus phase is used to determine the average of this term. The consensus is computed correspondingly to the one in the previous subsection and Proposition 4.3. After reaching consensus, each agent can check whether $\Phi^{(k)}$ is smaller or equal to zero, and whether $\Psi^{(k)}$ is larger or equal to zero. If this is not the case, the step size needs to be reduced until the conditions are satisfied. Each test with a new step size requires a new consensus phase.

Admittedly, the test of the Powell-Wolfe conditions raises the computational effort of the approach, but we need to stress here that this is a necessary step regardless of a distributed or centralized approach. The distributed setting, however, makes the step more expensive because of the required consensus phase.

4.4.4 Distributed computation of conjugate gradient search direction

So far, Algorithm 8 uses a gradient descent, which means that the search direction d_k is simply the negative gradient, that is $d_k = -\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k)})$. A different optimization method that is superior in many applications in terms of convergence speed, is the so-called conjugate gradient (CG) method. The search direction is then

$$d_k = -\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k)}) + \beta_k d_{k-1}.$$

This means that the gradient is perturbed by the scaled previous search direction. There are several different methods to choose the scaling parameter β_k, but we restrict our attention to the Fletcher-Reeves (FR) [78] method, which gives

$$\beta_k = \frac{\text{vec}(\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k)}))^{\mathsf{T}} \text{vec}(\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k)}))}{\text{vec}(\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k-1)}))^{\mathsf{T}} \text{vec}(\nabla_{K_{\text{dist}}} J(K_{\text{dist}}^{(k-1)}))}.$$

By inspection, it becomes clear that the structure of the problem to compute β_k is identical to the structure of computing the BB step size. Therefore, we can apply the same technique of using a consensus phase for both numerator and denominator from Proposition 4.3 to converge to the result.

4.4.5 Numerical comparison of step sizes and conjugate gradient method

In order to evaluate the different step size methods, namely constant and BB, and in order to compare them to a conjugate gradient approach, Algorithm 8 is applied to 100 different test systems of two different classes; stable and unstable systems.

4.4.5.1 Stable test systems

Each of the 100 test systems has $N = 20$ subsystems and each subsystem has $n_i = 2$ states, and hence $n = 40$. The systems are created randomly and the probability of a connection between subsystems is 0.1. In this subsection, the system parameters are chosen randomly while ensuring that the overall open-loop systems are stable. In the following, we compare the overall number of iterations needed with the five alternatives (1) BB step length, (2) constant step length $\gamma = 0.01$, (3) constant step length $\gamma = 1$, (4) constant step length $\gamma = 10$, and (5) conjugate gradient method with constant step length $\gamma = 10$. Furthermore, we also compare the required tests of the Powell-Wolfe conditions for each of the 5 alternatives.

First, the optimization horizon is set to $t_\mathrm{f} = 1$. The achieved costs for all 5 cases are virtually identical with the largest difference being on the order of 10^{-4} where the range of achieved costs is between 0.39 and 0.53. Hence, in terms of achieved result, all step sizes perform equally well. Naturally, this is expected because in principle, the step size should not have a decisive influence on the final result, but this validates the approach. The results in terms of overall iterations and overall number of Powell-Wolfe (PW) checks are summarized in Table 4.2. It shows that the BB step length is vastly superior to the constant step sizes $\gamma = 0.01$ and $\gamma = 1$. For $\gamma = 10$, still less than half of the iterations are required, but the number of PW checks is larger. This needs to be taken into account because the effort of a PW check corresponds to roughly twice the effort of an algorithm iteration. Overall, however, BB is still superior because the difference in PW checks is not as large as the difference in iterations. Next, we perform the comparison with the CG method. While the overall number of iterations is still smaller, the CG method requires virtually no PW checks. Therefore, the CG method seems to be the best method overall.

Next, the same analysis is performed with an optimization horizon of $t_\mathrm{f} = 10$ with the same example systems. Again, the cost differences are smaller than the desired accuracy of the algorithm where the achieved costs are between 0.48 and 1.00 indicating identical performance in terms of the final result. The number of iterations and PW checks are summarized in Table 4.3. In this case, the CG method is the worst method with regards to iterations. In terms of PW checks, the CG method is still vastly superior to the other ones by requiring merely two checks on average. But because the BB method needs fewer main iterations, in this case, the BB method is the best method overall.

In conclusion, it is hard to make an ultimate judgment on which method is the best. It becomes clear that the BB step size is always better than a constant step size, but no clear preference can be made between BB step size or CG method. This is because the actual optimization problem to be solved by the algorithm is highly dependent on the considered system and optimization parameters. However, both clearly outperform the constant step sizes.

4.4.5.2 Unstable test systems

Consistently to the previous subsection, each of the 100 test systems has $N = 20$ subsystems and each subsystem has $n_i = 2$ states, and hence $n = 40$. The systems are created randomly and the probability of a connection between subsystems is 0.1, but in this subsection, the overall open-loop systems are not stable. As was the case for the stable systems, we compare

Table 4.2: Performance comparison for $t_f = 1$ and stable systens: (1) BB, (2) $\gamma = 0.01$, (3) $\gamma = 1$,(4) $\gamma = 10$, (5) CG.

	BB	$\gamma = 0.01$	$\gamma = 1$	$\gamma = 10$	CG
Average # Iterations	7.03	24.67	25.14	14.92	12
Average # PW checks	18.29	272.24	109.7	15.34	1.04
Average $\frac{\text{Iterations BB}}{\text{Iterations x}}$	-	0.29	0.28	0.47	0.59
Average $\frac{\text{PW checks BB}}{\text{PW checks x}}$	-	0.068	0.17	1.23	18.04

Table 4.3: Performance comparison for $t_f = 10$ and stable systens: (1) BB, (2) $\gamma = 0.01$, (3) $\gamma = 1$,(4) $\gamma = 10$, (5) CG.

	BB	$\gamma = 0.01$	$\gamma = 1$	$\gamma = 10$	CG
Average # Iterations	18.38	46.33	52.43	115.14	120.87
Average # PW checks	23.76	391.61	231.51	194.30	2.25
Average $\frac{\text{Iterations BB}}{\text{Iterations x}}$	-	0.39	0.45	0.30	0.26
Average $\frac{\text{PW checks BB}}{\text{PW checks x}}$	-	0.06	0.15	0.25	17.69

the overall number of iterations needed with the five alternatives (1) BB step length, (2) constant step length $\gamma = 0.01$, (3) constant step length $\gamma = 1$, (4) constant step length $\gamma = 10$, and (5) conjugate gradient method.

The optimization horizon is set to $t_f = 1$. The achieved costs for all 5 cases are again very similar to each other with the largest difference being on the order of 10^{-3} where the range of achieved costs is between 0.75 and 1.3. Hence, in terms of achieved result, all step sizes perform equally well. The results on overall iterations and overall number of Powell-Wolfe (PW) checks are summarized in Table 4.4.

While the overall number of iterations and PW checks increase for all 5 alternatives, the overall conclusion remains unchanged in the sense that the BB method is superior to all alternatives in terms of algorithm iterations. However, the CG method is vastly superior of PW checks which outweighs the algorithm iterations. Hence, in this case, the CG method is best overall.

Table 4.4: Performance comparison for $t_f = 1$ and unstable systems: (1) BB, (2) $\gamma = 0.01$, (3) $\gamma = 1$,(4) $\gamma = 10$, (5) CG.

	BB	$\gamma = 0.01$	$\gamma = 1$	$\gamma = 10$	CG
Average # Iterations	15.6	37.02	34.38	30.12	29.81
Average # PW checks	28.74	390.93	140.03	35.39	2.27
Average $\frac{\text{Iterations BB}}{\text{Iterations x}}$	-	0.41	0.44	0.67	0.64
Average $\frac{\text{PW checks BB}}{\text{PW checks x}}$	-	0.071	0.19	1.14	20.95

4.4.6 Summary

In this section, we developed a distributed method for computing the step size for Algorithm 8 in the form of the Barzilai-Borwein step size. We compared the performance of the step size with an alternative, namely the use of of a constant step size. Furthermore, convergence of the gradient descent is guaranteed by the Powell-Wolfe conditions, which are also evaluated in a distributed fashion. The key to these computations is consensus whose computation structure is designed to be in accordance with the computation graph $\mathcal{G}_{\text{comp}}$ such that only local information exchange is required. Therefore, the overall approach remains completely distributed and does not need a centralized computational entity. Finally, based on the insights from the step size computations, we extended the gradient descent approach to a conjugate gradient method. The effectiveness of both the Barzilai-Borwein step size and the conjugate gradient method were shown in numerical experiments. No general preference could be stated based on these results, however, both clearly outperformed the constant step size.

4.5 Event-based trajectory simulation

In the previous sections, we addressed the design of an optimal control law where the knowledge about the dynamical model is distributed among the subsystems. It is evident that the price that needs to be paid for this distribution is an increase in required communication among the agents. For Algorithm 9, we already mentioned that an event-based implementation is possible, see [68] for details. In this section, the trajectory simulation in Algorithm 8 is investigated more closely, and an approach is shown to reduce the communication effort.

4.5.1 Theoretical results

A typical approach to simulate the trajectories, i.e. to solve the ordinary differential equation (ODE) in a distributed fashion, is to use the Euler discretization. Thus, system (4.2) is approximated by

$$x_{\text{SE}}(k+1) = x_{\text{SE}}(k) + \Delta t\,[(Ax_{\text{SE}}(k) + Bu(k)]\,,\; x_{\text{SE}}(0) = x_0, \tag{4.17}$$

where $k = 0, \ldots, t_{\text{f}}/\Delta t$ and Δt is the time discretization step. However, in order to update its state, agent i needs the state information from every neighboring agent in every step, which leads to a high communication effort. In the following, the goal is to reduce this effort in Algorithm 8.

As a basic approach we propose here a novel event-based Euler scheme. We assume that local computation by the agents is cheap in comparison to information exchange. Hence, while agents can always use accurate information about their own state, they only receive quantized information from neighbors. Concretely, system (4.2) is approximated by

$$\begin{aligned} x_{\text{EBE}}(k+1) &= x_{\text{EBE}}(k) + \Delta t\big[\text{blockdiag}(A)x_{\text{EBE}}(k) + (A - \text{blockdiag}(A))q(k) + Bu(k)\big], \\ x_{\text{EBE}}(0) &= x_0, \end{aligned} \tag{4.18}$$

where $q(k)$ is given by

$$q_i(k) = \begin{cases} q_i(k-1) + \lfloor \frac{x_{\text{EBE},i}(k) - q_i(k-1)}{\Delta Q} \rfloor \Delta Q \text{ if } x_{\text{EBE},i}(k) - q_i(k-1) \geq \Delta Q \\ q_i(k-1) - \lfloor \frac{x_{\text{EBE},i}(k) - q_i(k-1)}{\Delta Q} \rfloor \Delta Q \text{ if } q_i(k-1) - x_{\text{EBE},i}(k) \geq \Delta Q \\ q_i(k-1) \text{ otherwise,} \end{cases} \tag{4.19}$$

and the block sizes of blockdiag(A) correspond to the subsystem block sizes. In words, $q_i(k)$ approximates $x_{\text{EBE},i}(k)$ by the closest multiple of the quantum ΔQ. The idea behind this is that in addition to discretizing time, the state is also discretized, thus requiring communication only when the quantized version of the state $q_i(k)$ changes. For an illustration of this quantization, see Figure 4.6.

The next theorem shows that the global error bound of this integration scheme scales linearly with Δt, which is identical to the conventional Euler method (4.17).

Theorem 4.2. *Let $\Delta Q = h\Delta t$ with $h \in \mathbb{R}_{++}$. Then, the norm of the global error $||x(k\Delta t) - x_{\text{EBE}}(k)||$ of the event-based Euler scheme (4.18),(4.19) is $O(\Delta t)$.*

Proof. The true state at step $k+1$ given by $x_{k+1} = x((k+1)\Delta t)$ is described using Taylor's theorem by

$$x_{k+1} = x_k + \Delta t A x_k + \frac{\Delta t^2}{2} \frac{\mathrm{d}}{\mathrm{d}t}(Ax(t^*)),$$

where t^* lies in the interval $[k\Delta t, (k+1)\Delta t]$. We can write $q_k = x_{\text{EBE}} + \Delta x_k$ where $|\Delta x_k| \leq \Delta Q$ from (4.19), and furthermore define the error as $e_k = x_k - x_{\text{EBE}}$. We can then write

$$e_{k+1} = e_k + \Delta t A e_k - \Delta t(A - \text{blockdiag}(A))\Delta x_k + \frac{\Delta t^2}{2} \frac{\mathrm{d}}{\mathrm{d}t}(Ax(t^*)).$$

As we are only interested in a bound on the error magnitude we can write

$$||e_{k+1}|| \leq (1 + \Delta t||A||)||e_k|| + \underbrace{\Delta t||A - \text{blockdiag}(A)||\Delta Q + \frac{\Delta t^2||A^2||}{2}}_{c(\Delta t, \Delta Q)}.$$

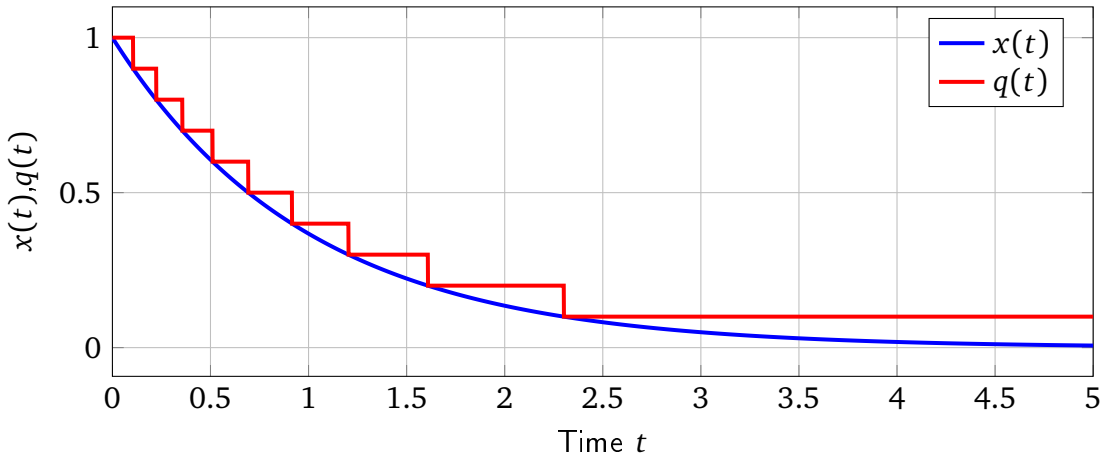


Figure 4.6: State trajectory $x(t)$ (blue) and quantized state trajectory $q(t)$ (red) for system $\dot{x}(t) = -x(t)$ and $\Delta Q = 0.1, \Delta t = 10^{-3}$.

Because the error is initially zero, i.e. $e_0 = 0$, this leads to

$$\begin{aligned} ||e_n|| &\leq c(\Delta t, \Delta Q) \sum_{k=0}^{n-1} (1 + \Delta t ||A||)^k \\ &= \left[(1 + \Delta t ||A||)^k - 1 \right] \frac{c(\Delta t, \Delta Q)}{\Delta t ||A||}. \end{aligned}$$

From this, one can obtain that [109, Chapter 2]

$$||e(t)|| \leq \frac{c(\Delta t, \Delta Q)}{\Delta t ||A||} \left[e^{||A|| t} - 1 \right].$$

Because $\Delta Q = h \Delta t$ and because the fraction depends linearly on Δt and ΔQ, it is clear that $||e(t)|| = O(\Delta t)$. □

The communication effort for the classical Euler method is comprised of the following. In each time step, each agent has to send n_i state values to all its neighbors. This is also the upper bound for the event-based Euler method. In the event-based method, however, each agent only has to send those states that changed at least by ΔQ. This reduces the required communication drastically while the computational effort remains the same. Naturally, the event-based Euler method has a larger error because the classical Euler method is a special case ($\Delta Q \to 0$), but the order of the error bound stays the same. For the ease of presentation, we focus here on the Euler method, but an extension to higher order methods is possible.

Because the dependence of the approximation error is linear in Δt and ΔQ from Theorem 4.2, both parameters need to be chosen in a similar fashion as they would be for the conventional Euler method. As this is highly dependent on the system dynamics, no general guideline can be given. However, our numerical evaluations have shown that choosing them to be on the same order leads to good results. The error should not be too large to ensure that the trajectories are accurate such that the gradient for Algorithm 8 is accurate. Otherwise convergence may not be guaranteed. With this in hand, we state our final result of this section.

Theorem 4.3. *Given are Assumptions 4.1-4.5. Using the event-based Euler scheme for the trajectory simulations in Algorithm 8 and the event-based implementation of Algorithm 9 [68] results in a complete event-based design method to determine an LQ-optimized static feedback law K_{dist} solving Problem (4.4) which is stabilizing for system (4.1),(4.2), and which is a feasible control law according to Definition 4.1.*

Proof. The theorem follows directly from Theorem 4.1, Theorem 4.2 and the results in [68]. □

Note that the idea presented here is related to the results in [110, 111], the so-called quantized state system (QSS) method. The QSS method discretizes only the state and not the time. This, however, leads to an asynchronous method that requires coordination between the agents when to take steps, thus requiring additional communication. Our approach remains a synchronous method.

4.5.2 Numerical evaluation of communication effort in Algorithm 8

In order to evaluate the results of this section, we use the same 100 example systems as in Section 4.3.4.2 but restrict our attention to the first scenario ($t_\mathrm{f} = 1, S = S_Y$). We compare the communication effort of Algorithm 8 in the case that we use a standard Euler discretization with the event-based Euler discretization. The selected quantization width is $\Delta Q = 10^{-3}$, which is identical to the time discretization $\Delta t = 10^{-3}$. The goal is to compare the ratio of the communication events that occur. We denote the number of communication events for the event-based method by $c_{x/\lambda,\mathrm{EBE}}$ and the number of communication events for the standard Euler method by $c_{x/\lambda,\mathrm{SE}}$. A communication event is the requirement to send a new state value to a neighbor. As mentioned before, the standard Euler method requires this in every step for every state and for every neighbor. The event-based method only sends those states that changed at least by ΔQ. The results are summarized in Table 4.5. It shows that the communication effort is drastically reduced using the event-based scheme because only a fraction of the communication events is required. It has to be noted, however, that the resulting costs are always slightly higher with the event-based approach, on average by 3.8%. The reason is that the gradient is slightly disturbed in comparison to the standard Euler method.

As a side comment, we note that using an emulation-based control approach, see e.g. [112], it is also possible to use event-based communication online, i.e. when the control law is in use, instead of only in the design phase. This is, however, outside of the scope of this thesis and is possible future work.

4.5.3 Summary

In this section, we developed a new event-based distributed solution method for ODE systems that reduces the communication effort between subsystems to simulate the overall system behavior. Message sending is only required when the state changes by a certain quantization value. It was shown that the error bound is on the same order as for the standard Euler method. The advantages were illustrated using numerical experiments.

Table 4.5: Comparison of communication effort for standard Euler and event-based Euler discretization in Algorithm 8.

	x-trajectories	λ-trajectories	Total
Average $c_{x/\lambda,\mathrm{SE}}$	$1.42 \cdot 10^9$	$1.42 \cdot 10^9$	$2.84 \cdot 10^9$
Minimal $c_{x/\lambda,\mathrm{SE}}$	$7.15 \cdot 10^8$	$7.14 \cdot 10^8$	$1.43 \cdot 10^9$
Maximal $c_{x/\lambda,\mathrm{SE}}$	$6.12 \cdot 10^9$	$6.11 \cdot 10^9$	$1.22 \cdot 10^{10}$
Average $c_{x/\lambda,\mathrm{EBE}}$	$1.43 \cdot 10^7$	$3.07 \cdot 10^7$	$4.50 \cdot 10^7$
Minimal $c_{x/\lambda,\mathrm{EBE}}$	$6.90 \cdot 10^6$	$3.91 \cdot 10^6$	$1.16 \cdot 10^7$
Maximal $c_{x/\lambda,\mathrm{EBE}}$	$6.92 \cdot 10^7$	$1.86 \cdot 10^8$	$2.55 \cdot 10^8$
Average $\frac{c_{x/\lambda,\mathrm{EBE}}}{c_{x/\lambda,\mathrm{SE}}}$	0.0102	0.0196	0.0149
Minimal $\frac{c_{x/\lambda,\mathrm{EBE}}}{c_{x/\lambda,\mathrm{SE}}}$	0.0048	0.0046	0.0072
Maximal $\frac{c_{x/\lambda,\mathrm{EBE}}}{c_{x/\lambda,\mathrm{SE}}}$	0.0184	0.0543	0.0340

4.6 Optimal distributed control of singular systems

Throughout Sections 4.1–4.5, the considered system dynamics consisted of an ordinary differential equation (ODE). In this section, we present an extension of the approach to singular systems, also referred to as differential-algebraic systems (DAE). Many application examples of large-scale interconnected dynamical systems can be modeled to contain some kind of conservation constraint. For example, the exact representation of the power system needs to satisfy the Kirchhoff laws. Water or other distribution systems also need to satisfy mass, energy or current balances. These conservation laws or other algebraic constraints lead to a differential-algebraic form for the system equation. For an introduction to singular systems, see [113]. In the general nonlinear form, DAE systems are often written as

$$\dot{x}(t) = f(x_{\mathrm{d}}, x_{\mathrm{a}}, u), \tag{4.20a}$$

$$0 = g(x_{\mathrm{d}}, x_{\mathrm{a}}, u), \tag{4.20b}$$

where x_{d} are dynamic states, x_{a} are purely algebraic states, and u is the input. One method to control this system class is to substitute the solution of the algebraic constraints (4.20b) into the dynamic equations (4.20a) and then to proceed with standard control design methods. However, for a general distributed system, this approach destroys the inherent sparsity structure. This structure, however, is necessary when distributed design methods with local model information are desired, e.g. in large-scale systems. The contribution of this section is a gradient descent method to iteratively and distributedly determine an optimal state feedback controller for linear singular systems. The algebraic part of the system equations are taken into account explicitly during the controller design and the algebraic variables are also used for feedback. The method to design the control law is based on the results from Section 4.3.

4.6.1 Problem formulation

In this section, we consider an interconnected LTI system consisting of N subsystems subject to linear equality constraints. More specifically, we assume that the subsystems are not coupled dynamically but only through the constraints. The dynamics of the ith agent can thus be written as a semi-explicit DAE of the form

$$\dot{x}_{\mathrm{d}_i} = A_{\mathrm{dd}_i} x_{\mathrm{d}_i} + A_{\mathrm{da}_i} x_{\mathrm{a}_i} + B_{\mathrm{d}_i} u_i, \tag{4.21a}$$

$$0 = \sum_{j=1}^{N} A_{\mathrm{ad}_{ij}} x_{\mathrm{d}_j} + \sum_{j=1}^{N} A_{\mathrm{aa}_{ij}} x_{\mathrm{a}_j} + B_{\mathrm{a}_i} u_i, \tag{4.21b}$$

where $x_{\mathrm{d}_i} \in \mathbb{R}^{n_{\mathrm{d}_i}}$, $x_{\mathrm{a}_i} \in \mathbb{R}^{n_{\mathrm{a}_i}}$, $u_i \in \mathbb{R}^{m_i}$, $A_{\mathrm{dd}_i} \in \mathbb{R}^{n_{\mathrm{d}_i} \times n_{\mathrm{d}_i}}$, $A_{\mathrm{da}_i} \in \mathbb{R}^{n_{\mathrm{d}_i} \times n_{\mathrm{a}_i}}$, $A_{\mathrm{ad}_{ij}} \in \mathbb{R}^{n_{\mathrm{a}_i} \times n_{\mathrm{d}_j}}$, $A_{\mathrm{aa}_{ij}} \in \mathbb{R}^{n_{\mathrm{a}_i} \times n_{\mathrm{a}_j}}$, $B_{\mathrm{d}_i} \in \mathbb{R}^{n_{\mathrm{d}_i} \times m_i}$ and $B_{\mathrm{a}_i} \in \mathbb{R}^{n_{\mathrm{a}_i} \times m_i}$. Again, x_{d_i} denotes the dynamic state of subsystem i and x_{a_i} is the algebraic state of subsystem i. More compactly, the system dynamics are written as

$$\dot{x}_{\mathrm{d}} = A_{\mathrm{dd}} x_{\mathrm{d}} + A_{\mathrm{da}} x_{\mathrm{a}} + B_{\mathrm{d}} u,\ x_{\mathrm{d}}(0) = x_{\mathrm{d},0}, \tag{4.22a}$$

$$0 = A_{\mathrm{ad}} x_{\mathrm{d}} + A_{\mathrm{aa}} x_{\mathrm{a}} + B_{\mathrm{a}} u. \tag{4.22b}$$

Here, as in (4.20), we denote by $x_\mathrm{d} \in \mathbb{R}^{n_\mathrm{d}}$ the differential or dynamic variables, by $x_\mathrm{a} \in \mathbb{R}^{n_\mathrm{a}}$ the algebraic variables and by $u \in \mathbb{R}^m$ the input to the system. The dimensions satisfy $n_\mathrm{d} = \sum_{k=1}^{N} n_{\mathrm{d}_k}$, $n_\mathrm{a} = \sum_{k=1}^{N} n_{\mathrm{a}_k}$, $n = n_\mathrm{d} + n_\mathrm{a}$ and $m = \sum_{k=1}^{N} m_k$. Note that $x_\mathrm{a}(0)$ is not specified, but it is determined through the algebraic constraints. This is necessary to ensure that the whole initial condition of the system is admissible and does not cause any discontinuous behavior. Furthermore, we have that $A_\mathrm{dd} = \mathrm{diag}(A_{\mathrm{dd}_i})$, $A_\mathrm{da} = \mathrm{diag}(A_{\mathrm{da}_i})$, i.e. the systems are not coupled through the dynamic equations. Instead, in this section, the coupling is expressed in the matrices of the algebraic constraints A_ad and A_aa. These are sparse matrices that signify which subsystems are coupled. This is typical for large-scale DAE systems, where every agent has only few neighbors relative to the total number of agents.

For the subsequent distributed optimal control design, we make the following assumptions about A_aa.

Assumption 4.6.

- A_aa is invertible.
- A_aa is row diagonally dominant.

The first part of Assumption 4.6 ensures that there is a unique solution to the algebraic constraint, and it makes the system a DAE of index 1, where the index is the number of derivations of parts of the system equations with respect to time necessary to obtain an ODE. The second part of the assumption is necessary to guarantee that the system can be simulated in a distributed fashion using a Jacobi algorithm (see [114]), as will be seen in Section 4.6.2.2. If this second part is not met, there are alternative methods to solve the algebraic part, e.g. BiCGstab as presented in [115]. This method can also be completely distributed (see e.g. [116]), when scalar products are computed using a consensus scheme. These methods require, however, a higher communication effort and for ease of presentation we focus on the well-known Jacobi algorithm.

A typical real world example satisfying these assumptions is (a linearized version of) a DAE model of the power system (see [117]), where the dynamic states are the states of each generator (frequency, mechanical power, etc.) and the algebraic states are voltages, currents or electrical power flows.

Lemma 4.4. *[118] If the matrix A_aa in system (4.22) is invertible, the system has no impulsive modes and the pencil $(sE - A)$ is regular, where*

$$E = \begin{bmatrix} I^{n_\mathrm{d} \times n_\mathrm{d}} & 0 \\ 0 & 0 \end{bmatrix}, A = \begin{bmatrix} A_\mathrm{dd} & A_\mathrm{da} \\ A_\mathrm{ad} & A_\mathrm{aa} \end{bmatrix}.$$

By Lemma 4.4 and Assumption 4.6, the considered system class in this section has the important characteristic of no discontinuous behavior.

As stated in the opening remarks of this section, instead of directly addressing (4.22), one could plug the solution of (4.22b) into (4.22a) to obtain an ODE. This yields

$$\dot{x}_\mathrm{d} = (A_\mathrm{dd} - A_\mathrm{da} A_\mathrm{aa}^{-1} A_\mathrm{ad}) x_\mathrm{d} + (B_\mathrm{d} - A_\mathrm{da} A_\mathrm{aa}^{-1} B_\mathrm{a}) u.$$

However, this system description in general lacks the sparsity structure of the original system because of the inversion of the matrix A_{aa}. The structure is important to enable distributed control design methods with local information exchange. Furthermore, a controller designed for the system in this reduced form does not explicitly incorporate feedback information of the static states. Therefore, we work with the system in its original form given in (4.22).

In this section, the neighborhood structure is not determined by the dynamical part of A as in the previous sections, but it is determined by the coupling through the algebraic constraints and is thus derived from the sparsity structure of the matrices A_{ad} and A_{aa}. Thus, if systems are coupled in the constraints, they are considered to be neighbors. In order to define the set of neighbors of a subsystem i, we therefore slightly modify the system graph definition and consider the directed graph $\mathcal{G}_{\mathrm{s,DAE}}(\mathcal{V}_{\mathrm{s,DAE}}, \mathcal{E}_{\mathrm{s,DAE}})$ associated with the matrices A_{ad} and A_{aa}. The vertex set $\mathcal{V}_{\mathrm{s,DAE}}$ is given by the set of subsystems $\mathcal{V}_{\mathrm{s,DAE}} = \{1, ..., N\}$, and the edge set $\mathcal{E}_{\mathrm{s,DAE}}$ contains the edge $(j,i) \in \mathcal{E}_{\mathrm{s,DAE}}$ iff $A_{\mathrm{ad},ij} \neq 0 \vee A_{\mathrm{aa},ij} \neq 0$. This means an edge $(j,i) \in \mathcal{E}_{\mathrm{s,DAE}}$ iff the algebraic constraint of subsystem i is influenced directly by either the dynamic or the algebraic states or both of agent j. We also introduce the undirected version of this system graph and denote it by $\mathcal{G}_{\mathrm{s,u,DAE}}$. The control graph $\mathcal{G}_{\mathrm{control}}$ and the computation graph $\mathcal{G}_{\mathrm{comp}}$ remain unchanged.

Definition 4.1 of feasible control laws still holds in this section except that the feedback law is now $u = -[K_{\mathrm{d}}, K_{\mathrm{a}}][x_{\mathrm{d}}^{\mathsf{T}}(t),\, x_{\mathrm{a}}^{\mathsf{T}}(t)]^{\mathsf{T}} = -K_{\mathrm{dist}}[x_{\mathrm{d}}^{\mathsf{T}}(t),\, x_{\mathrm{a}}^{\mathsf{T}}(t)]^{\mathsf{T}}$. With this definition, we declare the goal of this section, which is to solve the following optimization problem:

$$\min_{x,u} J(x,u) = \int_0^{t_{\mathrm{f}}} x_{\mathrm{d}}^{\mathsf{T}}(t) Q_{\mathrm{dd}} x_{\mathrm{d}}(t) + x_{\mathrm{a}}^{\mathsf{T}}(t) Q_{\mathrm{aa}} x_{\mathrm{a}}(t) + u^{\mathsf{T}}(t) R u(t) \mathrm{d}t, \tag{4.23a}$$

$$\text{s.t. } \dot{x}_{\mathrm{d}}(t) = A_{\mathrm{dd}} x_{\mathrm{d}}(t) + A_{\mathrm{da}} x_{\mathrm{a}}(t) + B_{\mathrm{d}} u(t),\ x_{\mathrm{d}}(0) = x_{\mathrm{d},0}, \tag{4.23b}$$

$$0 = A_{\mathrm{ad}} x_{\mathrm{d}}(t) + A_{\mathrm{aa}} x_{\mathrm{a}}(t) + B_{\mathrm{a}} u(t), \tag{4.23c}$$

$$u(t) = -K_{\mathrm{dist}}[x_{\mathrm{d}}^{\mathsf{T}}(t),\, x_{\mathrm{a}}^{\mathsf{T}}(t)]^{\mathsf{T}}, \tag{4.23d}$$

$$K_{\mathrm{dist}} \in \mathcal{K}. \tag{4.23e}$$

By Assumption 4.6 and Lemma 4.4, the system has no impulsive modes, thus guaranteeing that the cost functional (4.23a) exists. Similarly to Assumption 4.2 in the ODE case, we make the following assumption for the DAE case.

Assumption 4.7. The weighting matrices $Q_{\mathrm{dd}} \in \mathbb{S}_+^{n_{\mathrm{d}}}$ and $Q_{\mathrm{aa}} \in \mathbb{S}_+^{n_{\mathrm{a}}}$ have at most the block sparsity structure according to the computation graph $\mathcal{G}_{\mathrm{comp}}$, i.e. $Q_{\mathrm{d},ij} = 0$ and $Q_{\mathrm{a},ij} = 0$ if $(i,j) \notin \mathcal{E}_{\mathrm{comp}}$. The weighting matrix $R \in \mathbb{S}_{++}^{n}$ is block diagonal. The block sizes result from the subsystem dimensions.

Furthermore, for the distributed computation with local model information in accordance with Definition 4.1, we require the following assumption to be satisfied.

Assumption 4.8. The computation graph $\mathcal{G}_{\mathrm{comp}}$ contains the undirected DAE system graph $\mathcal{G}_{\mathrm{s,u,DAE}}$ as a subgraph, and the computation graph $\mathcal{G}_{\mathrm{comp}}$ contains the control graph $\mathcal{G}_{\mathrm{control}}$ as a – possibly different – subgraph, i.e. $\mathcal{G}_{\mathrm{s,u,DAE}}(\mathcal{V}_{\mathrm{s,DAE}}, \mathcal{E}_{\mathrm{s,u,DAE}}) \subseteq \mathcal{G}_{\mathrm{comp}}(\mathcal{V}_{\mathrm{s,DAE}}, \mathcal{E}_{\mathrm{comp}})$ and $\mathcal{G}_{\mathrm{control}}(\mathcal{V}_{\mathrm{s,DAE}}, \mathcal{E}_{\mathrm{control}}) \subseteq \mathcal{G}_{\mathrm{comp}}(\mathcal{V}_{\mathrm{s,DAE}}, \mathcal{E}_{\mathrm{comp}})$.

The consequence of Assumption 4.8 is that systems that are physical neighbors according to the system graph and systems that share control information need to cooperate in the computation during the design. However, further cooperation is possible in the approach.

4.6.2 Control synthesis

In this subsection, we present the adaptation of the results from Section 4.3 to the system dynamics (4.22). The approach follows the same steps as in the ODE case: Adjoint dynamics are derived which are employed to distributedly compute a gradient. This gradient is then used to iteratively determine the optimal control law. Afterwards, we explain in detail how everything can be computed distributedly.

4.6.2.1 Adjoint states and gradient descent direction

In this subsection, we derive the adjoint dynamics and the gradient descent direction which is then used to find the optimal control law.

To derive the adjoint states, we use the following abbreviations: $A_{K,\mathrm{dd}} = (A_{\mathrm{dd}} - B_{\mathrm{d}}K_{\mathrm{d}}), A_{K,\mathrm{da}} = (A_{\mathrm{da}} - B_{\mathrm{d}}K_{\mathrm{a}}), A_{K,\mathrm{ad}} = (A_{\mathrm{ad}} - B_{\mathrm{a}}K_{\mathrm{d}}), A_{K,\mathrm{dd}} = (A_{\mathrm{dd}} - B_{\mathrm{a}}K_{\mathrm{a}})$.

Proposition 4.4. *Given optimization problem (4.23), the gradient of the cost functional (4.23a) with respect to the controller blocks $K_{\mathrm{d}_{ij}}$ and $K_{a_{ij}}$ are given by*

$$(\nabla_{K_{\mathrm{d}}} J)_{ij} = \int_0^{t_{\mathrm{f}}} -2R_i u_i x_{\mathrm{d}_j}^{\mathsf{T}} + (B_{\mathrm{d}_i}^{\mathsf{T}}\lambda_{\mathrm{d}_i} - B_{\mathrm{a}_i}^{\mathsf{T}}\lambda_{\mathrm{a}_i}) x_{\mathrm{d}_j}^{T}\,\mathrm{d}t,$$
$$(\nabla_{K_{\mathrm{a}}} J)_{ij} = \int_0^{t_{\mathrm{f}}} -2R_i u_i x_{\mathrm{a}_j}^{\mathsf{T}} + (B_{\mathrm{d}_i}^{\mathsf{T}}\lambda_{\mathrm{d}_i} - B_{\mathrm{a}_i}^{\mathsf{T}}\lambda_{\mathrm{a}_i}) x_{\mathrm{a}_j}^{\mathsf{T}}\,\mathrm{d}t,$$

where the adjoint states follow the dynamics

$$\dot{\lambda}_{\mathrm{d}} = -A_{K,\mathrm{dd}}^{\mathsf{T}}\lambda_{\mathrm{d}} + A_{K,\mathrm{ad}}^{\mathsf{T}}\lambda_{\mathrm{a}} + 2(Q_{\mathrm{dd}} + K_{\mathrm{d}}^{\mathsf{T}}RK_{\mathrm{d}})x_{\mathrm{d}} + 2K_{\mathrm{d}}^{\mathsf{T}}RK_{\mathrm{a}}x_{\mathrm{a}}, \quad \lambda_{\mathrm{d}}(t_{\mathrm{f}}) = 0, \tag{4.24a}$$
$$0 = -A_{K,\mathrm{da}}^{\mathsf{T}}\lambda_{\mathrm{d}} + A_{K,\mathrm{aa}}^{\mathsf{T}}\lambda_{\mathrm{a}} + 2(Q_{\mathrm{aa}} + K_{\mathrm{a}}^{\mathsf{T}}RK_{\mathrm{a}})x_{\mathrm{a}} + 2K_{\mathrm{a}}^{\mathsf{T}}RK_{\mathrm{d}}x_{\mathrm{d}}. \tag{4.24b}$$

Proof. The proof is similar to the proof of Proposition 4.1.

The Lagrange function for the problem is written as

$$L = \int_0^{\mathsf{T}} \Big(x_{\mathrm{d}}^{\mathsf{T}}Q_{\mathrm{dd}}x_{\mathrm{d}} + x_{\mathrm{a}}^{\mathsf{T}}Q_{\mathrm{aa}}x_{\mathrm{a}} + x_{\mathrm{d}}^{\mathsf{T}}K_{\mathrm{d}}^{\mathsf{T}}RK_{\mathrm{d}}x_{\mathrm{d}} + x_{\mathrm{a}}^{\mathsf{T}}K_{\mathrm{a}}^{\mathsf{T}}RK_{\mathrm{a}}x_{\mathrm{a}} + 2x_{\mathrm{a}}^{\mathsf{T}}K_{\mathrm{a}}^{\mathsf{T}}RK_{\mathrm{d}}x_{\mathrm{d}}$$
$$+ \lambda_{\mathrm{d}}^{\mathsf{T}}(\dot{x}_{\mathrm{d}} - A_{K,\mathrm{dd}}x_{\mathrm{d}} - A_{K,\mathrm{da}}x_{\mathrm{a}}) + \lambda_{\mathrm{a}}^{\mathsf{T}}(A_{K,\mathrm{ad}}x_{\mathrm{d}} + A_{K,\mathrm{aa}}x_{\mathrm{a}})\Big)\mathrm{d}t + \mu^{\mathsf{T}}(x_{\mathrm{d},0} - x_{\mathrm{d}}(0)).$$

The first adjoint equation is obtained by requiring that $\frac{\partial L}{\partial x_{\mathrm{d}}} = 0$. From this, we obtain (4.24a) and

$$\mu = -\lambda_{\mathrm{d}}(0).$$

Since μ is not necessary in the following, we disregard it, but it gives us the justification that $\lambda_{\mathrm{d}}(0)$ is free while $\lambda_{\mathrm{d}}(t_{\mathrm{f}})$ is fixed to 0. The second adjoint equation comes from $\frac{\partial L}{\partial x_{\mathrm{a}}} = 0$. The gradient is then obtained from the derivatives $\frac{\partial L}{\partial K_{\mathrm{d}}}$ and $\frac{\partial L}{\partial K_{\mathrm{a}}}$. □

Using the proposed gradient descent direction, Algorithm 10 can be used to find a locally optimal controller.

As the initializing feedbacks $K_{\mathrm{d}}^{(0)}$ and $K_{\mathrm{a}}^{(0)}$ every choice is possible which satisfies the allowed structure of the controller. An obvious choice would be the zero matrix of appropriate size.

For the step size γ_k, we suggest a Barzilai-Borwein (BB) step size as presented in Section 4.4.2, which can be computed distributedly using consensus. Also note that it is possible to average the initial condition similarly to the results in Section 4.3.2 because all states and adjoint states depend linearly on $x_{\mathrm{d},0}$. This results in Algorithm 11.

Remark 4.8. Unless the resulting controller has entries such that rows or columns of $(A_{\mathrm{aa}} - B_{\mathrm{a}}K_{\mathrm{a}})$ become linearly dependent, thus making the matrix non-invertible, the controller does not cause any impulsive modes and the pencil $(sE-(A-BK))$ is regular.

Remark 4.9. The presented approach finds a controller for systems that inherently satisfy the given equality constraints which are part of the system dynamics of semi-explicit singular systems by themselves. The problem considered in this section is not directly related to controller design for ODE systems that ensures satisfaction of equality constraints which we would want to impose on the system.

As expected, and as observed in the ODE case, the communication effort of the distributed design is higher than for a centralized design. The reason for this is, again, the requirement to simulate the system trajectories which requires neighborhood information. However, note that the design can be performed entirely offline. Online, that is during the process, a state measurement exchange between neighboring nodes is necessary to compute the control input using the feedback K_{dist}, but this is typical for the use of distributed control laws and no disadvantage of the design.

Algorithm 10 Gradient descent algorithm for optimal feedback design for singular systems.

1. Simulate the states $x_{\mathrm{d}_i}(t), x_{\mathrm{a}_i}(t)$ of system (4.21) for the finite horizon t_{f}.
2. Simulate the adjoint states $\lambda_{\mathrm{d}_i}(t), \lambda_{\mathrm{a}_i}(t)$ for the same finite horizon t_{f} in the backwards direction according to Eqs. (4.24).
3. Every agent calculates the respective entries of the gradient by the formulas given in Proposition 4.1.
4. For each neighboring agent j, update
$$K_{\mathrm{d}_{ij}}^{(k+1)} = K_{\mathrm{d}_{ij}}^{(k)} - \gamma_k (\nabla_{K_{\mathrm{d}}} J)_{ij}^{(k)},$$
$$K_{\mathrm{a}_{ij}}^{(k+1)} = K_{\mathrm{a}_{ij}}^{(k)} - \gamma_k (\nabla_{K_{\mathrm{a}}} J)_{ij}^{(k)},$$
with a suitable scalar step length γ_k, independent of i, j.
5. If all $||(\nabla_K J)_{ij}^{(k)}|| < \epsilon$, or if a different stopping criterion is satisfied, stop. Otherwise, increase k and go back to 1.

Algorithm 11 Gradient descent algorithm for optimal feedback design for singular systems including initial condition averaging.

1. Simulate the states $x_{\mathrm{d}_i,l}(t), x_{\mathrm{a}_i,l}(t)$ of system (4.21) for the finite horizon t_f for every initial condition e_l with $l = 1,\dots,n$.
2. Simulate the adjoint states $\lambda_{\mathrm{d}_i,l}(t), \lambda_{\mathrm{a}_i,l}(t)$ for the same finite horizon t_f in the backwards direction according to Eqs. (4.24).

Steps 3-5 are identical to Algorithm 10.

4.6.2.2 Distributed computation

In this subsection, we show how it is possible to simulate the states and adjoint states using only local and neighborhood information according to the computation graph $\mathcal{G}_\mathrm{comp}$ and hence how to compute the gradient. One method to simulate semi-explicit singular systems as considered in this section is to solve the algebraic part and to plug in the solution into the dynamic part in every step of the simulation.

The closed loop of the dynamic state (4.21a) for subsystem i is written as

$$\dot{x}_{\mathrm{d}_i} = A_{\mathrm{dd}_i} x_i - B_{\mathrm{d}_i} \sum_{(j,i)\in\mathcal{G}_\mathrm{control}} (K_{\mathrm{d}_{ij}} x_{\mathrm{d}_j} + K_{\mathrm{a}_{ij}} x_{\mathrm{a},j}).$$

It is easy to see that each agent i can update its dynamic state by communicating state information with its neighbors according to the control graph $\mathcal{G}_\mathrm{control}$ which is a subgraph of the computation graph $\mathcal{G}_\mathrm{comp}$. As for the algebraic part, we can rewrite (4.21b) as

$$-\sum_{(j,i)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{aa}_{ij}} x_{\mathrm{a}_j} = \sum_{(j,i)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{ad}_{ij}} x_{\mathrm{d}_j}.$$

Using a simple Jacobi algorithm, this system of linear equations can be solved for x_{a_j} using only information from the neighbors following the iteration

$$x_{\mathrm{a}_i} = -A_{K,\mathrm{aa}_{ii}}^{-1}\Big[\sum_{(j,i)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{aa}_{ij}} x_{\mathrm{a}_j} + \sum_{(j,i)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{ad}_{ij}} x_{\mathrm{d}_j}\Big].$$

Similar investigations can be performed for the adjoint state. The dynamic part of subsystem i is written as

$$\dot{\lambda}_{\mathrm{d}_i} = -\sum_{(i,j)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{dd}_{ji}}^\mathsf{T} \lambda_{\mathrm{d}_j} - 2\sum_{(i,j)\in\mathcal{G}_\mathrm{control}} K_{\mathrm{d}_{ji}}^\mathsf{T} R_j u_j + \sum_{(i,j)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{ad}_{ji}}^\mathsf{T} \lambda_{\mathrm{a}_j} + 2Q_{\mathrm{d}_i} x_i.$$

Similarly, for the static co-state we obtain

$$0 = -\sum_{(i,j)\in\mathcal{G}_\mathrm{control}} A_{K,\mathrm{da}_{ji}}^\mathsf{T} \lambda_{\mathrm{d}_j} - 2\sum_{(i,j)\in\mathcal{G}_\mathrm{control}} K_{\mathrm{a}_{ji}}^\mathsf{T} R_j u_j + (i,j)\in\mathcal{G}_\mathrm{control} A_{K,\mathrm{aa}_{ji}}^\mathsf{T} \lambda_{\mathrm{a}_j} + 2Q_{\mathrm{a}_i} y_i.$$

This can also be solved for λ_{a_j} using a Jacobi algorithm using only information (states, inputs, system model) from neighbors of the computation graph $\mathcal{G}_\mathrm{comp}$.

As for the gradient, the formulation in Proposition 4.1 makes it obvious that the gradients can be computed locally if agent i can communicate with agent j, and because $\mathcal{G}_\mathrm{control} \subseteq \mathcal{G}_\mathrm{comp}$ this is clearly possible.

4.6.3 Numerical results

In this subsection, we present several numerical experiments to illustrate the contributions of this section of the thesis. First, we give a short illustrative example. Second, we compare the resulting controllers to a centralized approach. Third, the advantages of using the averaged initial condition are shown. Last, we show a practical example of a small power system.

4.6.3.1 Illustrative example

To illustrate the approach to the reader, we treat an example system with $N = 4$ subsystems, each having only one dynamic state, one static state and one input.

The corresponding graph $\mathcal{G}_{\text{s,DAE}}$ is shown in Figure 4.7. We also show the allowed communication topology which is the undirected version of the same graph. From this, we obtain that both feedback matrices K_{d} and K_{a} have the following form for all iterations (except for the first iteration where the feedback matrices are zero)

$$K_{\text{d/a}} = \begin{bmatrix} * & 0 & * & * \\ 0 & * & 0 & * \\ * & 0 & * & 0 \\ * & * & 0 & * \end{bmatrix},$$

which reflects the desired communication structure. The nonzero entries are optimized by the distributed algorithm and every agent only manipulates its own row using information from its neighbors, e.g. the first subsystem makes changes to K_{11}, K_{13} and K_{14}. In Figure 4.8, we also show the cost resulting from using the feedback matrices in every iteration and we see that the cost is monotonically decreasing as guaranteed by the Powell-Wolfe conditions.

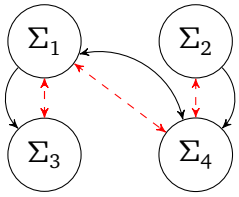


Figure 4.7: Graph for the example system. Physical coupling in black (solid), control communication in red (dashed).

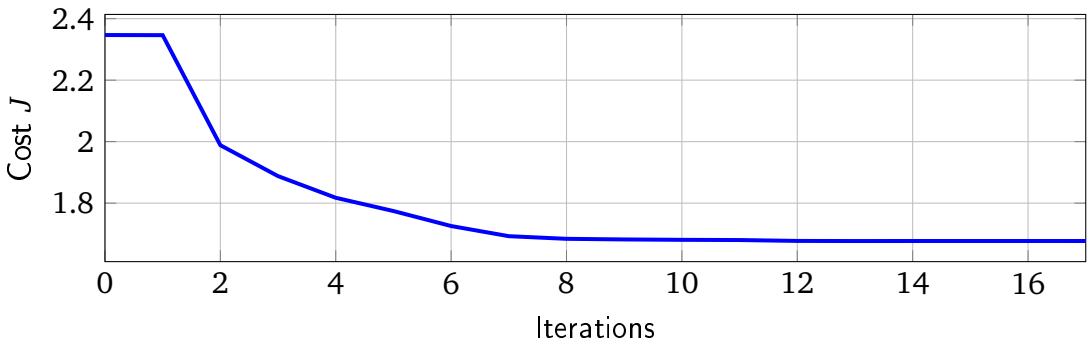


Figure 4.8: Cost evolution over the iterations of the algorithm.

4.6.3.2 Comparison between sparse and non-sparse controller

In order to evaluate the performance of the presented distributed controller, we compare the performance of the resulting controller of Algorithm 11 with a prescribed sparsity structure with the resulting controller without the structure (K is a full matrix). As test systems, we randomly construct singular systems of the form given in Eq. (4.22). We use systems with $N = 4$ subsystems, each having a random number of dynamic (between 1 and 3) and static states (1 or 2) and each having 1 input. We perform the comparison for 100 randomly created systems for which a stabilizing feedback is obtained with the gradient method. On average, the systems have a total number of 12 states (dynamic and static combined). The weighting matrices Q and R are set to be identity matrices of appropriate sizes. We choose a finite horizon of $t_\mathrm{f} = 5$. With these parameters, the gradient method needs 27 iterations on average to converge. Convergence is assumed to be achieved in these simulations when the largest element of the gradient is smaller than 10^{-3}. It turns out that the average cost difference is only 0.41% in favor of the non-sparse controller. The reason for this is that the optimal controller obtained without a demanded sparsity structure still results in a sparse controller. Note that the same is true for the feedback matrices resulting from the centralized infinite horizon result from [118]. Naturally, it is to be expected that the difference increases with increasing system size. Nevertheless, only the presented approach is able to combine the goal of an optimal controller with the desire to secure model data privacy of the subsystems.

4.6.3.3 Averaged initial condition

In this subsection, we investigate the difference between Algorithm 10 and Algorithm 11. To perform this comparison, we compute the controller matrices with both algorithms and then compare the cost caused by each controller. We use two different scenarios for the comparison: First, we use a different initial condition for the cost simulations than for the controller computations in Algorithm 10. Second, we use the same initial condition that was used in the design. We perform this test for 100 randomly generated systems which were created the same way as in the previous numerical example. In this case, on average, the total number of states is also 12. The simulation horizon is $t_\mathrm{f} = 1$. As for the simulation effort, the Jacobi algorithm for typical systems of the used system size needs 16 iterations on average to converge. When using a different initial condition for the cost simulations, the resulting controllers with the averaged initial condition produce a cost that is on average 10.72% smaller than the cost with one arbitrary initial condition, displaying the positive effect of the averaging process. In addition, according to these simulations, Algorithm 11 only needs 20 iterations on average to converge, while Algorithm 10 needs 42. Granted, the individual iterations of the averaging process are more costly, but combined with the cost improvement, Algorithm 11 should be the preferred option in this scenario.

In the second case, where we use the same initial condition for the cost calculation as in the design, the cost with the controller of Algorithm 10 is 0.22% lower than the cost with the controller of Algorithm 11, which can be considered to be negligible. This illustrates that Algorithm 11 achieves optimality independently of the initial condition of the process.

4.6.3.4 Small power system

As a practical example, we apply the method to a small power system with $n_{\text{total}} = 8$ buses where we have $n_g = 5$ generator nodes and 3 load nodes. We consider the following example of a power system with 8 nodes [119]. The interconnection graph of the generators and loads is shown in Figure 4.9. Each of the 5 generators follows the differential equation given in [120]

$$\begin{bmatrix} \dot{\delta}_i \\ \dot{\omega}_i \\ \dot{P}_{m,i} \\ \dot{a}_i \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{D_i}{J_i} & \frac{1}{J_i} & 0 \\ 0 & 0 & -\frac{1}{T_{u,i}} & \frac{1}{T_{u,i}} \\ 0 & -\frac{1}{T_{g,i}} & 0 & -\frac{R_i}{T_{g,i}} \end{bmatrix}}_{A_{\text{dd}_i}} + \underbrace{\begin{bmatrix} 0 \\ -\frac{1}{J_i} \\ 0 \\ 0 \end{bmatrix}}_{A_{\text{ad}_i}} E_{0,i} I_{q,i} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{T_{g,i}} \end{bmatrix}}_{B_{\text{d}_i}} u,$$

where δ_i is the phase angle of the generator, ω_i is the angle velocity, $P_{m,i}$ is the mechanical power, a_i is the valve position, D_i is the damping coefficient, J_i is the inertia constant, $T_{u,i}$ is a time constant representing the delay between the control valves and the turbine nozzles, $T_{g,i}$ is the time constant of the valve servomotor, and R_i is the permanent speed droop of the turbine [121]. Each generator thus has the dynamic states $x_{\text{d}_i} = [\delta_i, \omega_i, P_{m,i}, a_i]^{\mathsf{T}}$ and x_{d} of the total system is the stacked vector of all x_{d_i}.

Each generator node has the four algebraic variables $x_{\text{a,G}_i} = [V_i, \theta_i, I_{d,i}, I_{q,i}]$, each load node has only two algebraic states $x_{\text{a,L}_i} = [V_i, \theta_i]$, where V_i is the bus voltage magnitude, θ_i is the bus voltage angle, $I_{d,i}$ is the d-axis current and $I_{q,i}$ is the q-axis current. All algebraic variables need to satisfy the following algebraic constraints given in [120]

$$0 = V_i e^{j\theta_i} + (R_{s,i} + jX_{d,i})(I_{d,i} + jI_{q,i}) e^{j(\delta_i - \frac{\pi}{2})} - E_{0,i} e^{j\delta_i},\ i = 1, ..., n_g, \tag{4.25a}$$

$$0 = V_i e^{j\theta_i} (I_{d,i} - jI_{q,i}) e^{-j(\delta_i - \frac{\pi}{2})} + P_{L,i}(V_i) + jQ_{L,i}(V_i) - \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})},\ i = 1, ..., n_g, \tag{4.25b}$$

$$0 = P_{L,i}(V_i) + jQ_{L,i}(V_i) - \sum_{k=1}^{n} V_i V_k Y_{ik} e^{j(\theta_i - \theta_k - \alpha_{ik})},\ i = n_g + 1, ..., n_{\text{total}}, \tag{4.25c}$$

where $R_{s,i}$ and $X_{d,i}$ are internal resistors and impedances of the generators, Y is the magnitude of the admittance matrix of the network and α is the corresponding angle of the

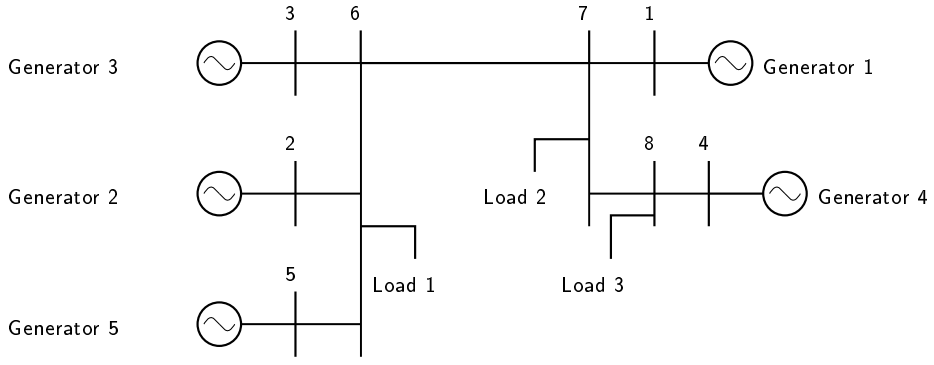


Figure 4.9: 8 bus power system with 5 generators and 3 load nodes.

admittance matrix. The loads in this system are assumed to have a linear dependence with respect to the bus voltage, i.e. $P_{\mathrm{L},i}(V_i) = k_{p,i}V_i$ and $Q_{\mathrm{L},i}(V_i) = k_{q,i}V_i$. The equations (4.25a)-(4.25c) can be split into real and imaginary parts and then linearized around operating points $\delta_{i,0}, V_{i,0}, \theta_{i,0}, I_{d,i,0}, I_{q,i,0}$. The operating points result from a load flow calculation. These linear equations can then be written in the form $0 = A_{\mathrm{ad}}x_{\mathrm{d}} + A_{\mathrm{aa}}x_{\mathrm{a}}$, where A_{ad} is block diagonal, because the equations for node i only depend on x_{d_i} in the form of δ_i. The block A_{aa} has a sparsity structure resembling that of the admittance matrix Y. The algebraic equations have no input so B_{a} is zero in this case.

The dynamical part can also be written in a form $\dot{x}_{\mathrm{d}} = A_{\mathrm{dd}}x_{\mathrm{d}} + A_{\mathrm{da}}x_{\mathrm{a}} + B_{\mathrm{d}}u$, where A_{dd} and A_{da} are block diagonal with A_{dd_i} and A_{da_i} on the respective diagonal. B_{d} is also block diagonal with B_{d_i} on the diagonal. For the considered system size, we have $x_{\mathrm{d}} \in \mathbb{R}^{20}$ and $x_{\mathrm{a}} \in \mathbb{R}^{26}$.

Thus, we obtain a singular system of the considered system class. We apply Algorithm 11 to the system with a time horizon $t_{\mathrm{f}} = 10\mathrm{s}$. The weighting matrices are $Q = \mathrm{diag}(Q_{\mathrm{d}}, Q_{\mathrm{a}}) = I^{46\times 46}$ and $R = I^{5\times 5}$ with appropriate units. The algorithm stops after 60 iterations when the change in the gradient is considered to be small (less than $\epsilon = 0.02$). During these iterations, the nodes only communicate with their neighbors according to $\mathcal{G}_{\mathrm{comp}}$ to simulate the trajectories, which requires considerable information exchange but realizes privacy.

As we take a closer look at the interconnection graph of the system, we observe that none of the generators (the dynamic parts of the system) are directly coupled. This means that during the design phase, the generators do not communicate directly with each other but only with the load nodes. In the process phase, K_{d} is, for this reason, essentially a decentralized controller and no dynamic states are communicated between the systems. However, the nodes 6-8, which only have static states, are coupled to the generator nodes and they can be used. Thus, K_{a} is not decentralized, but its coupling structure is based on the structure of the admittance matrix. Therefore, K_{d} and K_{a} have the following structures

$$K_{\mathrm{d}} = \begin{bmatrix} * & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & * \end{bmatrix}, K_{\mathrm{a}} = \begin{bmatrix} * & 0 & 0 & 0 & 0 & 0 & * & 0 \\ 0 & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & * & * & 0 & 0 \end{bmatrix}.$$

In summary, for this example system, the main difference to classical decentralized control is that this controller also makes use of local static variables (voltages, currents) and additionally static variables from adjacent load nodes.

For power systems, it is of interest how the system reacts to a disturbance. For this reason, we simulate a load increase in one of the loads by offsetting the corresponding bus voltage between time 0.1 and 0.2. The simulated state trajectories of the phase angles are shown in Figure 4.10. It shows that the controller from the presented algorithm handles the disturbance well and stabilizes the system to the equilibrium.

4.6.4 Summary

In this section, we adapted the results from the previous sections to differential-algebraic systems. The general approach remained unchanged and consists of deriving adjoint state

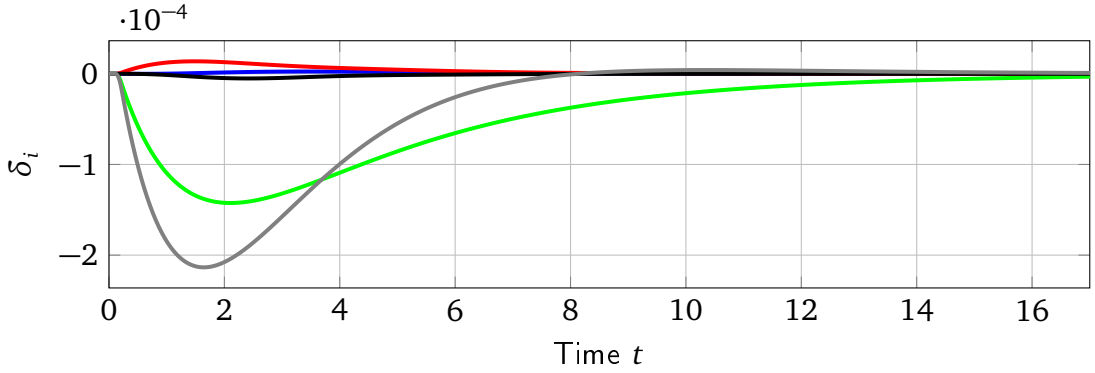


Figure 4.10: Evolution of the phase angles δ_i of the 5 generators.

dynamics and a gradient of the cost functional with respect to the feedback matrix entries. Averaging of the initial condition is still possible and we made use of the same step size techniques as in the ODE case. It has to be noted that the computational effort of the approach is quite large because it requires the solution of a Jacobi iteration in each time step.

4.7 Application of distributed control in optimal formation control

A relevant and important application of distributed or structured control laws is formation control of a multi-robot system. In this section, we present the adaptation of our iterative control design method to this problem. We start by formulating a state space model for multi-robot cooperation. Then, we make use of the approach of the adjoint states and simulated trajectories already employed in the previous Sections 4.3 and 4.6 to derive an optimal control law that maintains a relaxed rigidity requirement. The focus in this section is not on the distributed design under model data privacy constraints, which enables us to use a Quasi-Newton method. Therefore, in addition to a gradient descent, we also employ the Quasi-Newton method BFGS and analyze its benefits. The approach is used to solve an optimization problem that does not allow a standard analytical solution. The considered cost functional is similar to the previously considered LQ cost functional but is extended by a biquadratic term to include the formation constraint. Numerical evaluations are presented to validate the approach.

4.7.1 Problem formulation

In this subsection, we specify the present problem raised in this section. We start with describing the overall situation, after which we set up a state space model for interconnected cooperative multi-robot teams and then formulate the formation constraint in the form of an optimization problem that resembles the ones we have encountered in previous sections.

The schematics of the cooperating mobile manipulators are depicted in Figure 4.11 with the attached coordinate systems. For the ith manipulator, position and orientation of the end-effector frame Σ_i are expressed in a world coordinate system Σ_w. An object-centered

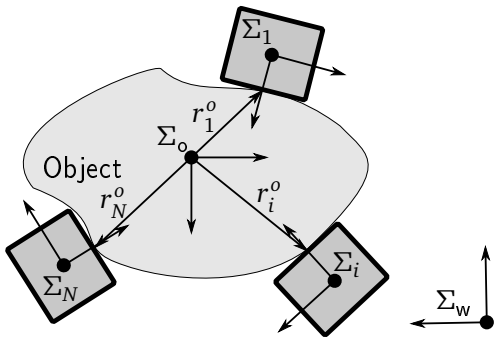


Figure 4.11: Illustration of the coordinate frames for robots, object, and world [25].

frame Σ_o is aligned to the principal axes of the object. The matrix R_o^i describes the rotation of Σ_i relative to Σ_o.

4.7.1.1 State space model for multi-robot cooperation

In this subsection, we formulate the required state space model for the optimal control design problem. We consider a cooperative team of interacting robots $i = 1, \ldots, N$, each one evolving according to the inverse dynamic feedback-linearized impedance control.

Impedance control scheme
The fact that an impedance control scheme enforces a compliance of the manipulator to its environment makes it attractive for manipulation tasks where it is hence widely used. According to the impedance control scheme the system dynamics of one manipulator evolve according to [122]

$$M_i \ddot{\xi}_i + D_i \left(\dot{\xi}_i - \dot{\xi}_{i,\mathrm{d}} \right) + K_i \left(\xi_i - \xi_{i,\mathrm{d}} \right) = f_{i,\mathrm{d}} - f_i. \tag{4.26}$$

Here, $\xi_i \in \mathbb{R}^n, i = 1, \ldots, N$ is the Cartesian position of the ith manipulator, its time derivatives $\dot{\xi}_i$ and $\ddot{\xi}_i$ are velocity and acceleration, respectively. The matrices $M_i \in \mathbb{S}^n_{++}$, $D_i \in \mathbb{S}^n_{++}$, and $K_i \in \mathbb{S}^n_{++}$ are the inertia, damping, and stiffness constituting the motion control scheme with respect to the control inputs; the desired force $f_{i,\mathrm{d}}$ and the desired velocity $\dot{\xi}_{i,\mathrm{d}}$. Because our multi-robot system is driven by a velocity interface, the desired manipulator position is integrated from the desired velocity as $\xi_{i,\mathrm{d}} = \int_{t_0}^{t} \dot{\xi}_{i,\mathrm{d}} \mathrm{d}\tau$. Furthermore, $f_i \in \mathbb{R}^n$ denotes the resulting force. For a single robotic manipulator, this force f_i arises from contact with the environment. However, since cooperating manipulators are in contact through the object, an internal force among the physically cooperating manipulators is present.

Partition of f_i
In general, we consider a measured end-effector force f_i^* composed of rigid-body dynamics f_{motion}, external force f_{ext}, and internal force $f_{i,\mathrm{int}}$. We make the following assumption.

Assumption 4.9. The rigid-body dynamics f_{motion} and external force f_{ext} are equally distributed among the manipulators.

The approximation in Assumption 4.9 demands equal impedance parameters M_i, D_i, and K_i for all manipulators. With this assumption, we write the measured end-effector force as

$$f_i^* = \frac{1}{N}\left(f_{\text{motion}} + f_{\text{ext}}\right) + f_{i,\text{int}}. \tag{4.27}$$

Assumption 4.9 leads to the factor $\frac{1}{N}$. The resulting force f_i in (4.26) can be unequal to the measured force f_i^*, and in our case we set $f_i = f_i^* - \frac{1}{N} f_{\text{motion}}$. Hence, the rigid body dynamics f_{motion} are suppressed in the impedance (4.26). This is reasonable because a compliance resulting from the dynamics of the object leads to a permanent undesired position deviation of the multi-robot team, for example in the case where the object mass pulls down the manipulators.

The resulting object force f_{motion} arises from Newton's second law of motion and it can generally be expressed as

$$f_{\text{motion}} = M_{\text{o}}\ddot{\xi}_{\text{o}} + f_{\text{o}}(\xi_{\text{o}}, \dot{\xi}_{\text{o}}), \tag{4.28}$$

where ξ_{o} is the position of the object with its inertia M_{o}. Here, we assume that M_{o} and f_{o} are precisely known and thus the impedance control is independent of the object dynamics. The external force f_{ext} can originate from an undesired obstacle or from a desired physical human input.

To calculate the virtual linkage model of internal forces, all measured end-effector forces in the object frame Σ_{o} are aggregated into $f^{\text{o}} = \left[f_1^{\text{o}^\mathsf{T}}, \ldots, f_N^{\text{o}^\mathsf{T}}\right]^\mathsf{T}$. Following [123], we know that internal forces f_{int} lie in the null-space of the grasp matrix

$$G = (G_1, G_2, \ldots, G_N) \quad \text{with} \quad G_i = I_3$$

with respect to all measured forces. In the case of Cartesian positions, the grasp matrix is simply the identity matrix. The grasp matrix G describes the relation of the forces of each robot frame and the object frame, and is well-established for robotic grasping and dexterous multi-fingered manipulation. As G is not square, its Moore-Penrose pseudo-inverse

$$G^\dagger = \left(G_1^\dagger, G_2^\dagger, \ldots, G_N^\dagger\right)^\mathsf{T} \quad \text{with} \quad G_i^\dagger = \frac{1}{N} I_3$$

is used for the null-space calculation of

$$f_{\text{int}}^{\text{o}} = \left[I - G^\dagger G\right] f^{\text{o}}. \tag{4.29}$$

For the sake of clarity, we make the following assumptions.

Assumption 4.10. Because we are mainly concerned with the tracking performance in this section, we assume that at the initial time t_0 all current and desired positions are equal, $\xi_i(t_0) = \xi_{i,\text{d}}(t_0)$. Initial configurations $\xi_{i,\text{d}}(t_0)$ are chosen to have no internal stress at the beginning. This is required because an internal stress at the initial time leads to an induced movement of the robotic manipulators, which is undesired for maintaining formation.

Assumption 4.11. There is no external force, that is $f_{\text{ext}} = 0$, and the exact dynamic model (4.28) is known and can be used as a basis for the subtraction from f_i as a feedforward term.

Assumption 4.12. The multi-robot team moves only at moderate velocities. Thus, the distributed impedances are in a quasi-equilibrium state and act thereby primarily via its stiffness K_i. The influences of the damping D_i and of the inertia M_i are then negligible for the internal forces in (4.29), which simplifies the cooperative robot model significantly.

With these assumptions, the major part of each force f_i^{o} in (4.29) arises from the difference between desired and current manipulator position. A balance of forces due to the impedance model is then approximated as

$$f_i^{\text{o}} = K_i(\xi_{i,\text{d}} - \xi_i), \tag{4.30}$$

where terms involving M_i and D_i vanish due to the previously made assumptions. When we establish the internal force model, we obtain the ith internal force partition by inserting (4.30) into (4.29) and evaluating the result row-wise as

$$f_{i,\text{int}} = K_i\left(\xi_{i,\text{d}} - \xi_i\right) - R_{\text{w}}^{\text{o}} G_i^{\dagger} \sum_{j=1}^{n} G_j R_{\text{o}}^{\text{w}} K_j \left(\xi_{j,\text{d}} - \xi_j\right). \tag{4.31}$$

Note that for the transformation of the forces to the world frame Σ_{w}, each agent is aware of all rotation matrices R_{w}^{o} and R_{o}^{w}.

Cooperative impedance control
Combining the results (4.26), (4.27), and (4.31), we obtain

$$\begin{aligned} &M_i\ddot{\xi}_i + D_i\left(\dot{\xi}_i - \dot{\xi}_{i,\text{d}}\right) + K_i\Big(\xi_i - \int_{t_0}^{t_{\text{f}}} \dot{\xi}_{i,\text{d}}\,\text{d}\tau\Big) = f_{i,\text{d}} - \\ &K_i\Big(\int_{t_0}^{t_{\text{f}}} \dot{\xi}_{i,\text{d}}\,\text{d}\tau - \xi_i\Big) + R_{\text{w}}^{\text{o}} G_i^{\dagger} \sum_{j=1}^{n} G_j R_{\text{o}}^{\text{w}} K_j \Big(\int_{t_0}^{t_{\text{f}}} \dot{\xi}_{j,\text{d}}\,\text{d}\tau - \xi_j\Big). \end{aligned} \tag{4.32}$$

The last term on the left-hand side and the second term on the right-hand side cancel, and furthermore we have that $G_i^{\dagger} G_i = \frac{1}{N} I_3$. With this, we can present the complete state space model.

Complete state space model
Let $x_i = \left[\left(\int_{t_0}^{t} \dot{\xi}_{i,\text{d}}\,\text{d}\tau\right)^{\mathsf{T}}, \xi_i^{\mathsf{T}}, \dot{\xi}_i^{\mathsf{T}}\right]^{\mathsf{T}}$ be the system state and $u_i = \left[\dot{\xi}_{i,\text{d}}^{\mathsf{T}}, f_{i,\text{d}}^{\mathsf{T}}\right]^{\mathsf{T}}$ be the control input. Then, a state space model for a single manipulator in cooperation results in

$$\dot{x}_i = A_{ii} x_i + B_i u_i + \sum_{j \neq i} A_{ij} x_j,$$

with

$$A_{ii} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ \frac{1}{N}M_i^{-1}K_i & -\frac{1}{N}M_i^{-1}K_i & -M_i^{-1}D_i \end{pmatrix},$$

$$A_{ij} = M_i^{-1}R_{\mathrm{w}}^{\mathrm{o}}G_i^{\dagger}\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ G_jR_{\mathrm{o}}^{\mathrm{w}}K_j & -G_jR_{\mathrm{o}}^{\mathrm{w}}K_j & 0 \end{pmatrix},$$

$$B_i = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ M_i^{-1}D_i & M_i^{-1} \end{pmatrix},$$

where A_{ii} is the system matrix of a single manipulator i, B_i is its input matrix, and A_{ij} represents the physical coupling from manipulator j to manipulator i.

These individual subsystem dynamics of each manipulator can be combined to the overall multi-robot cooperative system and we obtain the overall LTI system as

$$\dot{x} = Ax + Bu, \tag{4.33}$$

with the aggregated state vector $x = \left[x_1^{\mathsf{T}}, \ldots, x_N^{\mathsf{T}}\right]^{\mathsf{T}}$ and the aggregated input vector $u = \left[u_1^{\mathsf{T}}, \ldots, u_N^{\mathsf{T}}\right]^{\mathsf{T}}$. The complete system is written as $A = \left[A_{ij}\right]$ and $B = \mathrm{diag}(B_1, \ldots, B_N)$. This model is the basis for the optimal control problem which we formulate in the next subsection.

4.7.1.2 Formation rigidity in multi-robot cooperation

In this subsection, we describe the control goal considered in this section. A typical control approach for driving a multi-robot team from an initial configuration to a desired final configuration is depicted in Figure 4.12.

A task plan is obtained from a supervisory high-level instance, e.g. a human operator commanding the goal. Here, the task plan involves an initial configuration x_0 and a final

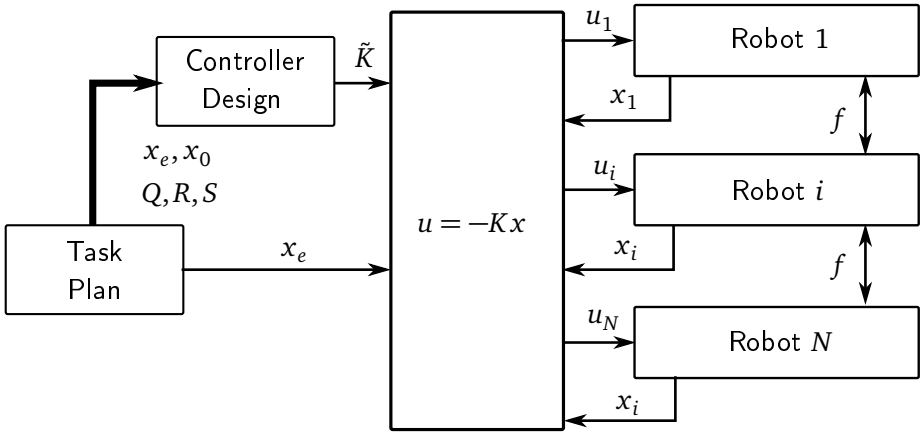


Figure 4.12: Schematic overview of the cooperative mobile manipulation control architecture [25].

configuration x_f of the multi-robot team. To achieve this, we design a linear state feedback controller $u = -Kx$ which optimally drives a formation of interconnected manipulators described by the system dynamics (4.33) from the initial condition x_0 to the desired end point x_f while maintaining the initial formation. In the following, we describe the cost functional of our LQR-like optimal control problem and how the desired rigidity is relaxed.

In order to achieve goal regulation to x_f, we use the standard transformation of our state x into

$$\tilde{x} = x - x_\mathrm{f}. \tag{4.34}$$

Then, we can formulate the LQR cost functional which gives a controller driving to the end point in an optimal fashion as

$$J = \tilde{x}^\mathsf{T}(t_\mathrm{f})S\tilde{x}(t_\mathrm{f}) + \int_{t_0}^{t_\mathrm{f}} \tilde{x}^\mathsf{T}(t)Q\tilde{x}(t) + u^\mathsf{T}(t)Ru(t)\mathrm{d}t, \tag{4.35}$$

where $S \in \mathbb{S}_+^n$, $Q \in \mathbb{S}_+^n$ and $R \in \mathbb{S}_{++}^m$ are weighting matrices expressing the desired performance.

Next, we define rigidity of the formation and explain how we can add a relaxed form of the rigidity requirement into the cost functional above. The formation is described by a static set of edges $\mathscr{E}$ with cardinality $||\mathscr{E}||$ between the manipulators such that the virtual structure of the formation is rigid during the movement phase. Rigidity of the formation is described by an edge function $f(x) = \big(\ldots, \|\xi_i - \xi_j\|, \ldots\big) \in \mathbb{R}^{\|\mathscr{E}\|}$ which is required to satisfy $f(x) = p$. Here, x is the system state from (4.33) which contains manipulator positions ξ_i, and $p = \big(\ldots, p_{ij}, \ldots\big)$ is the desired rigid distance between two manipulators. The distances p_{ij} are constant when the formation is rigid. Taking the derivative of $f(x)$ with respect to time leads to

$$\big(\xi_i - \xi_j\big)^\mathsf{T}\big(\dot{\xi}_i - \dot{\xi}_j\big) = 0 \;\forall (i,j) \in \mathscr{E}. \tag{4.36}$$

The geometrical interpretation of (4.36) is that the difference in position between two linked manipulators is orthogonal to the difference in velocity. This equation represents our second control goal of maintaining formation rigidity. Given Assumption 4.10, it is sufficient to maintain the formation instead of having to establish it because the manipulators start with the desired one. In order to include the rigidity condition (4.36) into our LQR cost functional (4.35), we transform it into a quadratic term of the states. Thus, (4.36) is written as $x_{i,j}^\mathsf{T} Q_{ij} x_{i,j}$ with $x_{i,j} = [x_i^\mathsf{T}, x_j^\mathsf{T}]^\mathsf{T}$ by defining the blocks

$$[q_{ii}] = \begin{pmatrix} 0_{n\times n} & 0_{n\times n} & 0_{n\times n} \\ 0_{n\times n} & 0_{n\times n} & \frac{1}{2}I_n \\ 0_{n\times n} & \frac{1}{2}I_n & 0_{n\times n} \end{pmatrix} \forall (i,j) \in \mathscr{E},\; \big[q_{ij}\big] = \big[q_{ji}\big] = \begin{pmatrix} 0_{n\times n} & 0_{n\times n} & 0_{n\times n} \\ 0_{n\times n} & 0_{n\times n} & -\frac{1}{2}I_n \\ 0_{n\times n} & -\frac{1}{2}I_n & 0_{n\times n} \end{pmatrix} \forall (i,j) \in \mathscr{E}.$$

The resultant matrix $Q_{ij} = \left[\begin{smallmatrix} q_{ii} & q_{ij} \\ q_{ji} & q_{jj} \end{smallmatrix}\right]$ is symmetric but indefinite and thus it cannot be employed in a standard LQR problem directly. Because the equality constraint described in Eq. (4.36) can be violated in both directions, the indefiniteness of $x_{i,j}^\mathsf{T} Q_{ij} x_{i,j}$ is obvious, and its global minimum is $-\infty$. The biquadratic term $(x_{i,j}^\mathsf{T} Q_{ij} x_{i,j})^2$, on the other hand, has a minimum

of 0, and is thus suitable to be included in an optimization to ensure relaxed rigidity. In other words, minimizing $(x_{i,j}^{\mathsf{T}} Q_{ij} x_{i,j})^2$ for all $(i,j) \in \mathscr{E}$ relaxes the equality constraint (4.36) into a minimization problem. Relaxation means that the resulting controller does not guarantee exact satisfaction of (4.36) for all times, but for appropriate weighting matrices the controller design leads to values that are at least close to zero. Proper partitioning allows writing $(x_{i,j}^{\mathsf{T}} Q_{ij} x_{i,j})^2$ as

$$(x^{\mathsf{T}} Q_k x)^2 \ \forall\ k \in \{1, ..., ||\mathscr{E}||\}\,. \tag{4.37}$$

While the design objective of goal regulation requires the transformation to the coordinates $\tilde{x}$ from (4.34), it is important to note that the relaxed rigidity condition (4.37) still needs to be satisfied in the original coordinate system x. In order to formulate a cost functional that contains both coordinate systems, we introduce an extended state vector

$$\hat{x} = \left(\tilde{x}^{\mathsf{T}}, 1\right)^{\mathsf{T}}. \tag{4.38}$$

With this state vector, we reformulate the relaxed rigidity condition (4.37) into

$$\begin{aligned}
&\left(x^{\mathsf{T}} Q_k x\right)^2 \ \forall\ k \in \{1, ..., ||\mathscr{E}||\} \\
&= \left(\begin{pmatrix} \tilde{x} \\ 1 \end{pmatrix}^{\mathsf{T}} \underbrace{\begin{pmatrix} Q_k & Q_k x_{\mathrm{f}} \\ x_{\mathrm{f}}^{\mathsf{T}} Q_k & x_{\mathrm{f}}^{\mathsf{T}} Q_k x_{\mathrm{f}} \end{pmatrix}}_{\hat{Q}_k} \begin{pmatrix} \tilde{x} \\ 1 \end{pmatrix} \right)^2 \ \forall\ k \in \{1, ..., ||\mathscr{E}||\} \\
&= \left(\hat{x}^{\mathsf{T}} \hat{Q}_k \hat{x}\right)^2 \ \forall\ k \in \{1, ..., ||\mathscr{E}||\}\,.
\end{aligned} \tag{4.39}$$

We can now combine all the terms into one cost functional and restate our control goal. The goal of our optimal control problem is to find a controller $u = -\hat{K}\hat{x}$ with structure $\hat{K} = [K, 0_{2nN\times 1}]$ in order to minimize the following cost functional

$$J = \hat{x}(t_{\mathrm{f}})^{\mathsf{T}} \hat{S} \hat{x}(t_{\mathrm{f}}) + \int_{t_0}^{t_{\mathrm{f}}} \sum_{k=1}^{||\mathscr{E}||} \left(\hat{x}^{\mathsf{T}}(t) \hat{q_k} \hat{Q}_k \hat{x}(t)\right)^2 + u^{\mathsf{T}}(t) R u(t) + \hat{x}^{\mathsf{T}} \hat{Q} \hat{x} \,\mathrm{d}t,$$

where $\hat{S}$ and $\hat{Q}$ have the structure $\hat{S} = \mathrm{diag}(S, 0)$ and $\hat{Q} = \mathrm{diag}(Q, 0)$ in order not to penalize the additional 1-state, $\hat{Q}_k$ is given in (4.39) and $\hat{q}_k$ is a positive scalar weighting factor. The term $\hat{x}(t_{\mathrm{f}})^{\mathsf{T}} \hat{S} \hat{x}(t_{\mathrm{f}})$ represents the penalty term resulting from the distance between x and x_{f} for the final time t_{f}. Control input constraints are indirectly realized by $u^{\mathsf{T}}(t) R u(t)$. This cost functional represents our combined control goals of maintained formation by the term $\sum_{k=1}^{||\mathscr{E}||} \left(\hat{x}^{\mathsf{T}}(t) \hat{Q}_k \hat{x}(t)\right)^2$, and goal regulation by the term $\hat{x}^{\mathsf{T}} \hat{Q} \hat{x}$. The zero column in $\hat{K}$ is necessary to discard the augmented 1-state from (4.38).

The overall control design can then be summarized in the following optimization problem

$$\min_{\hat{K}} J = \hat{x}(t_{\mathrm{f}})^{\mathsf{T}} \hat{S} \hat{x}(t_{\mathrm{f}}) + \int_{t_0}^{t_{\mathrm{f}}} \sum_{k=1}^{||\mathscr{E}||} \left(\hat{x}^{\mathsf{T}}(t) \hat{q_k} \hat{Q}_k \hat{x}(t)\right)^2 + u^{\mathsf{T}}(t) R u(t) + \hat{x}^{\mathsf{T}} \hat{Q} \hat{x} \,\mathrm{d}t, \tag{4.40a}$$

$$\text{s. t.} \quad \dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{B}u(t) \tag{4.40b}$$

$$u(t) = -\hat{K}\hat{x}(t) \tag{4.40c}$$

$$x(t_0) = x_0, \tag{4.40d}$$

where $\hat{A} = \text{diag}(A, 0)$ and $\hat{B} = [B^\mathsf{T}, 0^\mathsf{T}_{1\times 2nN}]^\mathsf{T}$. In the next section, an approach is presented to solve this optimization problem.

4.7.2 Iterative optimal control design for relaxed rigidity formation control

In this subsection, we present two algorithms to solve the optimization problem (4.40) in order to design a control law that achieves our two control goals; goal regulation of a group of robot manipulators and maintaining a rigid formation. Furthermore, an idea is presented to alleviate the local character of the resulting control law.

4.7.2.1 Optimal control design via gradient descent method using adjoint states

In this subsection, we describe a solution algorithm to determine a suboptimal feedback to solve the optimization problem (4.40), based on the results in Sections 4.3 and 4.6. Disregarding structural constraints, there is a linear relationship in the standard LQR problem between the primal states x and the adjoint states λ given by $\lambda(t) = P(t)x(t)$, allowing for the centralized solution to use a (differential) Riccati equation for the matrix $P(t)$, see Section 2.2.1. This is not the case here. Because of the biquadratic term in the cost functional we are forced to use an alternative method based on the previously presented iterative approach of using simulated trajectories to compute a gradient. In previous sections, the approach of using simulated trajectories is motivated by the desire to obtain a control law using only local model information. In this section, we actually use the approach out of necessity because no analytical solution is possible even in the centralized, full information case. The basis for the gradient method is the following proposition.

Proposition 4.5. *Given optimization problem (4.40), the gradient of the cost functional (4.40a) with respect to the feedback matrix $\hat{K}$ is*

$$\nabla_{\hat{K}} J = \int_{t_0}^{t_\text{f}} 2R\hat{K}\hat{x}(t)\hat{x}^\mathsf{T}(t) + B^\mathsf{T}\hat{\lambda}(t)\hat{x}^\mathsf{T}(t)\text{d}t, \tag{4.41}$$

where the adjoint states follow the dynamics

$$\dot{\hat{\lambda}}(t) = (\hat{A} - \hat{B}\hat{K})^\mathsf{T}\hat{\lambda}(t) - 2\hat{K}^\mathsf{T}R\hat{K}\hat{x}(t) - 2\hat{Q}\hat{x}(t) - 4\sum_{k=1}^{||E||}(\hat{x}^\mathsf{T}(t)\hat{q}_k\hat{Q}_k\hat{x}(t))\hat{q}_k\hat{Q}_k\hat{x}(t), \tag{4.42a}$$

$$\hat{\lambda}(t_\text{f}) = -2\hat{S}\hat{x}(t_\text{f}). \tag{4.42b}$$

Proof. The proof is similar to the proof of Proposition 4.1. The corresponding Lagrangian function of problem (4.40) is

$$\begin{aligned} L =& \hat{x}(t_\text{f})^\mathsf{T}\hat{S}\hat{x}(t_\text{f}) + \int_0^{t_\text{f}} \sum_{k=1}^{||E||}\left(\hat{x}^\mathsf{T}(t)\hat{q}_k\hat{Q}_k\hat{x}(t)\right)^2 + \hat{x}^\mathsf{T}\hat{Q}\hat{x} \\ &+ \hat{\lambda}^\mathsf{T}(t)(\dot{\hat{x}}(t) - (\hat{A} - \hat{B}\hat{K})\hat{x}(t)) + \hat{x}^\mathsf{T}(t)\hat{K}^\mathsf{T}R\hat{K}\hat{x}(t)\text{d}t + \hat{\mu}(\hat{x}(0) - \hat{x}_0). \end{aligned} \tag{4.43}$$

Partial integration of (4.43) gives

$$L = \hat{x}(t_\mathrm{f})^\mathsf{T}\hat{S}\hat{x}(t_\mathrm{f}) + \int_0^{t_\mathrm{f}} \sum_{k=1}^{||E||} \left(\hat{x}^\mathsf{T}(t)\hat{q}_k\hat{Q}_k\hat{x}(t)\right)^2 + \hat{x}^\mathsf{T}(t)\hat{K}^\mathsf{T}R\hat{K}\hat{x}(t) + \hat{x}^\mathsf{T}\hat{Q}\hat{x} - \hat{x}^\mathsf{T}(t)\dot{\hat{\lambda}}(t) - \hat{x}^\mathsf{T}(t)(\hat{A}-\hat{B}\hat{K})^\mathsf{T}\hat{\lambda}(t)\mathrm{d}t + \left[\hat{\lambda}(t)^\mathsf{T}\hat{x}(t)\right]_0^{t_\mathrm{f}} + \hat{\mu}(\hat{x}(0)-\hat{x}_0).$$

With this, we can derive the dynamics of the adjoint state through the optimality condition $\frac{\partial L}{\partial \hat{x}} = 0$ which eventually leads to (4.42). The gradient comes from the derivative of the Lagrangian L with respect to $\hat{K}$. This concludes the proof. □

Given the gradient from Proposition 4.5, the feedback matrix is iteratively determined using Algorithm 12. The choice of the step size γ_k is important for the speed of convergence. A possible choice is the BB step size (see Sections 4.4 and A.1.1) given by

$$\gamma_k = \frac{(\Delta\mathrm{vec}(\hat{K}))^\mathsf{T}(\Delta\mathrm{vec}(\hat{K}))}{(\Delta\mathrm{vec}(\hat{K}))^\mathsf{T}(\Delta\mathrm{vec}(\nabla_{\hat{K}}J))}, \tag{4.44}$$

where $\Delta\mathrm{vec}(\hat{K}) = \mathrm{vec}(K^{(k)}) - \mathrm{vec}(\hat{K}^{(k-1)})$ and $\Delta\mathrm{vec}(\nabla_{\hat{K}}J) = \mathrm{vec}(\nabla_{\hat{K}}J^{(k)}) - \mathrm{vec}(\nabla_{\hat{K}}J^{(k-1)})$. An alternative is to search for a suitable step size based on the Powell-Wolfe conditions (see Sections 4.4.3 and A.1.1) given by

$$J(\hat{K} + \gamma_k\mathrm{mat}(s_k)) - J(\hat{K}) \leq \gamma_k c_1(\mathrm{vec}(\nabla_{\hat{K}}J))^\mathsf{T}s_k \tag{4.45a}$$

$$(\mathrm{vec}(\nabla_{\hat{K}+\gamma_k s_k}J))^\mathsf{T}s_k \geq c_2(\mathrm{vec}(\nabla_{\hat{K}}J))^\mathsf{T}s_k, \tag{4.45b}$$

where $c_1 \in (0,1)$ and $c_2 \in (c_1,1)$. For Algorithm 12, the search direction s_k is given by the vectorization of the negative gradient, i.e. $s_k = -\mathrm{vec}(\nabla_{\hat{K}}J)$. If the BB step size is used, it also needs to satisfy the conditions (4.45) to ensure monotonous behavior.

In the next subsection, we replace the gradient method with an improvement to a Quasi-Newton method, but the approach is still based on the same general concept.

Algorithm 12 Iterative solution of (4.40) using gradient descent.

1. Simulate the states $\hat{x}(t)$ for the finite horizon t_f.
2. Simulate the adjoint states $\hat{\lambda}(t)$ for the same horizon according to (4.42).
3. Compute the gradient according to (4.41).
4. Update the feedback matrix

   $$\hat{K}^{(k+1)} = \hat{K}^{(k)} - \gamma_k\nabla_{\hat{K}}J^{(k)},$$

   where γ_k is a scalar step length.
5. If $\left|\frac{J^{(k)}-J^{(k-1)}}{J^{(k-1)}}\right| < \epsilon$, stop. Otherwise, increase k and go back to step 1.

4.7.2.2 Minimization via BFGS method

The downside of Algorithm 12 presented in the previous section is that gradient methods generally converge slowly. However, the availability of the gradient according to (4.41) allows us to use more advanced optimization methods, like the Quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, for details see Section A.1.2. Thus, instead of the negative gradient we use the following search direction

$$s_k = -H_k \mathrm{vec}(\nabla_{\hat{K}^{(k)}} J),$$

where H_k approximates the inverse of the Hessian matrix.

With this, we obtain Algorithm 13.

Remark 4.10. Despite the fact that the two algorithms should lead to the same control law with the same cost this is not always the case. We attribute this to the nonconvexity of the cost functional causing the algorithms to converge to different local minima caused by the differences in search directions and step sizes.

4.7.2.3 Averaging over the initial configuration x_0

One possible problem concerning the resulting feedback matrices of Algorithms 12 and 13 is that they are optimized with respect to one specific initial configuration x_0. In practice,

Algorithm 13 Iterative solution of (4.40) using BFGS method.

1. Choose $c_1, c_2 \in \mathbb{R}$, $\hat{K}^{(0)} \in \mathbb{R}^{m\times n}$. Pick a positive definite matrix $H_0 \in \mathbb{R}^{mn\times mn}$, e.g. $H_0 = I_{mn}$.
2. Compute the search direction s_k as
$$s_k = -H_k \mathrm{vec}(\nabla_{\hat{K}} J),$$
where the gradient $\nabla_{\hat{K}} J$ is given by to (4.5).
3. Compute the step size γ_k according to the Powell-Wolfe conditions (4.45).
4. Update the feedback matrix
$$\hat{K}^{(k+1)} = \hat{K}^{(k)} + \gamma_k \mathrm{mat}(s_k).$$
5. Set $p_k = \mathrm{vec}(\hat{K}^{(k+1)}) - \mathrm{vec}(\hat{K}^{(k)})$ and $q_k = \mathrm{vec}(\nabla_{\hat{K}} J(\hat{K}^{(k+1)})) - \mathrm{vec}(\nabla_{\hat{K}} J(\hat{K}^{(k)}))$. Update H_k as
$$H_{k+1} = H_k + \frac{(p_k - H_k q_k)p_k^\mathsf{T} + p_k(p_k - H_k q_k)^\mathsf{T}}{p_k^\mathsf{T} q_k} - \frac{(p_k - H_k q_k)^\mathsf{T} q_k}{(p_k^\mathsf{T} q_k)^2} p_k p_k^\mathsf{T}.$$
6. If $\left|\frac{J^{(k)} - J^{(k-1)}}{J^{(k-1)}}\right| < \epsilon$, stop. Otherwise, increase k and go back to step 1.

however, the initial configuration might not be known in advance or might be slightly disturbed from the assumed one. This problem was already encountered in Section 4.3 where a solution was presented. A similar approach is presented here in order to circumvent this problem. We propose to average over several initial configurations for the simulated trajectories to obtain a control law that performs well for an area. The algorithms principally remain unchanged except for the gradient which is now given by

$$\nabla_{\hat{K}} J = \frac{1}{n_{\text{samples}}} \left(\sum_{i=1}^{n_{\text{samples}}} \int_0^{t_\text{f}} 2R\hat{K}\hat{x}_i(t)\hat{x}_i^\top(t) + B^\top \hat{\lambda}_i(t)\hat{x}_i^\top(t) \mathrm{d}t \right),$$

where $\hat{x}_i(t)$ and $\hat{\lambda}_i(t)$ are the trajectories resulting from the ith initial configuration, and n_{samples} is the number of selected initial configurations, see Section 4.3 for more details.

Remark 4.11. While this extension enlarges the area of possible initial configurations, this does not lead to a globally optimal control law. In fact, our numerical investigations show that initial configurations that are not considered directly in the design may lead to undesirable performance. This is also later shown in the example in Section 4.7.3.1. Furthermore, notice that the averaging approach from Section 4.3 of using all unit vectors as initial condition is not completely suitable here because of the nonlinearity of the adjoint states caused by the biquadratic term. Therefore, the chosen approach of averaging several reasonable initial conditions is preferred.

4.7.3 Numerical results

In this subsection, we validate the control design algorithm using several numerical experiments. We provide an illustrating example, evaluate the respective performance of the two algorithms, and compare the presented approach with an open loop input optimization.

4.7.3.1 Illustrating numerical example

In this subsection, we illustrate the result of Algorithm 13 with averaging over the initial configuration x_0. The considered system consists of three physically interconnected robots with system dynamics described by (4.33), all having identical parameters $M_i = I_3$, $D_i = 2\sqrt{3}I_3$ and $K_i = 3I_3$. Because each subsystem has nine states, we have an overall system dimension of 27. As weighting matrices, we choose $R = 5I_{18}$, $\hat{S} = \operatorname{diag}(10I_{27}, 0)$, $\hat{q}_k = 100$ and $\hat{Q} = \operatorname{diag}(0.1I_{27}, 0)$ because we emphasize the maintenance of the formation. The optimization horizon is 40 seconds. The optimization algorithm stops when the change in cost between iterations is less than 10^{-3}. For the design, we select four different initial configurations. The coordinates of the center points of the respective triangles are given in Table 4.6 and are marked as gray crosses in Figure 4.13. In the simulations, the goal is to move the triangle formation of the three interconnected robots from four different initial configurations to a desired end point. These four initial configurations and the end point are also given in the table. Three of the initial configurations belong to the area surrounded by the four points used in the design, with two of them used directly in the design, while the fourth point is outside the area.

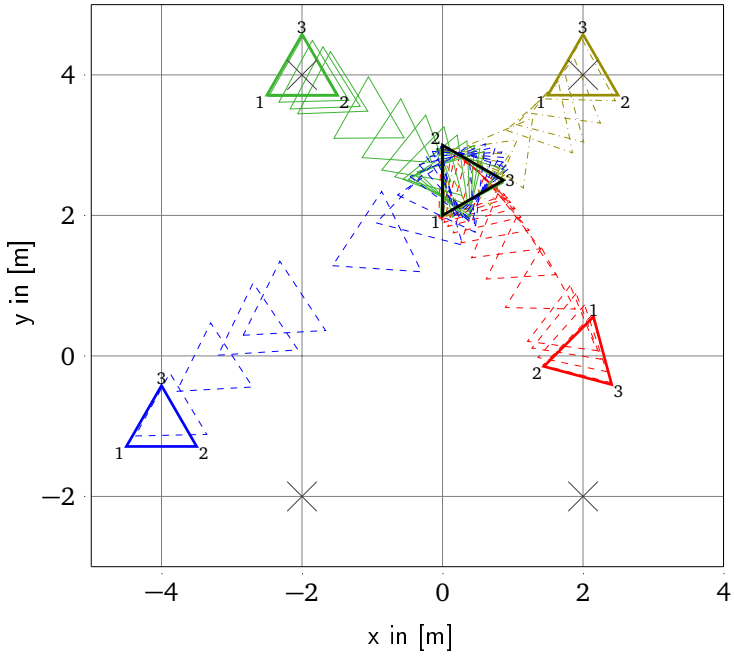


Figure 4.13: Three mobile robots drive from four different initial configurations (red dashed, olive dashed dotted, green solid, blue dashed) to a common goal while trying to maintain the formation. Bold colored triangles illustrate the initial robot configuration, the bold black triangle is the final configuration. The blue triangle clearly loses formation because the shape of the triangle stretches during the movement, while the other three triangles maintain their shape.

Figure 4.13 shows the resulting movements from all four initial configurations. We can see that the control design works well for all three points inside the area because the desired end point is reached and all the intermediate steps also show the initial formation. This shows that the relaxed rigidity condition is satisfied by the control design and the control law achieves all of its goals. For the point outside the area the desired end point is still reached, but it can clearly be seen that the formation is violated in the intermediate steps because the formation stretches in all directions. This illustrates that the optimized control law has a local character and is not guaranteed to work well away from the initial configurations used during the optimization.

Remark 4.12. A phenomenon we observed during our numerical investigations is that while the classical LQR problem is invariant to scaling in the cost functional, meaning that the control law for Q and R is identical as the control law for cQ and cR with $c > 0 \in \mathbb{R}$, this is not the case here. The reason is that the adjoint states depend nonlinearly on x as well as $\hat{Q}_k$.

4.7.3.2 Comparison between the two presented algorithms

In this subsection, we compare the computational performance between Algorithm 12 using the Barzilai-Borwein step size (4.44) and Algorithm 13. The system parameters are identical to the previous subsection. As a comparison scenario we move an initial triangle on the

Table 4.6: Starting and end points for visualizing example.

Phase	Center Point
Design	$x_{0,1} = [2, -2]$
Design	$x_{0,2} = [2, 4]$
Design	$x_{0,3} = [-2, 4]$
Design	$x_{0,4} = [-2, -2]$
Simulation	$x_{0,1} = [2, 0]$
Simulation	$x_{0,2} = [2, 4]$
Simulation	$x_{0,3} = [-2, 4]$
Simulation	$x_{0,4} = [-4, -1]$
Design & Simulation	$x_{\mathrm{f}} = (\frac{\sqrt{3}}{6}, 2.5)$

edges of the rectangle marked by the four crosses in Figure 4.13 in steps of length 0.5, resulting in 40 different starting points. The weighting matrices are chosen as $\hat{q}_k = 5$, $R = I_{18}$, $\hat{Q} = \mathrm{diag}(I_{27}, 0)$, $\hat{S} = \mathrm{diag}(I_{27}, 0)$ with a horizon of 40. The optimization algorithms stop when the change in cost between iterations is less than 10^{-3}. The results of the comparison are summarized in Table 4.7. We see that the number of iterations is comparable for both algorithms. The fact that the number of iterations is lower for the gradient method is counter-intuitive and might be due to the fact that different local minima are found by the two algorithms. We observed in our investigations that even though both algorithms may achieve almost comparable costs, the actual resulting control matrices might be completely different. While the gradient descent algorithm has advantages in the computation time, the BFGS algorithm always achieves lower costs. Note that for some categories we choose the median instead of the average because the BFGS algorithm produces some outliers that are not representative for its overall performance. In conclusion, if computational time is not an issue, the BFGS algorithm should be preferred because of the lower achieved cost.

4.7.3.3 Comparison with open loop input

In this subsection, we compare the performance of the resulting feedback matrix from Algorithm 13 with an open loop input obtained with the Matlab function `fminunc`. The test scenario is the movement of a formation of three robots with identical system parameters as in Section 4.7.3.1. The open loop case corresponds to the generation of desired trajectories which are tracked by the impedance control law. The shape is a triangle with an initial cen-

Table 4.7: Comparison between Algorithms 12 and 13.

	Algorithm 12	Algorithm 13
Median number of iterations	184	191
Average achieved cost	82.09	75.92
Median computation time [s]	59.1	142.4
Average cost decrease by BFGS	-	9.2%
Median cost decrease by BFGS	-	8.1%

ter point $(0,0)$. The desired end formation is a triangle with center point $(\frac{\sqrt{3}}{6}, 2.5)$ which is rotated by $-\frac{\pi}{2}$. The starting and end formations are shown in Figure 4.14. We choose the weighting matrices $\hat{q}_k = 40$, $R = 5I_{18}$, $\hat{S} = \text{diag}(10I_{27}, 0)$ and $\hat{Q} = \text{diag}(10^{-2}I_{27}, 0)$, with the horizon 20.

The presented Algorithm 13 leads to a feedback control law which achieves the cost 11.13 after 958 iterations and 1543 seconds of computation time. The open loop trajectory is able to achieve a lower cost of 7.92 which is to be expected by the additional degree of freedom in the input signal but the computation time is much higher (117021 seconds $\approx$ 1.35 days). In addition to a lower computation time, the feedback clearly has advantages when disturbances or uncertainties are considered.

4.7.4 Summary

In this section, we presented an extension of our control design method to optimal formation control. First, a new non-quadratic term was included into the cost functional which expresses a relaxed form of the rigidity requirement of the formation. Then, we applied the same idea as in the previous sections by introducing adjoint states and iteratively optimize the feedback law. In addition to the previously used gradient method, we also made use of a Quasi-Newton method. All results were illustrated and validated in numerical experiments. A full-scale robotic experiment was also completed, but the details are outside of the scope of this thesis. For details, we refer to [25].

4.8 Generalization of distributed control design approach

In this section, we present two generalizations of the distributed control design approach used throughout this chapter, namely the iterative optimization based on simulated trajec-

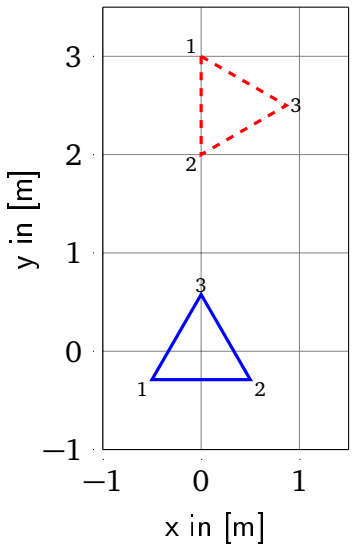


Figure 4.14: Initial condition (blue, solid) and end formation (red, dashed) for comparison between Algorithm 13 and the open loop trajectory.

tories. The first one is the generalization to a time-varying control law, the second one is to nonlinear system dynamics. Naturally, both generalizations can be combined, but we present them separately to allow for an easier understanding.

4.8.1 Time-varying control law

In this subsection, we essentially consider optimization problem (4.4), but we allow the control law to have the additional degree of freedom to be time-varying. Thus, the problem is

$$\min_{x,u} J(t,x,u) = x^{\mathsf{T}}(t+t_{\mathrm{f}})Sx(t+t_{\mathrm{f}}) + \int_{t}^{t+t_{\mathrm{f}}} x^{\mathsf{T}}(\tau)Qx(\tau) + u^{\mathsf{T}}(\tau)Ru(\tau)\mathrm{d}\tau, \tag{4.46a}$$

$$\text{s.t. } \dot{x}(\tau) = Ax(\tau) + Bu(\tau), \tag{4.46b}$$

$$u(\tau) = -K_{\mathrm{dist}}(\tau)x(\tau), \tag{4.46c}$$

$$K_{\mathrm{dist}} \text{ is stabilizing}, \tag{4.46d}$$

$$K_{\mathrm{dist}} \in \mathscr{K}. \tag{4.46e}$$

Based on the same ideas as before, we can state the following proposition.

Proposition 4.6. *Given optimization problem (4.46), the gradient of the cost functional (4.46a) with respect to the control law blocks $K_{\mathrm{dist},ij}(\tau)$ is given by*

$$(\nabla_{K_{\mathrm{dist}}} J)_{ij}(\tau) = -2R_i u_i(\tau)x_j^{\mathsf{T}}(\tau) + B_i^{\mathsf{T}}\lambda_i(\tau)x_j^{\mathsf{T}}(\tau), \tag{4.47}$$

where

$$\dot{\lambda}(\tau) = -A_K^{\mathsf{T}}\lambda(\tau) + 2(Q + K_{\mathrm{dist}}^{\mathsf{T}}(\tau)RK_{\mathrm{dist}}(\tau))x(\tau), \tag{4.48a}$$

$$\lambda(t+t_f) = -2Sx(t+t_{\mathrm{f}}),\ \tau \in [t, t+t_{\mathrm{f}}], \tag{4.48b}$$

with $A_K = A - BK_{\mathrm{dist}}(\tau)$.

Proof. The proof is identical to the proof of Proposition 4.1. □

The proposition essentially remains unchanged and the adjoint states are also the same, except that K_{dist} is time-varying. The only major difference is that the gradient itself is not an integral anymore, but that it is also, naturally, time-varying. What is noteworthy is that the simulation based design approach presented in this thesis also allows intermediate variants. Instead of having only a time-invariant control law, or a completely time-varying control law we can also take the wish into account of splitting the time horizon t_f into several parts where the control law is constant, or that the control law is time-varying only in parts of the horizon.

Remark 4.13. A remark is in order on the optimality of the time-invariant or the time-varying control law presented here and in Section 4.3.1. In Section 2.2.1, we saw that the full information optimal control law for a finite LQ problem is time-varying. However, given the sparsity structure of the problem it is actually an open question what form the optimal control law has. Therefore, we cannot be certain if a time-varying control law is superior to a time-invariant one or if, for example, a nonlinear feedback may be better than our assumed linear one. We compare the performance of the time-varying control law of this section with the time-invariant one later in Section 4.8.3 in a numerical experiment.

4.8.2 Nonlinear system dynamics and control law

In this subsection, we consider generalized forms of both the system dynamics and of the control law. The system dynamics are input affine nonlinear systems and the control law is a nonlinear control law with a known parameterization. The dynamics of each subsystem are

$$\dot{x}_i(t) = f_i(x_i(t), x_{j_1}(t), \dots, x_{j_k}(t)) + g_i(x_i(t), x_{j_1}(t), \dots, x_{j_k}(t))u_i(t),$$

where all $j_1, \dots, j_k \in \mathcal{N}_{\text{in},i}$ and where $f : \mathbb{R}^n \to \mathbb{R}^{n_i}$ and $g : \mathbb{R}^n \to \mathbb{R}^{n_i \times m_i}$. That means that we still consider an interconnected dynamical system with a sparsity structure, but the dynamics are now nonlinear. The overall dynamics are

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

where $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are obtained by stacking f_i and g_i. The control law is given by

$$u_i(t) = -k_{\text{dist},i}(x_i(t), x_{j_1}(t), \dots, x_{j_k}(t)),$$

where $k_{\text{dist},i} : \mathbb{R}^n \to \mathbb{R}^{m_i \times n}$ all $j_1, \dots, j_k \in \mathcal{N}_i$, i.e. the control graph is the undirected system graph. Furthermore, the function $k_{\text{dist},i}$ is parameterized by the parameters $\{k_{i,1}, \dots, k_{i,p_i}\}$ where p_i is the number of parameters of the control law of subsystem i.

The optimal control problem is

$$\min_{x,u} J(t,x,u) = s(x(t+t_\text{f})) \int_t^{t+t_\text{f}} q(x(\tau)) + r(u(\tau))\mathrm{d}\tau, \tag{4.49a}$$

$$\text{s.t. } \dot{x}(\tau) = f(x(\tau)) + g(x(\tau))u(\tau), \tag{4.49b}$$

$$u(\tau) = -k_\text{dist}(x(\tau)), \tag{4.49c}$$

$$k_\text{dist} \in \mathcal{K}. \tag{4.49d}$$

Again, using the Lagrangian of the problem, we can derive adjoint states and the gradient and arrive at the following proposition.

Proposition 4.7. *Given optimization problem (4.49), the gradient of the cost functional (4.49a) with respect to the control law parameter $k_{i,j}$ is given by*

$$(\nabla_{k_{i,j}} J) = \int_t^{t+t_\text{f}} \frac{\partial r(-k(x(\tau)))}{\partial k_{i,j}} + g(x(\tau))^\mathsf{T} \lambda(\tau) \frac{\partial k(x(\tau))}{\partial k_{i,j}}^\mathsf{T} \mathrm{d}\tau, \tag{4.50}$$

where

$$\dot{\lambda}(\tau) = -\left[\frac{\partial f(x(\tau))}{\partial x(\tau)} - \frac{\partial g(x(\tau))}{\partial x(\tau)} k_\text{dist}(x(\tau)) - g(x(\tau)) \frac{\partial k_\text{dist}(x(\tau))}{\partial x(\tau)}\right]^\mathsf{T} \lambda(\tau) + \frac{\partial q(x(\tau))}{\partial x(\tau)} + \frac{\partial r(-k_\text{dist}(x(\tau)))}{\partial x(\tau)}, \tag{4.51a}$$

$$\lambda(t+t_\text{f}) = -\frac{\partial s(x(t+t_\text{f}))}{\partial x(t+t_\text{f})}, \ \tau \in [t, t+t_\text{f}]. \tag{4.51b}$$

Proof. The proof follows in the same fashion as the proof of Proposition 4.1. We construct the Lagrangian of the optimization problem and then require that $\frac{\partial L}{\partial x} = 0$ to derive the adjoint states. The gradient also results from the Lagrangian. □

It can be easily verified that the result of Proposition 4.1 is recovered from Proposition 4.7 when the special case of LTI dynamics and the linear control law is considered.

Remark 4.14. The results in this subsection serve as an illustration of the overall applicability of the iterative design approach based on simulated trajectories. In principle, completely nonlinear cost functionals, system dynamics and control laws can be addressed as long as the control law is parameterized with known parameters. A typical approach for a nonlinear control law with a known parameterization is a polynomial control law. In practice, however, there is no guarantee that the approach will give acceptable results. Even for LTI systems the design is non-convex, but for nonlinear dynamics the form of the optimization problem becomes even more complicated such that the presented gradient approach can give unsatisfactory results.

4.8.3 Numerical evaluation

In this subsection, we evaluate the performance of the time-varying control law in comparison with the time-invariant result from Section 4.3. As stated in Remark 4.13, it is unclear for structured control laws whether the time-varying control law is indeed better in terms of performance. To compare the two control laws, we consider the same test systems that we used in Section 4.4.5, i.e. we first look at 100 stable test systems where each system has $N = 20$ subsystems and the total number of states is $n = 40$. Then, we look at the same number of unstable test systems.

4.8.3.1 Stable test systems

We compare the performance of the result of Algorithm 8 with the time-invariant and with the time-varying control law. Because we expect that performance advantages of one of the two may depend on the optimization horizon t_f, we solve the problem for the following horizons: 0.01, 0.05, 0.1, 0.5, 1, 5, 10. The evaluation of the resulting costs is summarized in Table 4.8.

We observe that the time-varying control law has a slightly better performance for short horizons, but the advantage is negligible. However, the longer the horizon gets, the better the time-invariant control law becomes in comparison to the time-varying one. We attribute this

Table 4.8: Performance comparison of time-invariant (TI) and time-varying (TV) control law based on relative cost difference $\frac{J_\mathrm{TI}-J_\mathrm{TV}}{J_\mathrm{TV}}$ for stable test systems.

Horizon t_f	0.01	0.05	0.1	0.5	1	5	10
Average $\frac{J_\mathrm{TI}-J_\mathrm{TV}}{J_\mathrm{TV}}$ [%]	$9.79 \cdot 10^{-4}$	0.019	0.067	0.55	0.64	-0.95	-2.38
Maximum $\frac{J_\mathrm{TI}-J_\mathrm{TV}}{J_\mathrm{TV}}$ [%]	0.0016	0.031	0.11	0.82	1.07	0.99	-0.73
Minimum $\frac{J_\mathrm{TI}-J_\mathrm{TV}}{J_\mathrm{TV}}$ [%]	$5.45 \cdot 10^{-4}$	0.011	0.037	0.24	0.26	-2.12	-4.71

result to the non-convexity of the problem because, in principle, the time-invariant control law is just a special case of the time-varying one and should not be able to outperform the more general case. However, numerical issues and local minima can lead to this result. Future work should investigate the optimal solution of the structured control design further.

4.8.3.2 Unstable test systems

We perform the same analysis for 100 unstable test systems for the same horizons as in the stable case. The results are summarized in Table 4.9. For unstable systems, the trend seems to be that the time-varying control law outperforms the time-invariant one for every horizon except for the last one. This result is more in accordance with intuition but still not fully conclusive. Furthermore, a disadvantage of the time-varying control law is the increased storage capacity required for the control law.

4.8.4 Summary

In this section, we developed two generalizations of the control design approach presented throughout this chapter. The first generalization is the time-varying control law, the second addresses nonlinear dynamics, cost functionals and control laws. Numerical investigations showed that there is an advantage of the time-varying control law in some cases but not in all. Future work on the optimal form of the control law is required for an ultimate judgment.

4.9 Chapter summary

In this chapter, we developed and analyzed a distributed optimal control design method that relies only on local model information. The problem formulation is similar to the standard LQR problem but is complicated by the aim to design the control law without a central entity, and also by the desired structure in the control law. The design method is based on deriving adjoint dynamics specific to the optimal control problem. Then, simulated trajectories of the states and adjoint states are used to distributedly compute a gradient of the cost functional with respect to the feedback matrix. This gradient is then used to iteratively optimize the feedback law. Because it is based on simulated trajectories, the design approach can only consider finite horizon cost functionals. Therefore, we needed to use a specific terminal cost term to guarantee stability of the closed loop. A distributed computation method to compute this terminal cost term was also presented, and it makes use of the techniques used

Table 4.9: Performance comparison of time-invariant (TI) and time-varying (TV) control law based on relative cost difference $\frac{J_{\mathrm{TI}}-J_{\mathrm{TV}}}{J_{\mathrm{TV}}}$ for unstable test systems.

Horizon t_{f}	0.01	0.05	0.1	0.5	1	5	10
Average $\frac{J_{\mathrm{TI}}-J_{\mathrm{TV}}}{J_{\mathrm{TV}}}$ [%]	$9.77 \cdot 10^{-4}$	0.021	0.078	1.25	3.69	3.51	-1.03
Maximum $\frac{J_{\mathrm{TI}}-J_{\mathrm{TV}}}{J_{\mathrm{TV}}}$ [%]	0.0015	0.034	0.13	2.23	16.78	25.09	2.00
Minimum $\frac{J_{\mathrm{TI}}-J_{\mathrm{TV}}}{J_{\mathrm{TV}}}$ [%]	$4.85 \cdot 10^{-4}$	0.011	0.041	0.71	1.29	-1.12	-2.22

in Chapter 3 to solve a Lyapunov inequality. The effectiveness of the terminal cost term and of the gradient algorithm was shown in numerical simulations.

Considerable effort was put into the complete distribution of the presented gradient algorithm and we presented methods to distributedly decide on a step size and how to guarantee convergence. In particular, we employed a consensus algorithm to compute the Barzilai-Borwein step size in a distributed fashion. Also, a conjugate gradient method was derived that makes use of the same consensus techniques. The performance of the different step size and descent methods was evaluated in a numerical experiment. The BB step size and the conjugate gradient method both perform well and no ultimate judgment on the better alternative could be made.

Because the approach relies only on local model information and local information exchange, it requires a large communication effort to obtain an optimal control law. One particular problem is the communication required for the trajectory simulation to obtain the gradient. We proposed a new event-based Euler method to reduce this communication effort and its positive effect is illustrated in numerical fashion.

Furthermore, we extended the control design method using adjoint states to two problem classes beyond the standard LQR problem formulation. The first one is the case of DAE systems where the control design method also only requires local model information. The resulting control law makes explicit use of algebraic states. The second one is the application of the approach to an optimal formation control problem where the rigidity requirement is reformulated into a biquadratic form. The biquadratic form does not allow an analytic solution so the approach of using adjoint states and simulated trajectories was especially useful.

Last, we presented two generalizations of the form of the control law and their corresponding gradient method, namely a time-varying structure and a nonlinear one for nonlinear systems.

Note that the results of this chapter are partially based on [21, 22, 24, 25, 26].

5

Decay analysis in interconnected systems

In this chapter, we address the third and last aspect of this thesis: the decay analysis in interconnected systems. Specifically, we analyze how an input or a disturbance at one node spreads throughout a network of subsystems and how far the effect of the signal is noticeable. We abandon the notion of performing the analysis from a distributed perspective, but the results of this chapter give further insight into the behavior of interconnected systems. The main contribution of this chapter is an analytical investigation of the decay between subsystems in response to a steady state disturbance. We consider the case of a constant input at a single node and then look at the generalization to a sinusoidal input. The results are illustrated using numerical simulations. In order to obtain meaningful results, certain assumptions on the system parameters need to be made. Therefore, *we consider stable systems whose stability can be shown using a vector Lyapunov function*. The concept and the idea of vector Lyapunov functions is summarized well in [15] and we briefly introduced them in Section 2.1. In addition, we already made use of vector Lyapunov functions in Chapter 3. We think that this assumption is reasonable on several grounds. First, the analysis in this type of setup is known to remain valid even when interaction gains change, a property which makes the results more robust. Second, it allows the analysis of nonlinear large-scale systems using linear system tools. Last, it is an old and in a sense tested theory which has been rigorously evaluated with many established results.

The main idea behind our analysis is that, instead of analyzing the original system dynamics, we use a comparison system that approximates every multi-dimensional subsystem by a scalar representation. This approach is motivated by our interest in the overall interaction between the subsystems along the system graph and not in the detailed behavior of each state of every subsystem.

The remainder of this chapter is organized as follows. Section 5.1 contains the problem statement. An overview on related work is given in Section 5.2. Then, we explain the general

approach and give preliminary information in Section 5.3. The main analysis is presented in Section 5.4, and these results are illustrated in Section 5.5. We conclude with a summary in Section 5.6.

5.1 Problem formulation

In this chapter, we consider an interconnected dynamical system with a structure described by a graph as shown in Figure 5.1.

The original system dynamics are of the standard LTI form already considered in the previous chapters. Thus, every subsystem follows

$$\dot{x}_i(t) = A_{ii}x_i(t) + \sum_{\substack{j=1 \\ j\neq i}}^{N} A_{ij}x_j(t),\ i = 1,...,N, \tag{5.1}$$

and the overall system is

$$\dot{x} = Ax. \tag{5.2}$$

In the context of this chapter, we do not try to achieve a distributed analysis method so the only graph of interest is the system graph $\mathcal{G}_s$. However, to make the notation more intuitive in terms of the analysis in this chapter, we consider the transpose system graph $\mathcal{G}_s^T = (\mathcal{V}_s, \mathcal{E}_s^T)$ instead of the original system graph $\mathcal{G}_s$. Hence, there is an edge in the directed graph $\mathcal{G}_s^T$ with $(i,j) \in \mathcal{E}_s^T$ iff $A_{ij} \neq 0$. This can be interpreted as subsystem i accessing j to update its state. We denote the set of neighboring nodes of node i, i.e. nodes that affect it, by $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}_s^T\}$.

The problem raised in this chapter can be summarized as follows.

Problem 3. Given system (5.1),(5.2), how does an input signal at one node propagate through the network with increasing distance to the source, and under what conditions is there a decay property?

5.2 Related work

Despite the fact that the question of signal decay in interconnected systems is practically relevant and far from trivial, we are not aware of many previous results in the literature that address it directly. However, the decay property for constant disturbances considered here turns out to be related to a decay property of entries of the inverse of a matrix whose structure captures that of the underlying system graph with a path graph corresponding to a tridiagonal matrix. In [124], a decay property along the rows and columns of the inverse is developed for a *symmetric* tridiagonal matrix. In this chapter, a similar result is derived but for much more general matrix structures. Also related are the results in [125] where the

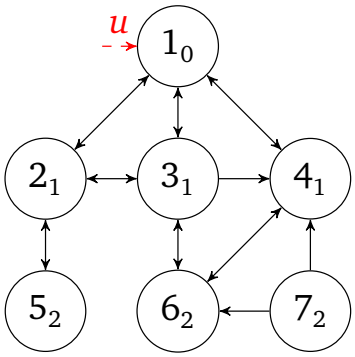


Figure 5.1: Example graph. The number denotes the nodes, the subscripted index denotes the distance to the source node. Note that the input is not an edge of the graph and does not follow the edge direction convention of the rest of the graph used throughout this chapter.

authors investigate properties of M-matrices in the context of linear systems of equations, and the sensitivity of the solution to changes in the entries of the M-matrix. The authors in [126] look at vehicle formations in a regular lattice structure, and investigate how local feedback laws can perform under stochastic disturbances. This is related to the question at hand because the considered signal can represent a disturbance. For consensus dynamics, the results in [127] provide an online optimization algorithm that changes interconnection weights in a graph to achieve an optimal disturbance rejection. For the special case of directed lattices with leader-follower-dynamics, the authors in [128] derive transfer functions from disturbances to the states of nodes. This approach goes in a similar direction as the work here, we, however, consider more general dynamics and graphs. Also, a similar research direction is taken in [129] where a local average consensus algorithm is developed, which involves both temporal and decaying spatial behavior. Our goal is to take that viewpoint of considering both temporal and spatial behavior as well, and investigate how physical signals propagate spatially through a dynamical system and whether the gain over a multihop path can be bounded in terms of the hop count.

5.3 Preliminaries and assumptions

In this section, we state preliminary assumptions and explain the principal idea of the presented analysis method. As mentioned earlier, we are interested in the overall influence one subsystem node has on another. This means that we are not necessarily interested in the behavior of individual states of each subsystem but rather in the total subsystem interaction. This leads to our first main idea: Instead of considering the original system dynamics (5.2) directly, we investigate the behavior of a comparison system where the multidimensional subsystem states $x_i \in \mathbb{R}^{n_i}$ are aggregated into scalar *superstates* $w_i \in \mathbb{R}$. This is illustrated in Figure 5.2.

One intuitive method, and our choice for the comparison system here, is to use vector Lyapunov functions that we have already encountered in the first distributed stability test in Section 3.3.1. There are two reasons for this choice. First, a decay analysis is only meaningful

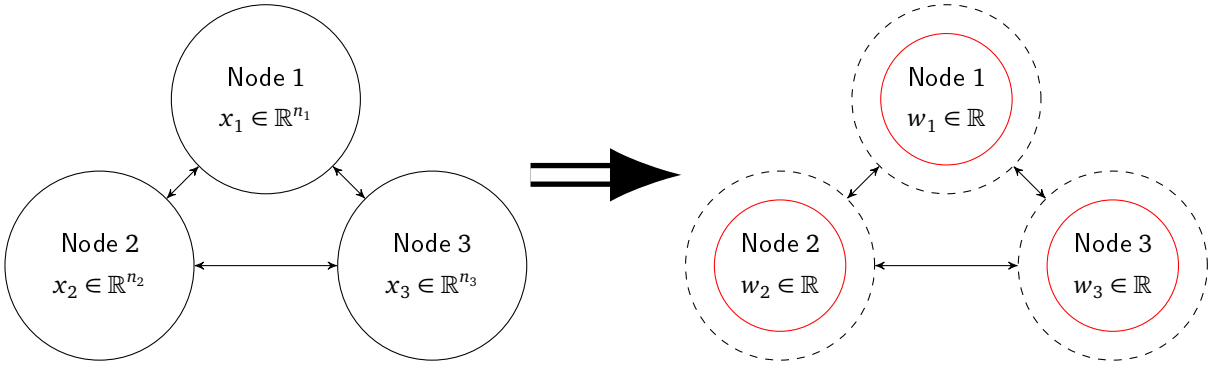


Figure 5.2: Subsystem states x_i are aggregated into scalar superstates w_i which correspond to a comparison system.

for stable systems and the vector Lyapunov function is a well-known tool to establish stability of interconnected systems. Second, the individual subsystem Lyapunov functions provide a weighted norm of the state, which is an appropriate property for a superstate. In order to further improve the understanding of the following analysis, we delve deeper into the details of vector Lyapunov functions. To achieve this, we provide additional information that go beyond the description in Section 2.1. We start by giving an extension of Lemma 2.1 with an additional aspect required only in this chapter.

Lemma 5.1. *[130, §6.2] A matrix $W \in \mathbb{R}^{N\times N}$ is a non-singular M-matrix if it has positive diagonal elements, nonpositive off-diagonal elements and if it satisfies the following equivalent conditions:*

- *There exists a vector $d \in \mathbb{R}^N_{++}$ such that $Wd \in \mathbb{R}^N_{++}$.*
- *There exists a positive diagonal matrix $D = \operatorname{diag}(d_1,\dots,d_N) \in \mathbb{S}^N_{++}$ such that WD is strictly diagonally dominant, i.e. $d_i W_{ii} > \sum_{j\neq i} d_j |W_{ij}| \; \forall i = 1,\dots,N$.*

Furthermore, Assumption 2.1 holds in this chapter, that is we assume that the isolated, decoupled subsystems described by

$$\dot{x}_i = A_{ii}x_i \tag{5.3}$$

are asymptotically stable. Hence, for any $Q_i \succ 0$, there is a $P_i \succ 0$ such that

$$P_iA_{ii} + A_{ii}^\mathsf{T}P_i = -Q_i, \tag{5.4}$$

and then clearly

$$v_i = (x_i^\mathsf{T}P_ix_i)^{\frac{1}{2}} \tag{5.5}$$

is a local Lyapunov function for the isolated subsystem i. Thus, with (5.4), we have that $\dot{v}_{i,\text{iso}} = -\frac{1}{2}(x_i^\mathsf{T}P_ix_i)^{-\frac{1}{2}}x_i^\mathsf{T}Q_ix_i < 0$ is the isolated derivative of v_i along the dynamics (5.3).

With the following bounds

$$\begin{aligned}
\sqrt{\lambda_m(P_i)}||x_i|| \leq v_i(x_i) \leq \sqrt{\lambda_M(P_i)}||x_i||,\\
\left\|\frac{\partial v_i}{\partial x_i}\right\| = \left\|(x_i^\mathsf{T}P_ix_i)^{-\frac{1}{2}}P_ix_i\right\| \leq \frac{\lambda_M(P_i)}{\sqrt{\lambda_m(P_i)}},\\
\left\|A_{ij}x_j\right\| \leq \sigma_M(A_{ij})||x_j||,
\end{aligned}$$

we can write the coupled derivative of v_i as

$$\begin{aligned}\dot{v}_i &= \dot{v}_{i,\text{iso}} + (x_i^\mathsf{T} P_i x_i)^{-\frac{1}{2}} x_i^\mathsf{T} P_i \sum_{j\in\mathcal{N}_i}^{N} A_{ij} x_j \\ &\leq -\frac{\lambda_m(Q_i)}{2\sqrt{\lambda_M(P_i)}}||x_i|| + \sum_{\substack{j=1\\ j\neq i}}^{N} \frac{\lambda_M(P_i)\sigma_M(A_{ij})}{\sqrt{\lambda_m(P_i)}}||x_j|| \\ &\leq -\tilde{\alpha}_{i,i}\, v_i + \sum_{j\in\mathcal{N}_i}^{N} \tilde{\alpha}_{i,j}\, v_j, \end{aligned} \tag{5.6}$$

where $\tilde{\alpha}_{i,i} = \frac{\lambda_m(Q_i)}{2\lambda_M(P_i)}$, $\tilde{\alpha}_{i,j} = \frac{\lambda_M(P_i)\sigma_M(A_{ij})}{\sqrt{\lambda_m(P_i)}\sqrt{\lambda_m(P_j)}}$. With this, we can state a modified form of Theorem 2.1.

Theorem 5.1. *[131, 15] Given Assumption 2.1, consider system (5.1),(5.2) and the local Lyapunov functions (5.5) with P_i and Q_{ii} from (5.4). If the matrix $-\tilde{\mathscr{A}}$ with elements from (5.6)*

$$\tilde{\mathscr{A}}_{ij} = \begin{cases} -\tilde{\alpha}_{i,i} & \text{if } i = j \\ \tilde{\alpha}_{i,j} & \text{if } i \neq j \end{cases}$$

is a non-singular M-matrix, then system (5.1),(5.2) is stable and $V(x) = d^\mathsf{T} v(x)$, where $d = [d_1, \ldots, d_N]^\mathsf{T} > 0$ and $v = [v_1(x_1), \ldots, v_N(x_N)]^\mathsf{T}$, is a Lyapunov function for the system. Furthermore, we call v with $\dot{v} \leq \tilde{\mathscr{A}} v$ a vector Lyapunov function of system (5.2).

There are several other ways to set up a vector Lyapunov function to show stability, but since stability investigations are not the focus of this chapter, we refer the reader to [15, §2] for more details.

Remark 5.1. Note that due to the use of vector Lyapunov function theory, the results in this chapter are extendable to a wide class of nonlinear systems. Necessary assumptions are the existence of constants $c_{1,i} > 0, c_{2,i} > 0$ such that $c_{1,i}||x_i|| \leq v_i \leq c_{2,i}||x_i||$, and the interconnection between subsystems needs to be bounded. To allow for an easier understanding, we restrict our attention in this chapter to linear systems.

In the following, we use the dynamics of v_i as the comparison system in order to investigate signal decay in the interconnected system. An illustration of the bound that the vector Lyapunov function v provides is shown in Figure 5.3.

Based on the comparison principle [131, §1.4], instead of the inequality (5.6), we consider $\tilde{w}(t)$ with $\tilde{w}(0) = v(0)$ and

$$\dot{\tilde{w}}_i = -\tilde{\alpha}_{i,i}\tilde{w}_i + \sum_{j\in\mathcal{N}_i}^{N} \tilde{\alpha}_{i,j}\tilde{w}_j, \tag{5.7}$$

$$\text{or} \quad \dot{\tilde{w}} = \tilde{\mathscr{A}}\tilde{w}, \tag{5.8}$$

where $\tilde{w} \in \mathbb{R}^N$ and $\tilde{\mathscr{A}} \in \mathbb{R}^{N\times N}$. Here, it is clear that $v_i(t) \leq \tilde{w}_i(t)$. Thus, the system (5.7) in a limited and special way approximates system (5.1) as a scalar system, and we can consider $\tilde{w}(t)$ as a super-state for the actual system state $x(t)$. In particular, we will use it to make estimates on the signal decay in interconnected systems. As already mentioned in the introductory remarks of this chapter, we make the following assumption on system (5.8).

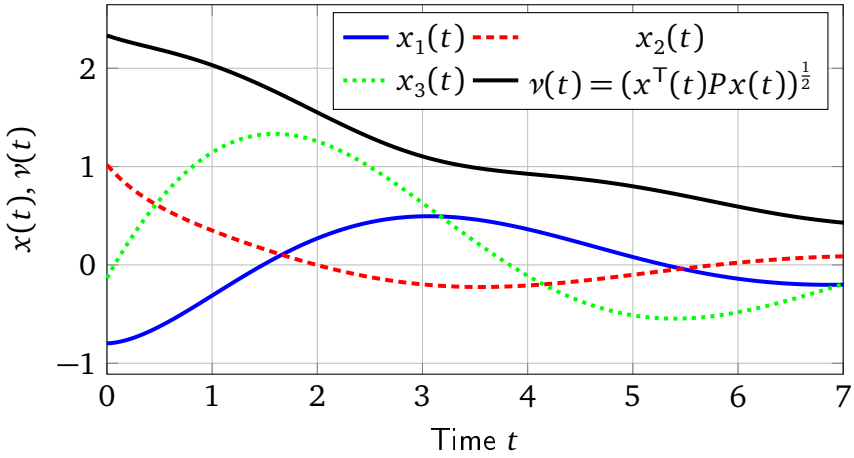


Figure 5.3: Vector Lyapunov function serves as a superstate to a multidimensional system.

Assumption 5.1. $-\tilde{\mathscr{A}}$ is a non-singular M-matrix.

Then, there is a positive diagonal matrix D such that the matrix $\mathscr{A} := D^{-1}\tilde{\mathscr{A}}D$ with elements α_{ij} is row diagonally dominant [130], i.e.

$$\alpha_{i,i} - \sum_{j \in \mathscr{N}_i} \alpha_{i,j} \geq 0 \; \forall \, i.$$

This also implies that the following condition holds

$$\alpha_{i,i} - \frac{\sum_{j \in \mathscr{N}_i} \alpha_{i,j}}{1-\gamma} \geq 0 \; \forall \, i, \tag{5.9}$$

for a known $0 < \gamma < 1$ where equality holds for at least one i.

Example 5.1. Consider the matrix

$$\mathscr{A} = \begin{bmatrix} -2 & 1 \\ 2 & -3 \end{bmatrix}.$$

From the first row, γ would be $\frac{1}{2}$ while it would be $\frac{1}{3}$ from the second row. The overall value of γ is then the minimum of them all, in this case $\frac{1}{3}$, such that equality of (5.9) holds for the second row, and is only a bound for the first row.

Hence, γ can be seen as a measure of how row dominant the matrix $\mathscr{A}$ is. When the original system does not satisfy Assumption 5.1, the authors in [132] present a system class for which decentralized controllers can be designed such that Assumption 5.1 is satisfied. The identification of further system classes where distributed controllers lead to Assumption 5.1 being satisfied is future work.

As a side note: Using (5.9) and Gershgorin arguments, the eigenvalues of $\mathscr{A}$ can easily be bounded. Finally, the transformed system dynamics according to Assumption 5.1 are

$$\dot{w} = \mathscr{A} w. \tag{5.10}$$

In the next section, we begin our decay analysis based on the comparison system dynamics (5.10).

5.4 Decay analysis

In this section, we present the main analytical results, namely the investigation of a way to describe the steady state of system (5.10) for two cases: (1) The system has a scalar constant input at a single node. (2) The system has a scalar sinusoidal input at a single node. In this way, we aim at developing an insight as to how a physical signal propagates spatially from system to system.

5.4.1 Steady state decay with constant input

In this subsection, we investigate how the steady state of an interconnected system behaves when there is one node with a constant and positive external input and the remaining nodes are not excited. The system dynamics (5.10) are extended to include the input and we obtain

$$\dot{w} = \mathscr{A} w + bu, \tag{5.11}$$

where $b \in \mathbb{R}^N$ is a vector consisting of only one nonzero entry, namely a one, and for convenience of derivation and presentation $u \in \mathbb{R}_{++}$. These are the system dynamics considered throughout the rest of the chapter. Note that the graphical structure of $\mathscr{A}$ is identical to the graph $\mathscr{G}_s^{\mathsf{T}}$, i.e. there is an edge from node i to node j when i uses information from j to update its state, in other words when i is influenced by j.

Remark 5.2. A few remarks about the superstate dynamics (5.11) are in order. The superstate represents an upper bound to a scaled vector Lyapunov function. In relation to the original system, it serves as a bound on a scaled and weighted norm of the original system state. The system dynamics represent a positive system because the matrix $\mathscr{A}$ is a Metzler matrix. Last, as the setup is linear, by the superposition principle the results are straightforward to expand to multiple or negative inputs. Also, since only the limit case $t \to \infty$ is considered, the results also cover inputs that become constant after a finite time.

In the following, we consider the steady state of w, which we denote by $\bar{w}$, i.e. $\lim_{t\to\infty} w(t) = \bar{w}$. Therefore, we can set the left hand side of (5.11) to zero. Next, we introduce the main concept of this chapter that we use throughout the analysis.

Definition 5.1. Let $\bar{w}_i \in \mathbb{R}$, $\bar{w}_j \in \mathbb{R}$ for $j \in \{j_1, \ldots, j_l\}$ where $i \notin \{j_1, \ldots, j_l\}$, and $0 < \gamma < 1$ from (5.9). If there exist $\beta_{i,j}$ for $j \in \{j_1, \ldots, j_l\}$ such that

$$\bar{w}_i = \sum_{k=j_1}^{j_l} \beta_{i,k} \bar{w}_k, \tag{5.12}$$

where $\sum_{k=j_1}^{j_l} \beta_{i,k} < 1-\gamma$ and $0 \leq \beta_{i,j} < 1-\gamma$ for all $j \in \{j_1, \ldots, j_l\}$, then $\bar{w}_i$ is termed a *steady state conic combination (SSCC)* of the steady states $\bar{w}_j$ with $j \in \{j_1, \ldots, j_l\}$. This is written as $\bar{w}_i \in C(\bar{w}_{j_1}, \ldots, \bar{w}_{j_l})$.

The notion of a steady state conic combination (SSCC) is illustrated in Figure 5.4.

Note that $C(\ldots)$ is a special case of a conic combination of steady state values. With this notion of SSCC, it can be shown that if $\bar{w}_i$ is an SSCC of the steady states of some

Figure 5.4: Illustration of steady state conic combination (SSCC): $\bar{w}_i \in C(\bar{w}_{j_1}, \ldots, \bar{w}_{j_l})$ can be larger than some $\bar{w}_j$ but is bounded by at least one $\bar{w}_j$ with $j \in \{j_1, \ldots, j_l\}$.

nodes $j_1, \ldots, j_l$, and if the steady state of one of those nodes $\bar{w}_{j_l}$ is an SSCC of the steady states of i and other nodes $k_1, \ldots, k_m$, then $\bar{w}_{j_l}$ can be replaced in the SSCC of $\bar{w}_i$ by the steady states of nodes $k_1, \ldots, k_m$. We summarize this more formally in the following lemma.

Lemma 5.2.
If $\bar{w}_i \in C(\bar{w}_{j_1}, \ldots, \bar{w}_{j_l})$ and $\bar{w}_{j_l} \in C(\bar{w}_i, \bar{w}_{k_1}, \ldots, \bar{w}_{k_m})$, then $\bar{w}_i \in C(\bar{w}_{j_1}, \ldots, \bar{w}_{j_{l-1}}, \bar{w}_{k_1}, \ldots, \bar{w}_{k_m})$ for $i \notin \{j_1, \ldots, j_l\}$.

Proof. Using (5.12), we start with

$$\bar{w}_i = \beta_{i,j_1}\bar{w}_{j_1} + \ldots + \beta_{i,j_{l-1}}\bar{w}_{j_{l-1}} + \beta_{i,j_l}(\beta_{j_l,i}\bar{w}_i + \beta_{j_l,k_1}\bar{w}_{k_1} + \ldots + \beta_{j_l,k_m}\bar{w}_{k_m})$$

and obtain

$$\bar{w}_i = \frac{\beta_{i,j_1}}{1-\beta_{i,j_l}\beta_{j_l,i}}\bar{w}_{j_1} + \ldots + \frac{\beta_{i,j_{l-1}}}{1-\beta_{i,j_l}\beta_{j_l,i}}\bar{w}_{j_{l-1}} + \frac{\beta_{i,j_l}\beta_{j_l,k_1}}{1-\beta_{i,j_l}\beta_{j_l,i}}\bar{w}_{k_1} + \ldots + \frac{\beta_{i,j_l}\beta_{j_l,k_m}}{1-\beta_{i,j_l}\beta_{j_l,i}}\bar{w}_{k_m}. \tag{5.13}$$

With the conditions of the lemma and (5.12), one can see that all coefficients on the right in (5.13) and their sum are smaller than $1-\gamma$, i.e.

$$\frac{1}{1-\beta_{i,j_l}\beta_{j_l,i}}\left(\sum_{r=1}^{l-1}\beta_{i,j_r} + \beta_{i,j_l}\sum_{s=1}^{m}\beta_{j_l,k_s}\right) < 1-\gamma,$$
$$\frac{\beta_{i,j_r}}{1-\beta_{i,j_l}\beta_{j_l,i}} < 1-\gamma, \quad \frac{\beta_{i,j_l}\beta_{j_l,k_s}}{1-\beta_{i,j_l}\beta_{j_l,i}} < 1-\gamma,$$

for $1 \le r \le l-1$ and $1 \le s \le m$. Notice that based on Definition 5.1, the lemma also subsumes the case where we only have $\bar{w}_{j_l} \in C(\bar{w}_{k_1}, \ldots, \bar{w}_{k_m})$ instead of $\bar{w}_{j_l} \in C(\bar{w}_i, \bar{w}_{k_1}, \ldots, \bar{w}_{k_m})$. The reason is that this is equivalent to setting $\beta_{j_l,i}$ to zero, which only decreases the coefficients in the right hand side of equation (5.13). This concludes the proof. □

To state the results in a precise fashion, we state some additional definitions before we come to the main result of this chapter.

Definition 5.2. [83] A *directed path* in the directed graph $\mathcal{G}_s^{\mathsf{T}}$ is a sequence $v_0, e_1, v_1, \ldots, e_n, v_r$ such that $v_i \in \mathcal{V}_s$ and $v_i \neq v_j$ for all $0 \le i, j \le r$, and such that $e_i \in \mathcal{E}_s^{\mathsf{T}}$ is a directed edge from v_{i-1} to v_i for every $1 \le i \le r$.

For example, in Figure 5.1, the node sequence $(7, 4, 3)$ with its corresponding edges does not constitute a directed path, but the node sequence $(7, 6, 3)$ does.

Definition 5.3. Given a designated source node in the interconnection graph $\mathcal{G}_s^{\mathsf{T}}$ associated with the system dynamics (5.11), the *distance* of a node is the least number of directed edges across all paths between the node and the source node.

Definition 5.4. Given a designated source node in the interconnection graph $\mathcal{G}_s^\mathsf{T}$ associated with the system dynamics (5.11), and given a node v with distance d from the source node, the *unique* $(d-1)$*-paths* are those paths starting at v and ending at a node with distance $(d-1)$ without containing any other node with distance $(d-1)$. The set of nodes that can be reached via a unique $(d-1)$-path from node v are called the *unique* $(d-1)$*-nodes* and is denoted by $\mathcal{V}_{v,d-1}$.

Seen from a different point of view, if there is a path from v_1 with distance d to the source node, and that path has only strictly decreasing distance after reaching a node v_2 with distance $(d-1)$, v_2 belongs to the set $\mathcal{V}_{v_1,d-1}$. For example, in Figure 5.1, we have the sets $\mathcal{V}_{6,d-1}=\{3,4\}$, $\mathcal{V}_{7,d-1}=\{3,4\}$ and $\mathcal{V}_{5,d-1}=\{2\}$.

In the following, there will be a slight abuse of notation: When $\bar{w}_i$ is an SSCC of the steady states of some set of nodes $\bar{\mathcal{V}}=\{j_1,\dots,j_l\}$, instead of having to write $\bar{w}_i \in C(\bar{w}_{j_1},\dots,\bar{w}_{j_l})$ where $\bar{w}_{j_k}$ is the steady state of node j_k, we will write $\bar{w}_i \in C(\bar{\mathcal{V}})$.

Theorem 5.2. *Given the system dynamics (5.11), Assumption 5.1 with (5.9), and a constant input $u \in \mathbb{R}_{++}$ at a designated source node, the steady state value of a node v with distance d from the source node is an SSCC as defined in (5.12) of the steady state values of unique $(d-1)$-nodes of node v, i.e.*

$$\bar{w}_v \in C(\mathcal{V}_{v,d-1}).$$

Proof. Given a designated source node, we denote the set of nodes with distance d from that source node by $\mathcal{V}_d$. The maximum distance in the network is d_M. All the nodes on level d_M from the source node either have only neighbors in the set $\mathcal{V}_{d_M-1}$ or additionally in the set $\mathcal{V}_{d_M}$. Thus, for any node $v_{d_M} \in \mathcal{V}_{d_M}$, we have

$$\bar{w}_{v_{d_M}} \in C(\mathcal{N}_{v_{d_M},d_M}, \mathcal{N}_{v_{d_M},d_M-1}),$$

because of (5.9) where the set $\mathcal{N}_{v_{d_M},d_M}$ denotes the neighboring nodes of node v_{d_M} with distance d_M from the source node, and it could be the empty set, while $\mathcal{N}_{v_{d_M-1},d_M}$ denotes those with distance d_M-1. In this case, we have $\bar{w}_{v_{d_M}} \in C(\mathcal{N}_{v_{d_M},d_M-1})$ and the result of Theorem 5.2 is apparent. Otherwise, suppose that node v_{d_M} is connected to nodes $v_{1,d_M},\dots,v_{n,d_M}$ in addition to nodes with distance d_M-1. Then, we have

$$\bar{w}_{v_{d_M}} \in C(\bar{w}_{v_{1,d_M}},\dots,\bar{w}_{v_{n,d_M}}, \mathcal{N}_{v_{d_M},d_M-1}).$$

Applying Lemma 5.2 to the $\bar{w}_{v_{i,d_M}}$ replaces them with their neighbors, all of which again belong to the sets $\mathcal{V}_{d_M}$ or $\mathcal{V}_{d_M-1}$, and all can be reached from node v_{d_M} via a directed path. Eventually, Lemma 5.2 can be used repeatedly to cancel out all nodes on the layer d_M, i.e. those belonging to the set $\mathcal{V}_{d_M}$, and replace them by nodes from the set $\mathcal{V}_{d_M-1}$. In the end, $\bar{w}_{v_{d_M}}$ is an SSCC of nodes on the level d_M-1 that can be reached from v_{d_M} via a directed path through its multihop neighbors on the level d_M. However, there may be nodes on the level d_M-1 that do not have neighbors in the part of the layer d_M that can be reached from v_{d_M} via several hops. Furthermore, there are nodes on the level d_M-1 that can only be reached through nodes on the level d_M-1, i.e. nodes not belonging to $\mathcal{V}_{v_{d_M},d-1}$. Both types of nodes do not occur in any of the SSCCs of the multihop neighbors and thus we obtain

$$\bar{w}_{v_{d_M}} \in C(\mathcal{N}_{v_{d_M},d_M-1}, \mathcal{N}_{v_{1,d_M},d_M-1},\dots,\mathcal{N}_{v_{n,d_M},d_M-1}, \mathcal{N}_{\mathcal{N}_{v_{1,d_M},d_M},d_M-1},\dots,\mathcal{N}_{\mathcal{N}_{v_{n,d_M},d_M},d_M-1},\dots).$$

Thus, we have the result of Theorem 5.2 for the nodes with maximum distance.

We proceed to the level with distance $d_M - 1$, and choose a node of interest, call it v_{d_M-1}. On this level, each node can have neighbors on the current level, the one below and the one above, i.e.

$$\bar{w}_{v_{d_M-1}} \in C(\mathcal{N}_{v_{d_M-1},d_M-1}, \mathcal{N}_{v_{d_M-1},d_M}, \mathcal{N}_{v_{d_M-1},d_M-2}).$$

With the results shown to be valid for nodes from level d_M and Lemma 5.2, the neighboring sets of nodes from level d_M can be replaced by multihop neighborhood nodes from the current level, i.e.

$$\bar{w}_{v_{d_M-1}} \in C(\mathcal{N}_{v_{d_M-1},d_M-1}, \mathcal{N}_{\mathcal{N}_{v_{d_M-1},d_M},d_M-1}, \mathcal{N}_{v_{d_M-1},d_M-2}).$$

The second term in the parentheses describes those nodes in the set $\mathcal{V}_{d_M-1}$ that are not direct neighbors of node v_{d_M-1}, i.e. $\mathcal{N}_{\mathcal{N}_{v_{d_M-1},d_M},d_M-1} \cap \mathcal{N}_{v_{d_M-1}} = \emptyset$, but which can be reached from v_{d_M-1} through nodes from the set $\mathcal{V}_{d_M}$ via a directed path. We proceed via the same reasoning as previously, and replace all dependencies on nodes from the current level $d_M - 1$ with nodes from the set $\mathcal{V}_{d_M-2}$ by repeatedly applying Lemma 5.2. All these neighborhoods on layer $d_M - 2$ contain nodes that can be reached either directly from v_{d_M-1} or through other nodes on the current layer $d_M - 1$ or the previous layer d_M. Notice again that there may be nodes with distance $d_M - 2$ that cannot be reached from v_{d_M-1} or any of its multihop neighbors and, therefore, never occur in their neighborhoods and their SSCCs. Thus, they can never enter the description of v_{d_M-1} and the result of Theorem 5.2 is obtained. The levels $d_M - 2 \leq d \leq 1$ can be treated in the same way which concludes the proof. □

We illustrate the result of Theorem 5.2 in the following example.

Example 5.2. *Tridiagonal systems*

For tridiagonal systems represented by a line graph as shown in Figure 5.5a, the gain from one node to the next is less than $1-\gamma$ as we move away from the source node. At the end of the line, we have

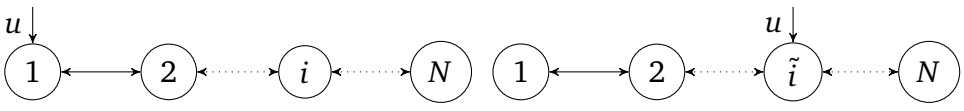
$$\bar{w}_N = \frac{\alpha_{N,N-1}}{\alpha_{N,N}} \bar{w}_{N-1} = \beta_{N,N-1} \leq (1-\gamma)\bar{w}_{N-1}$$

and for all other nodes with $i = 2, \ldots, N-1$

$$\bar{w}_i = \frac{\alpha_{i,i-1}}{\alpha_{i,i} - \alpha_{i,i+1}\beta_{i+1,i}} \bar{w}_{i-1} = \beta_{i,i-1} \leq (1-\gamma)\bar{w}_{i-1}.$$

Furthermore, at the source node, one has

$$\bar{w}_1 = \frac{1}{\alpha_{1,1} - \alpha_{1,2}\beta_{2,1}} u.$$



(a) Input at node 1. (b) Input at node $\tilde{i}$.

Figure 5.5: Line graph topology.

Clearly, there is a strict decay from $\bar{w}_1$ as we step through each one of the nodes to $\bar{w}_N$ and the gain is bounded by $1-\gamma$ per hop, or by $(1-\gamma)^d$, where d is the distance of a node from the source node.

Note that this result can be generalized to having the input at an arbitrary node $\tilde{i}$ as depicted in Figure 5.5b. One obtains the relationships

$$\bar{w}_i = \frac{\alpha_{i,i+1}}{\alpha_{i,i} - \alpha_{i,i-1}\beta_{i-1,i}}\bar{w}_{i+1} = \beta_{i,i+1}\bar{w}_{i+1} < (1-\gamma)\bar{w}_{i+1},\ i = 1,\ldots,\tilde{i}-1,$$
$$\bar{w}_i = \frac{\alpha_{i,i-1}}{\alpha_{i,i} - \alpha_{i,i+1}\beta_{i+1,i}}\bar{w}_{i-1} = \beta_{i,i-1}\bar{w}_{i-1} < (1-\gamma)\bar{w}_{i-1},\ i = \tilde{i}+1,\ldots,N.$$

For $i = \tilde{i}$, i.e. the node with excitation, we have

$$\bar{w}_{\tilde{i}} = \frac{1}{\alpha_{\tilde{i},\tilde{i}} - \alpha_{\tilde{i},\tilde{i}+1}\beta_{i+1,i} - \alpha_{\tilde{i},\tilde{i}-1}\beta_{i-1,i}}u.$$

This shows that we have the same result as before, only that now the decay is in both directions. Furthermore, each direction is essentially independent of the other one in steady state. This decay property in a line graph is depicted in Figure 5.6 in the form of the upper bound $(1-\gamma)^d$ for a particular selected value of $\gamma = 0.1$. Again, it is straightforward to extend this result to more general, but related, topologies such as stars or trees.

Note that for completely general graphs, there does not necessarily need to be a decay between neighboring nodes with increasing distance between hops. Nevertheless, there still is a form of a decay property because the steady state value of a node with distance d is smaller than the largest steady state value among the nodes in the set of unique $(d-1)$-nodes although it is not necessarily a direct neighbor of the considered node. In other words, for general topologies, the decay is from set to set rather than from node to node.

5.4.2 Steady state magnitude decay with sinusoidal input

In this subsection, we examine the same scenario as in the previous one, only now instead of $u(t) = |u|$ we consider $u(t) = |u|e^{j\omega t}$, i.e. we have a sinusoidal input. While in one sense the results of the previous subsection are a special case corresponding to $\omega = 0$ of the results of this subsection, we will see here that the results for general ω are best viewed in the light

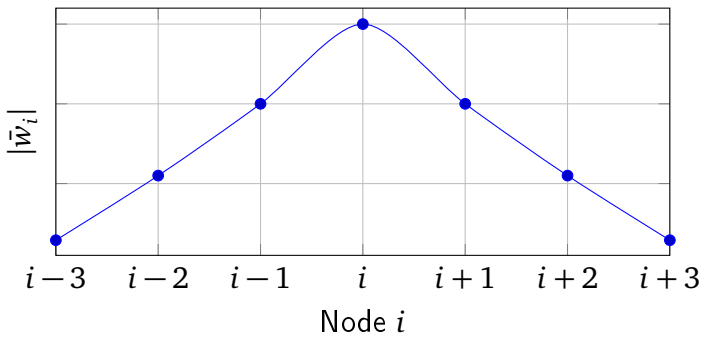


Figure 5.6: Illustration of decay in line graph for $\gamma = 0.1$.

of those for $\omega = 0$. Because the system is stable, the steady states of the subsystems are also of sinusoidal form with the same frequency but with a constant phase shift. Hence, we have

$$\bar{w}_i(t) = |\bar{w}_i| e^{j(\omega t + \varphi_i)} \tag{5.14}$$

for time t after an initial settling time, and we define the vector $\varphi = [\varphi_1, \ldots, \varphi_N]$.

To start off, we only consider the line graph case. Beginning at the end of the line, only considering the magnitudes and by using the differential equation (5.7) as well as the assumed solutions (5.14), we obtain the relationship

$$\begin{aligned} |\bar{w}_N| &= \frac{\alpha_{N,N-1}}{\sqrt{\alpha_{N,N}^2 + \omega^2}} |\bar{w}_{N-1}| = \beta_{N,N-1}(\omega, \varphi) |\bar{w}_{N-1}| \\ &< (1-\gamma)|\bar{w}_{N-1}|. \end{aligned}$$

For the other nodes with $i = 2, \ldots, N-1$, the bounds on the gain are given by

$$\begin{aligned} |\bar{w}_i| &= \frac{\alpha_{i,i-1}|\bar{w}_{i-1}|}{\sqrt{(\alpha_{i,i} - \alpha_{i,i+1}\beta_{i+1,i}(\omega,\varphi)\cos(\varphi_{i+1} - \varphi_i))^2 + (\omega - \alpha_{i,i+1}\beta_{i+1,i}(\omega,\varphi)\sin(\varphi_{i+1} - \varphi_i)^2}} \\ &= \beta_{i,i-1}(\omega,\varphi)|\bar{w}_{i-1}| < (1-\gamma)|\bar{w}_{i-1}|. \end{aligned}$$

The relationship for the source node is

$$|\bar{w}_1| < \frac{1}{\sqrt{(\alpha_{1,1} - \alpha_{1,2}\beta_{2,1}(\omega,\varphi)\cos(\varphi_2 - \varphi_1))^2 + (\omega - \alpha_{1,2}\beta_{2,1}(\omega,\varphi)\sin(\varphi_2 - \varphi_1))^2}} u.$$

By inspection, we notice that with $\omega = 0$ and all $\varphi_{i+1} - \varphi_i = 0$, we obtain that $\beta_{i,j}(0,0)$ corresponds to $\beta_{i,j}$ from the constant input case such that the results from the constant input are recovered and are in fact generalized here. Notice also that the *constant input case* $\beta_{i,j}(0,0)$ *is always an upper bound for the sinusoidal case.*

If we consider the general dynamics of subsystem i described by (5.7) with a sinusoidal input and neighbors in the set $\{j_1, \ldots, j_l\}$, one obtains for the steady state magnitude that

$$\begin{aligned} |\bar{w}_i| &= \frac{\sqrt{(\alpha_{i,j_1}|\bar{w}_{j_1}| + \sum_{j=j_2}^{j_l} \alpha_{i,j}|\bar{w}_j|\cos(\varphi_j - \varphi_{j_1}))^2 + (\sum_{j=j_2}^{j_l} \alpha_{i,j}|\bar{w}_j|\sin(\varphi_j - \varphi_{j_1}))^2}}{\sqrt{\alpha_{ii}^2 + \omega^2}} \\ &\leq \sum_{j=j_1}^{j_l} \frac{\alpha_{i,j}}{\alpha_{i,i}} |\bar{w}_j|. \end{aligned}$$

We observe that the major difference to the case with constant input is that one cannot express the steady state magnitude as a linear combination of the neighboring steady state magnitudes, but that it can be bounded by the same linear combination obtained in the constant input case. Hence, the steady state magnitude in the sinusoidal case is always bounded by the steady state of the constant input case. Therefore, we can state the following corollary, omitting the proof.

Corollary 5.1. Given the system dynamics (5.11), and a sinusoidal input $u(t) = |u|e^{j\omega t}$ at a designated source node, the steady state magnitude of a node v with distance d from the source node is upper bounded by an SSCC as defined in (5.12) of the steady state magnitudes of the unique $(d-1)$-nodes of node v.

5.5 Numerical example

In this section, we illustrate the obtained results in a numerical example.

We consider the following system dynamics for the ith subsystem

$$\begin{aligned}\dot{x}_{1,i} &= x_{2,i},\\ \dot{x}_{2,i} &= -k_{0,i}x_{1,i} - b_{0,i}x_{2,i} + \sum_{j\in\mathcal{N}_i}\left(k_{1,i}x_{1,j} + b_{1,i}x_{2,j}\right),\end{aligned}$$

and we assume that the interconnection topology is given by the line graph shown in Figure 5.5a. We consider $N = 10$ subsystems, and the parameters $k_{0,i}$ and $b_{0,i}$ are chosen from the set $\{1,2,3\}$ while the parameters $k_{1,i}$ and $b_{1,i}$ are selected randomly from the set $\{0.01, 0.02, 0.03, 0.04, 0.05\}$. The parameters are chosen in this manner to ensure that a matrix $\mathscr{A}$ exists such that $-\mathscr{A}$ is an M-matrix. The system dynamics can represent a platoon of vehicles where the velocity of the ith vehicle is influenced by the position and the velocity of the preceding and the following vehicle, e.g. through a control law, or because of an actual physical connection.

Given the state matrix A, we first construct $\tilde{\mathscr{A}}$ where we choose Q_i to be the identity matrix of size two. Then, we determine a diagonal matrix D such that $\mathscr{A} = D^{-1}\tilde{\mathscr{A}}D$ is row diagonally dominant. The resulting reduced system dynamics are described by (5.11), where

$$\mathscr{A} = \begin{bmatrix}
-0.96 & 0.86 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0.09 & -0.31 & 0.19 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0.5 & -0.57 & 0.01 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0.48 & -1.14 & 0.05 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0.07 & -0.97 & 0.22 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0.14 & -0.58 & 0.15 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0.34 & -0.87 & 0.09 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0.1 & -1.1 & 0.26 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.06 & -0.44 & 0.18\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.44 & -0.83
\end{bmatrix},$$

and we choose $b = [1, 0, \ldots, 0]^\mathsf{T}$.

In this case, $\gamma = 0.1021$. Therefore, we expect the gain between hops from subsystem to subsystem to be at most $(1-\gamma) = 0.8979$. The response of the system to the constant input signal $u(t) = 1$ is shown in Figure 5.7. The steady state values $\bar{w}_i$ of the subsystems are shown in Figure 5.8. The spatial decay is clearly observed. The gain between hops ($\frac{\bar{w}_i}{\bar{w}_{i-1}}$) lies between 0.0745 and 0.8808 as shown in Figure 5.9. It can be seen that the upper bound $(1-\gamma)$ per hop is conservative in general, but in one case it is very close to the actual gain. The bound would be accurate for identical subsystems, otherwise the conservativeness of the bound depends on the degree of heterogeneity between the subsystems.

Next, we apply a sinusoidal input $u(t) = \sin(\omega t)$ to the system. The system response for $\omega = 1$ is shown in Figure 5.10 and one can easily see that (1) the spatial decay between nodes, and (2) the phase shift between hops. In order to illustrate the spatial decay, the steady state magnitudes are also added to Figure 5.8. The gain between hops, now described

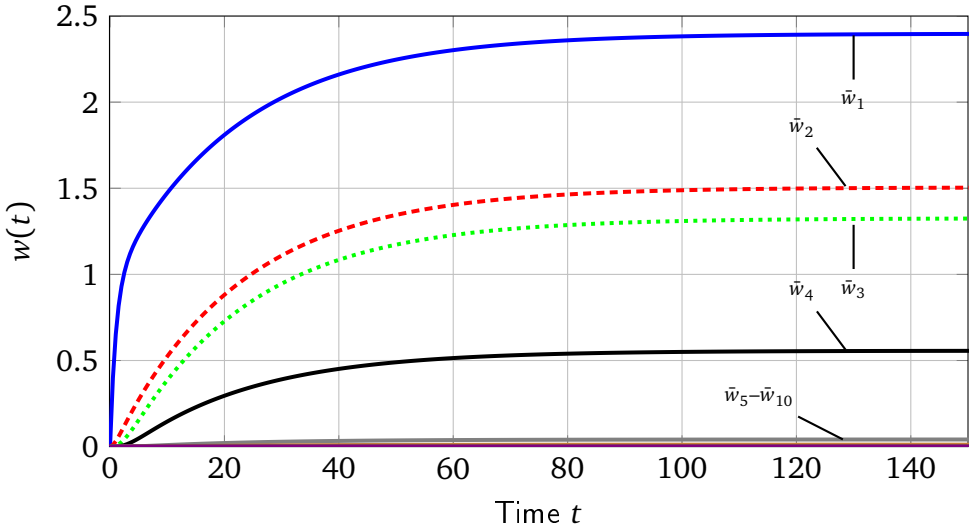


Figure 5.7: Response to $u(t) = 1$ for line-graph.

by $\frac{|\bar{w}_i|}{|\bar{w}_{i-1}|}$, is between 0.0484 and 0.4316. All the values, in addition to the values of the gain for $\omega = 0.5$ and $\omega = 2$, are shown in Figure 5.9. The constant input case is an upper bound for the sinusoidal case, as expected from the general analysis. Also, as expected, while the bound is close in some cases, it is quite conservative in others and it becomes more and more conservative for increasing ω. However, at least for relatively small values of ω, the constant input case can be considered to be a good approximation as a bound for the sinusoidal case and offers easier computations.

5.6 Chapter summary

In this chapter, we investigated the propagation of an input signal through a dynamical system. The first main idea was to replace the original dynamics by comparison dynamics. In order to achieve this, the individual subsystems were represented by a scalar approximation such that the overall system was in the form of a vector Lyapunov function. We then assumed that the new system dynamics matrix satisfied an M-matrix condition. For two special cases, a constant input and a sinusoidal input, an analytical analysis of the steady state values was carried out to see what the gain was between hops from subsystem to subsystem. The main result shows that the steady state value of a node with a specific distance d to the source node can be expressed as a so-called steady state conic combination of the steady state values of nodes with distance $(d-1)$ to the source node. For special cases such as line-graphs or trees, the gain between hops can be bounded by a parameter γ that is in direct relation to the entries of the system dynamics matrix. In addition, it was shown that the constant input case can serve as an upper bound to the case with sinusoidal input. The results were validated and illustrated using numerical simulations. Furthermore, the results of Chapter 5 have implications for linear algebra applications. The derived decay property is relevant for sparse approximations of inverse matrices used as preconditioners [133]. The results here represent such a decay property for general matrix structures.

Note that the results of this chapter are partially based on [27].

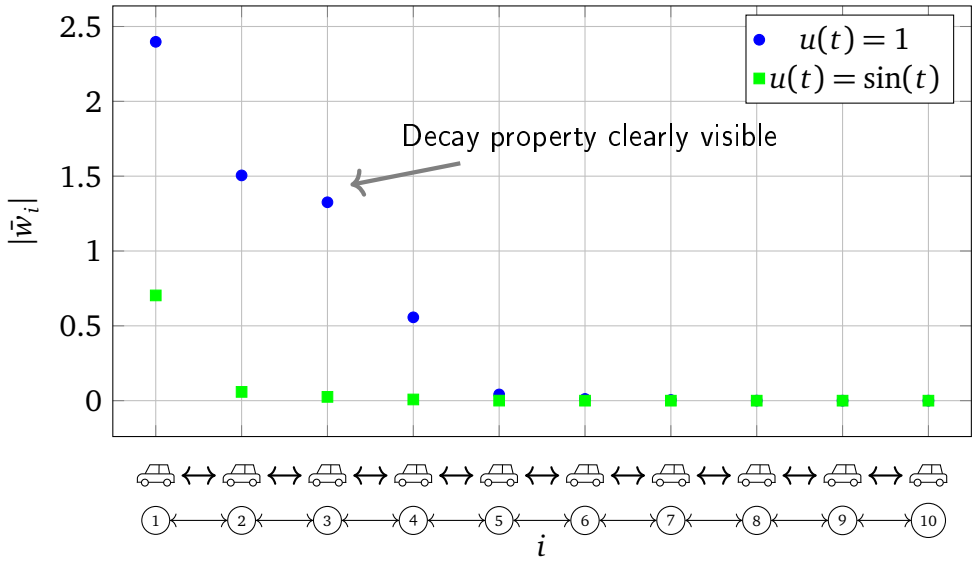


Figure 5.8: Steady state values showing spatial decay for line graph.

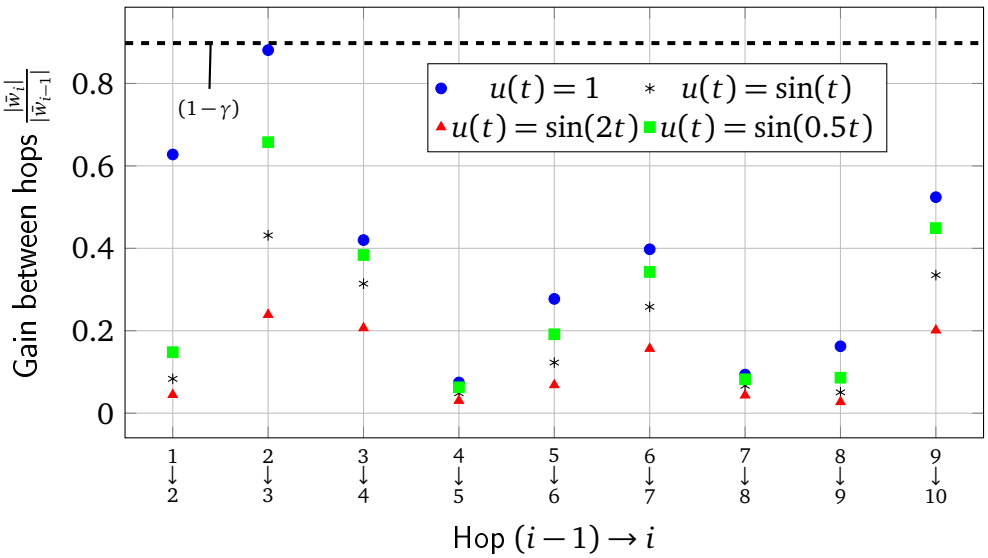


Figure 5.9: Actual gain between hops with upper bound $(1-\gamma) = 0.8979$.

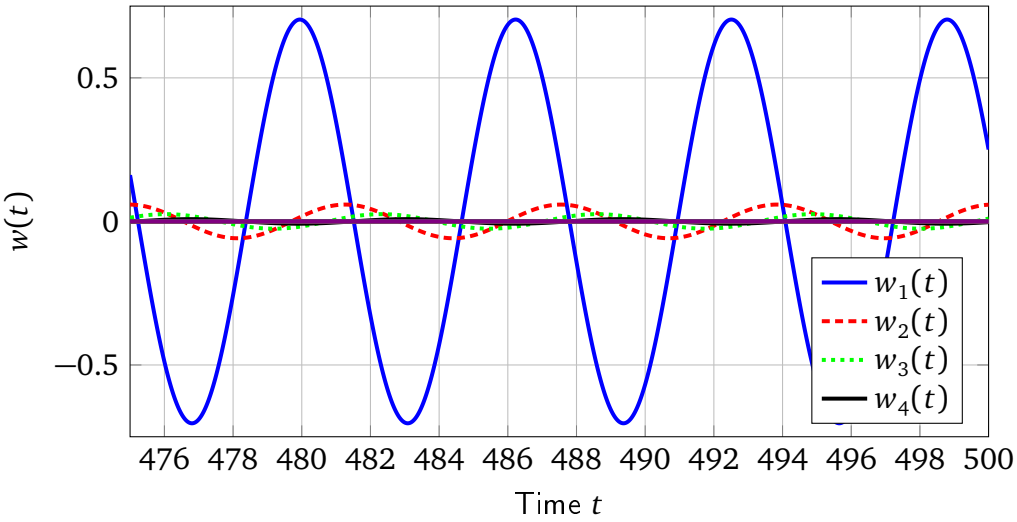


Figure 5.10: Response to $u(t) = \sin(t)$ for line-graph.

6

Conclusions and outlook

System analysis and distributed optimal control of large-scale interconnected systems in and of themselves are very challenging problems. However, the desire to preserve model data privacy for the subsystems as well as computational reasons related to ever increasing numbers of subsystems raise additional fundamental challenges. The central question is how to achieve these tasks without a central entity, and relying only on local model knowledge that is distributed among the subsystems. The overall problem is to develop control related methods to solve problems in a distributed setting while achieving a performance that is as close to a centralized solution as possible. This entails developing completely new methods as well as translating existing ones to the non-centralized perspective. Modern distributed optimization methods are an enabling technique to distributed decision-making to achieve this. This thesis contributes to this endeavor by addressing three separate, but ultimately related problems, presented in Chapters 3-5.

Distributed system analysis. The main contribution to the field of distributed system analysis with local model information was presented in Chapter 3 in the form of two different stability tests. The first developed method is based on vector Lyapunov functions and the test itself eventually consists of a linear program. While the formulation of the problem requires only local model information already, the main contribution is the application of distributed optimization methods such as the DCNA to solve it. This allows the systems to determine whether the overall interconnected system is connectively stable by cooperatively solving an optimization problem. In summary, this first developed stability test enables the distributed test of a well-known and established stability condition for interconnected dynamical systems using only local model information. The second developed method is based on the Lyapunov inequality. In general, however, the Lyapunov inequality does not exhibit a particular structure, which makes finding a solution in a distributed setting impossible. Therefore,

we focused our attention on the special case of α-block diagonal Lyapunov stability. While this sacrifices the necessity of the Lyapunov stability condition for LTI systems, it enabled us to set up an optimization problem that can be solved without requiring a centralized entity. One of the key steps is introducing additional slack variables which help in decomposing the LMI among subgroups of the overall system. Then, we could again employ the DCNA to solve the distributed optimization problem. The advantage of this stability test over existing distributed stability tests is that general LTI systems can be analyzed as long as the allowed communication topology is chordal.

The second test is essentially a generalization of the first one and is considerably less conservative at the price of a higher computational demand. In addition to the conservativeness comparison between the two tests, we also investigated the conservativeness of the α-block diagonal Lyapunov condition compared to the standard Lyapunov inequality. No definitive judgment could be made on the conservativeness because a high dependence on the system parameters was observed. However, we introduced a new necessary condition for α-block diagonal Lyapunov stability and showed in numerical experiments that it identifies the majority of the considered systems correctly.

The final contribution of Chapter 3 was that we showed that it is also possible to use the same approach from the α-block diagonal Lyapunov stability test to achieve distributed model reduction. Based on the method of generalized gramians, we formulated a distributed optimization problem that allowed us to determine the generalized Hankel values using only local model information. Based on the generalized Hankel values and minimum consensus, we could then make distributed decisions on which states can be reduced to obtain a reduced order model.

Distributed optimal control design. In Chapter 4, we presented our contributions to the distributed optimal control design with local model information. The problem that we addressed considers LTI systems with an LQ cost functional. The first idea was that instead of an infinite horizon cost functional, we considered a finite horizon cost functional but use a specific terminal cost term such that the finite horizon cost is an upper bound to the infinite horizon cost. The finite horizon then allowed us to compute the gradient of the cost with respect to the entries of the linear state feedback matrix based on simulated trajectories of the state and of an introduced adjoint system. With this gradient, we can iteratively optimize the distributed control law using only local model information. The approach allows that three graphs describing the problem, namely the computation, control and design graphs, do not have to be identical. This offers an additional degree of freedom. The price of the problem reformulation to a finite horizon is that stability of the closed loop is not straightforward and is not a direct result of optimality. We solved this challenge by using the terminal cost term to guarantee stability. Furthermore, and more importantly for the distributed problem setting, we employed the techniques from the distributed stability test to present an algorithm to compute the terminal cost term using only local model information.

To accelerate the gradient method, we also made use of a consensus algorithm to distributedly compute the BB step size. The same approach was also applied to compute the conjugate gradient search direction. The performance of different step size methods and of the conjugate gradient method revealed that no clear decision can be made between the gradient

approach with BB step size and the conjugate gradient method. However, both outperformed the naive choice of a constant step size. In addition, we can distributedly guarantee convergence, which is of high significance. Combining all these results, we presented a complete approach to determine a distributed control law without a centralized entity and with distributed model knowledge that guarantees stability and which is optimized with respect to an LQ cost functional.

The inherent price to computing the control law in a distributed fashion is the communication effort that is vastly increased in comparison to a centralized design. In the control design, a part of this communication effort is due to the need to simulate trajectories of the system, which requires the subsystems to exchange state information. To alleviate this problem, we presented an event-based Euler method which drastically reduces the required communication during the trajectory simulation.

Additionally, we were able to extend the design approach to two further problem classes. The first class was systems whose dynamics follow a differential-algebraic equation. This system class is of practical relevance as it is often used to model distribution networks in an exact way. The resulting algorithm, however, requires even more information exchange than in the ODE case to simulate the trajectories. The second problem class is optimal formation control of a multi-robot system. While we disregarded a distributed design here, the problem was still challenging for the following reason. To include the formation rigidity requirement into the cost functional, the cost had to be reformulated to a biquadratic form which violates the standard LQ problem form. Therefore, the approach of using simulated trajectories to obtain a gradient was useful in finding an optimal control law because no analytical solution is available. In addition to the gradient method, we also used a Quasi-Newton method to improve the performance of the iterative control design.

Finally, we presented two generalizations of the control design method. The first one was a time-varying control law, the second one was a nonlinear parameterized control law for nonlinear systems. These results illustrated the versatility of the design method. Interestingly, a numerical experiment indicated that the time-varying control law does not necessarily improve the performance in comparison to the time-invariant one, at least when it is determined with the presented approach. This raises questions for future work.

Decay analysis in interconnected systems. Chapter 5 addressed the final problem related to interconnected systems and their treatment from a distributed perspective. The chapter analyzed the problem of the decay of a signal, e.g. of a disturbance, along the graph representing the system. As we were mostly interested in the interaction between subsystems and not the detailed behavior inside each subsystem, we used the idea of introducing comparison system dynamics that approximate every multi-dimensional subsystem by a scalar representation. Based on the assumption that the stability of the overall system can be shown using a vector Lyapunov function, we showed a decay property with increasing distance from that source node given a single constant or sinusoidal input at one node. The decay property, however, is not necessarily from one node to a neighboring node, but rather between sets of nodes where the two sets have different distances to the source. For special cases such as line-graphs and combinations of line-graphs such as trees and stars, the property boils down to a simple decay from one node to the next.

While we abandoned the distributed perspective in this last chapter, there still is a relevant connection to the previous ones. The results of the decay analysis provide a considerable insight into the behavior of interconnected systems and how strong the interaction between subsystems is. This is relevant for the design of control laws in two ways. (1) If one knows that there is a decay property of the presented form in a given system, it indicates that subsystems at a certain distance can be removed from the control graph without any loss of performance. (2) In most applications, though not all, it is desirable to have such a decay property, which raises interest in new control design methods to achieve it. In addition, we have noticed some relevance of the results for linear algebra applications.

In summary, this thesis presents considerable contributions to the analysis and control of interconnected systems from a distributed perspective with local model information.

6.1 Outlook

In this thesis, we presented results to take a step towards system analysis and distributed control design with local model information. The results consist of novel and innovative methods as well as insightful analyses. However, several open research problems in the area remain to be addressed in the future and we summarize them in the following.

Localized (re-)design of control laws
In Chapter 4, we considered the control design for the overall LTI systems. Based on the results from Chapter 5, however, it is apparent that for many interconnected systems, the interaction between subsystems does not affect the whole system. This implies that it is possible to design local control laws for only parts of the system and then combine the parts to an overall control law in an appropriate way without sacrificing performance. This reduces the computational and communication effort of the approach and further localizes the control design. A related problem is the identification of subsystem clusters that collaborate in the control design.
Similarly, while we considered the offline design of a control law for LTI systems, it is unreasonable to assume that the system remains unchanged for all times. System changes can include new subsystems joining the system, old subsystems leaving the system or the change of system parameters. In the current state, the presented approach requires a complete redesign of the control law. It is, however, more desirable, and in the light of the results of Chapter 5 also likely possible, to re-design the new control law only locally around the change while keeping the rest intact and unchanged. The idea is that subsystems that are far away from the change are not affected anyway.
Another possibility is to adapt the control law online. Reinforcement learning is a promising possibility in this regard in that it can react to changes but still considers optimality with respect to a user-specified cost. So far, however, distributed results are not available.

Further reduction of communication effort
One inherent problem of distributed approaches is that the communication effort is vastly increased over centralized methods. We presented approaches to alleviate this problem, like

the event-based trajectory simulation and the possibility of the event-based DCNA implementation. Further steps include, for example, an emulation-based event-based controller. This means that the control law is designed according to the presented methods, and then implemented with event-based communication using an appropriate message scheduling. In addition, an event-based solution for the DAE dynamics is useful in reducing the communication effort for the control design for this system class. Furthermore, extending the event-based Euler approach to higher order ODE solution methods can also reduce the required communication. Finally, a connection to the previous paragraph can be made. If we can localize the control design, this can also reduce the communication effort by ensuring that subsystems do not have to exchange information in the first place. This aspect of future work is especially relevant on the way to implementations of distributed control design in practice.

Distributed design for more general optimal control problems
This thesis presented control design methods that mostly pertained to the linear quadratic problem setup. While we have formulated extensions and generalizations, there are still various optimal control problems that are not addressed. Of particular interest are the well-established problem classes of general H_2 and H_∞ optimal control. It is expected that the presented methods do not carry over directly to these problem classes. One major concern is that it is not clear how to represent the respective cost functionals based on simulated trajectories which is an important step in the present approach and one of the key enabling techniques to achieve distributed computation. It may be possible, however, to identify further subclasses that permit this approach.

Transient decay analysis
The analysis in Chapter 5 focused on the steady state analysis given a constant or sinusoidal input. Naturally, it is also of interest how the transient behavior is affected with increasing distance to a source node. An interesting question is the maximum state amplitude at a node, given a disturbance at a distant node. This insight can also help in justifying a localized redesign of control laws mentioned earlier. In addition, further control design methods should be developed that enforce the decay property presented in this work.

Detailed analysis of privacy constraints
In this thesis, we considered the notion of model data privacy. Under this privacy notion, we understood that subsystems do not need to share their model data globally or centrally but only with a small subset of the other subsystems. Two further points should be considered. (1) What other privacy notions can be included in the control design and system analysis. (2) Given simulated trajectories, what information can be inferred or reconstructed about the overall system. The latter question is also related to the results in Chapter 5 in that a decay property implies that far away subsystems cannot infer any information and only trajectories from close neighbors contain enough meaningful information. This needs to be analyzed in more detail because privacy is of enormous relevance when it comes to the widespread practical implementation of distributed control.

A

Appendix

In this appendix, we provide some introductions to well established concepts that are used in this thesis. The first section concentrates on descent methods for the minimization of a function. The second part is devoted to graph theory and the third one to model reduction based on balanced truncation.

A.1 Descent methods for minimization

In this section, we review some basic results on descent methods to minimize a given function. For more details, and a broader overview, we refer the reader to well-known textbooks such as [78, 134] upon which this introduction is based.

The goal of descent methods is to minimize the function $f(x)$, i.e.

$$\min_{x \in \mathbb{R}^n} f(x),$$

where $n \geq 1$. In order to find an approximation of the optimal solution x^* and of the optimal value f^*, the idea of descent methods is to iteratively improve the approximate solution which gives

$$x_{k+1} = x_k + \gamma_k d_k, \tag{A.1}$$

where $\gamma_k \in \mathbb{R}$ is a step length, and $d_k \in \mathbb{R}^n$ is a search direction. The search direction is a descent direction if it satisfies the property $d_k^\mathsf{T} \nabla f(x_k) < 0$.

The two typical forms for search directions that are used in this thesis are

$$d_k = -B_k^{-1} \nabla f(x_k) \tag{A.2}$$

and

$$d_k = -\nabla f(x_k) + \beta_k d_{k-1}. \tag{A.3}$$

The first form, Equation (A.2), reduces to the steepest descent or gradient descent method, when $B_k = I_n$; it is the Newton method when $B_k = \nabla^2 f(x_k)$ and it is a Quasi-Newton method when B_k is an approximation of the Hessian matrix $\nabla^2 f(x_k)$. The second form, Equation (A.3), is a conjugate gradient method when β_k satisfies certain properties.

A.1.1 Step length

The step length γ_k in (A.1) has a large influence on the convergence speed of a given descent method. For the Newton method, where $B_k = \nabla^2 f(x_k)$, the step size is typically always 1. For all other methods, there is no straightforward choice that is always optimal for any given problem. An intuitive way is to select the step length γ_k that gives the largest cost decrease, that is

$$\gamma_k = \min_{\gamma} f(x_k + \gamma d_k).$$

This represents a one-dimensional optimization problem, but in general it is computationally expensive. Furthermore, it has been shown that this approach is not the best one in terms of overall convergence speed because it may lead to unnecessarily small steps [135]. Instead, a typical choice in practice is to use the result of an inexact line search using the Powell-Wolfe conditions. The two conditions are

$$\begin{aligned} f(x_k + \gamma_k d_k) &\leq f(x_k) + c_1 \gamma_k \nabla f(x_k)^\mathsf{T} d_k, \\ \nabla f(x_k + \gamma_k d_k)^\mathsf{T} d_k &\geq c_2 \nabla f(x_k)^\mathsf{T} d_k, \end{aligned}$$

where the first one requires a sufficient decrease of the objective function value, while the second one is called curvature condition. It can be shown that there always exists a step length γ_k that satisfies both conditions if the function f is smooth and bounded from below [78]. Furthermore, the curvature condition can be replaced by the strong Wolfe condition which is

$$|\nabla f(x_k + \gamma_k d_k)^\mathsf{T} d_k| \leq |c_2 \nabla f(x_k)^\mathsf{T} d_k|.$$

Typical values for the parameters are $c_1 = 10^{-4}$ and $c_2 = 0.9$ for Quasi-Newton methods, or $c_2 = 0.1$ for gradient and conjugate gradient methods. The sufficient decrease condition guarantees that we have a monotonous decrease with every step, while the curvature condition ensures that the steps are not too short. The two conditions can be used to identify a good step length starting from an initial guess. If the initial guess satisfies the conditions, it can be used. Otherwise, the step length is iteratively decreased until the conditions are satisfied.

Constant step length
An easy and intuitive choice for the initial guess is a constant step size $\gamma_k = \gamma_0$ for all k. This step length choice requires no computations except for the Powell-Wolfe condition check, but it does not take into account any knowledge obtained during the solution. Furthermore, it can be difficult to select the right γ_0 as a good choice highly depends on the individual problem.

Barzilai-Borwein (BB) step length
A different approach is the Barzilai-Borwein (BB) step length, first introduced in [105]. The BB step length approximates a certain Quasi-Newton property. Quasi-Newton methods require that the Hessian approximate B_k satisfies the so-called secant equation

$$B_k s_k = y_k, \tag{A.4}$$

where $s_k = x_k - x_{k-1} = \gamma_k d_k$ and $y_k = \nabla f(x_k) - \nabla f(x_{k-1})$. In order to derive the BB step length, we assume that $B_k = \gamma_k I_n$ and we aspire to satisfy (A.4) as closely as possible. Hence, we aim to minimize $||s_k - \gamma_k y_k||_2$. It is easy to verify that this leads to

$$\gamma_k = \frac{y_k^\top s_k}{y_k^\top y_k}.$$

The step length has two special properties. First, its computation uses gradient information from the current and the previous iterations. Second, it is non-monotonous by itself which means that it does not necessarily lead to a decrease in the function value with each iteration. Convergence can be shown for quadratic problems, but monotonicity must be required for general nonlinear problems. The Powell-Wolfe conditions allow this test. Therefore, one can compute the BB step length, check the conditions and then possibly decrease it.

Investigations have shown that for quadratic problems, a gradient method with the BB step length is competitive even with conjugate gradient methods, and for nonquadratic problems, it might even outperform conjugate gradient methods [106].

A.1.2 Search direction

In this thesis, mainly three different search directions are employed.

Gradient descent
In this case, the search direction is $d_k = -\nabla f(x_k)$, i.e. the negative gradient, and it corresponds to the steepest possible descent. This method requires the least computational effort to compute the search direction itself, but it has generally slow convergence.

Conjugate gradient method
For a conjugate gradient method, the search direction is given by

$$d_k = -\nabla f(x_k) + \beta_k d_{k-1}.$$

There are several different methods to choose the parameter β_k, but the most popular ones are the Fletcher-Reeves (FR) method which gives

$$\beta_k = \frac{\nabla f(x_k)^\mathsf{T} \nabla f(x_k)}{\nabla f(x_{k-1})^\mathsf{T} \nabla f(x_{k-1})},$$

and the Polak-Ribière (PR) method which yields

$$\beta_k = \frac{\nabla f(x_k)^\mathsf{T} (\nabla f(x_k) - \nabla f(x_{k-1}))}{\nabla f(x_{k-1})^\mathsf{T} \nabla f(x_{k-1})}.$$

The conjugate gradient descent method requires additional computations but is generally faster than the steepest descent approach, especially for quadratic problems.

Quasi-Newton methods
As already mentioned, Quasi-Newton methods approximate the inverse of the Hessian matrix to avoid a costly inversion, and to avoid computing the Hessian matrix itself. There are several Quasi-Newton methods, but the most popular one is the Broyden-Fletcher-Goldfarb-Shanno method (BFGS). The approximation of B_k^{-1} is denoted by H_k and it follows the iteration

$$H_{k+1} = (I - \rho_k s_k y_k^\mathsf{T}) H_k (I - \rho_k y_k s_k^\mathsf{T}) + \rho_k s_k s_k^\mathsf{T},$$

with

$$\rho_k = \frac{1}{y_k^\mathsf{T} s_k}.$$

Quasi-Newton methods are generally faster than both gradient and conjugate gradient methods, especially for quadratic problems, but the computational effort is significantly higher and the distribution of the computation is difficult, if at all possible.

A.2 Basics of graph theory

This section gives a short introduction into the main graph theoretic concepts relevant to this thesis. For an in-depth introduction to the topic, the reader can consult the textbooks [83, 136, 137]. This overview is based on these textbooks.

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of the two sets $\mathcal{V}$ and $\mathcal{E}$ where the elements of $\mathcal{V}$ are the vertices or nodes, and the elements of $\mathcal{E}$ are the edges. The cardinality of the set $\mathcal{V}$ determines the number of nodes and with that the size of the graph N. Each edge has a set of two associated vertices which it joins to make up the structure of the graph. The two nodes v_i and v_j connected by an edge $e = (v_i, v_j)$ are called the endpoints of e and the nodes are said to be incident on e. Also, v_i and v_j are adjacent to each other in that case. A clique in a graph $\mathcal{G}$ is a subset of the vertices $\mathcal{C} \subseteq \mathcal{V}$ such that every combination of two vertices in the set is adjacent. A maximal clique is a clique that cannot be extended by including an additional node.

Each vertex v has a degree associated with it which is the number of edges incident on v and is denoted by $\deg(v)$. The adjacency matrix $A_{\mathcal{G}} \in \mathbb{R}^{N\times N}$ for a graph $\mathcal{G}$ is defined as

$$A_{\mathcal{G}}(i,j) = \begin{cases} 1, & \text{if } v_i \text{ and } v_j \text{ are adjacent,} \\ 0, & \text{otherwise.} \end{cases}$$

The adjacency matrix is enough to describe the structure of the graph. Additionally, we can give the degree matrix $D_{\mathcal{G}}$ which is given by

$$D_{\mathcal{G}}(i,j) = \begin{cases} \deg(v_i), & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

With this, the Laplacian matrix $L_{\mathcal{G}} \in \mathbb{R}^{N\times N}$ is

$$L_{\mathcal{G}} = D_{\mathcal{G}} - A_{\mathcal{G}}. \tag{A.5}$$

Because of the way it is constructed, clearly all rows of $L_{\mathcal{G}}$ sum to zero. Therefore, it has at least one zero eigenvalue. It can also be shown that it is positive semidefinite and we have the relationship

$$\lambda_1(L_{\mathcal{G}}) \leq \lambda_2(L_{\mathcal{G}}) \leq \cdots \leq \lambda_N(L_{\mathcal{G}}).$$

If the graph is connected, we have $\lambda_2(L_{\mathcal{G}}) > 0$. In fact, $\lambda_2(L_{\mathcal{G}})$ is a well-known graph measure and is called the algebraic connectivity of $\mathcal{G}$.

A different graph measure is the clustering coefficient C defined as

$$C = \frac{(\text{number of triangles}) \times 3}{(\text{number of connected triples})}.$$

Furthermore, there are several measures of centrality: Degree centrality, betweenness centrality, closeness centrality and eigenvalue centrality. Degree centrality corresponds identically to the degree of a node. Next, we define the two numbers g_{st}, which denotes the number of shortest paths between nodes s and t, and n_{st}^i, which denotes the number of those paths that pass through node i. Then, the betweenness centrality is defined as

$$b_i = \sum_{s\neq t\neq i} \frac{n_{st}^i}{g_{st}}.$$

Betweenness centrality is, therefore, a measure of how often a vertex lies on paths between other vertices. A different notion is given by closeness centrality which describes the mean distance from a vertex to other vertices. If the number d_{ij} denotes the length of the shortest path from node i to node j, then the mean distance from i to all other nodes is

$$l_i = \frac{1}{n}\sum_j d_{ij}$$

and the inverse of this value is the closeness centrality given by

$$c_i = \frac{1}{l_i} = \frac{n}{\sum_j d_i j}.$$

Last, the eigenvalue centrality of node i is given by

$$e_i = \frac{1}{\lambda_M(A_{\mathcal{G}})} \sum_j A_{\mathcal{G},ij} x_j,$$

where $\lambda_M(A_{\mathcal{G}})$ is the largest eigenvalue of the adjacency matrix $A_{\mathcal{G}}$.

A.3 Model reduction using balanced truncation

In this section, a brief overview on balanced truncation is given. For details, we refer the reader to [88] and the references therein.

The goal of balanced truncation is the following. Given a system with transfer function

$$G(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right],$$

where $A \in \mathbb{R}^{n\times n}$, we wish to find a reduced model

$$G_{\mathrm{r}}(s) = \left[\begin{array}{c|c} A_{\mathrm{r}} & B_{\mathrm{r}} \\ \hline C_{\mathrm{r}} & D_{\mathrm{r}} \end{array}\right],$$

where $A_{\mathrm{r}} \in \mathbb{R}^{r\times r}$ with $r < n$ such that

$$||G - G_{\mathrm{r}}||_\infty$$

is small. The first step in balanced truncation is the computation of the controllability gramian X_{c} and the observability gramian Y_{o}, which are the solutions to the following Lyapunov equations

$$AX_{\mathrm{c}} + X_{\mathrm{c}}A^{\mathsf{T}} + BB^{\mathsf{T}} = 0, \tag{A.6a}$$
$$A^{\mathsf{T}}Y_{\mathrm{o}} + Y_{\mathrm{o}}A + C^{\mathsf{T}}C = 0. \tag{A.6b}$$

In general, X_{c} and Y_{o} are different from each other. It can be shown, however, that there is always a coordinate transformation to obtain a so-called balanced realization where $\tilde{X}_{\mathrm{c}} = \tilde{Y}_{\mathrm{o}} = \Sigma$, where $\Sigma \succ 0$ has the diagonal form

$$\Sigma = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix}.$$

The diagonal entries σ_i are called Hankel singular values and they can be ordered such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n.$$

If we assume that there is a strict inequality for $r < n$ such that

$$\sigma_{r+1} < \sigma_r,$$

we can partition the original system as

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad C = \begin{bmatrix} C_1 & C_2 \end{bmatrix},$$

where $A_{11} \in \mathbb{R}^{r \times r}$, and the reduced order model of size r is given by

$$G_{\mathrm{r}}(s) = \left[\begin{array}{c|c} A_{11} & B_1 \\ \hline C_1 & 0 \end{array}\right].$$

It can then be shown, see [88], that this realization is again balanced with the Hankel singular values $\sigma_1, \ldots, \sigma_r$. Furthermore, by denoting the distinct values among $\sigma_{r+1}, \ldots, \sigma_n$ by $\sigma_1^t, \ldots, \sigma_k^t$, i.e. by removing repetitions, the following bound on the approximation error can be given:

$$||G - G_{\mathrm{r}}||_\infty \leq 2(\sigma_1^t + \cdots + \sigma_k^t).$$

Instead of solving the Lyapunov equations (A.6), one can also solve the more general Lyapunov inequalities given by

$$\begin{aligned} AX_{\mathrm{c,g}} + X_{\mathrm{c,g}}A^{\mathsf{T}} + BB^{\mathsf{T}} &\preceq 0, \\ A^{\mathsf{T}}Y_{\mathrm{o,g}} + Y_{\mathrm{o,g}}A + C^{\mathsf{T}}C &\preceq 0, \end{aligned}$$

and their solutions $X_{\mathrm{c,g}}$ and $Y_{\mathrm{o,g}}$ are called the generalized controllability and observability gramians. It is then possible to balance the generalized gramians in the same way as the system gramians X_{c} and Y_{o}. Then, one has the generalized Hankel singular values $\gamma_1 \geq \gamma_2 \geq \gamma_n$ where we have

$$\gamma_i \geq \sigma_i, \; i = 1, \ldots, n.$$

By removing duplicates again and by using the same notation as before, one can determine a truncated system model $G_{\mathrm{r,g}}$ in the same way based on generalized gramians and obtains the error bound

$$||G - G_{\mathrm{r,g}}||_\infty \leq 2(\gamma_1^t + \cdots + \gamma_k^t).$$

The benefit of using generalized gramians is twofold. First, LMIs are better suited for a distributed solution than equations. Second, the additional degree of freedom allows to arrange the generalized Hankel singular values to repeat more often and thus to decrease the error bound. The second point, however, is far from being trivial and for us, the first point is more relevant.

Bibliography

[1] D. J. Hill and G. Chen. “Power systems as dynamic networks.” In: *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*. 2006, pp. 722–725.

[2] M. Cantoni et al. “Control of large-scale irrigation networks.” In: *Proceedings of the IEEE* 95 (2007), pp. 75–91.

[3] R.R. Negenborn and H. Hellendoorn. “Intelligence in transportation infrastructures via modelbased predictive control.” In: *Intelligent Infrastructures*. Ed. by R.R. Negenborn, Z. Lukszo, and H. Hellendoorn. Springer, 2010, pp. 3–24.

[4] T. Salsbury. “A survey of control technologies in the building automation industry.” In: *Proc. 16th IFAC World Congress*. 2005, pp. 1396–1407.

[5] D. G. MacMartin et al. “Dynamic analysis of the actively-controlled segmented mirror of the thirty meter telescope.” In: *IEEE Transactions on Control Systems Technology* 22 (2014), pp. 58–68.

[6] K. D. Frampton, O. N. Baumann, and P. Gardonio. “A comparison of decentralized, distributed, and centralized vibro-acoustic control.” In: *The Journal of the Acoustical Society of America* 128.5 (2010), pp. 2798–2806.

[7] S. H. Low, F. Paganini, and J. C. Doyle. “Internet congestion control.” In: *IEEE Control Systems* 22 (2002), pp. 28–43.

[8] W. Ren and N. Sorensen. “Distributed coordination architecture for multi-robot formation control.” In: *Robotics and Autonomous Systems* 56.4 (2008), pp. 324–333.

[9] S. A. Levin et al. “Mathematical and computational challenges in population biology and ecosystems science.” In: *Science* 275.5298 (1997), pp. 334–343.

[10] A. Nagurney et al. “Dynamics of supply chains: a multilevel (logistical-informational-financial) network perspective.” In: *Environment and Planning B* 29.6 (2002), pp. 795–818.

[11] G. Iñiguez et al. “Opinion and community formation in coevolving networks.” In: *Physical Review E* 80.6 (2009), p. 066119.

[12] P. Kundur, N. J. Balu, and M. G. Lauby. *Power System Stability and Control*. McGraw-Hill New York, 1994.

[13] B. D. O. Anderson et al. “UAV formation control: theory and application.” In: *Recent advances in learning and control*. Ed. by V. D. Blondel, S. P. Boyd, and H. Kimura. Springer, 2008, pp. 15–33.

[14] J. A. Fax and R. M. Murray. “Information flow and cooperative control of vehicle formations.” In: *IEEE Transactions on Automatic Control* 49.9 (2004), pp. 1465–1476.

[15] D. D. Siljak. *Decentralized Control of Complex Systems*. Courier Dover Publications, 2011.

[16] M. D. Ilic. “From hierarchical to open access electric power systems.” In: *Proceedings of the IEEE* 95.5 (2007), pp. 1060–1084.

[17] J. Wang et al. “Cross-layer optimization in TCP/IP networks.” In: *IEEE/ACM Transactions on Networking* 13.3 (2005), pp. 582–595.

[18] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and design*. Vol. 2. Wiley New York, 2007.

[19] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice Hall New Jersey, 1996.

[20] F. Deroo et al. “Distributed stability tests for large-scale systems with limited model information.” In: *IEEE Transactions on Control of Network Systems* 2.3 (2015), pp. 298–309.

[21] F. Deroo et al. “Accelerated iterative Distributed Controller Synthesis with a Barzilai-Borwein Step Size.” In: *Proc. 51st IEEE Conference on Decision and Control (CDC)*. 2012, pp. 4864–4870.

[22] F. Deroo et al. “Distributed controller design for a class of sparse singular systems with privacy constraints.” In: *Proc. 4th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys)*. 2013, pp. 190–197.

[23] F. Deroo et al. “Distributed control design with local model information and guaranteed stability.” In: *Proc. 19th IFAC World Congress*. 2014, pp. 4010–4017.

[24] D. Sieber, F. Deroo, and S. Hirche. “Iterative optimal feedback controller design under relaxed rigidity constraints for cooperative manipulation.” In: *Proc. 52nd IEEE Conference on Decision and Control (CDC)*. 2013, pp. 971–976.

[25] D. Sieber, F. Deroo, and S. Hirche. “Formation-based approach for cooperative manipulation based on iterative optimal controller design.” In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2013, pp. 5227–5233.

[26] F. Deroo and S. Hirche. “Stabilizing distributed control design under model data privacy constraints.” In: *to be submitted* (2016).

[27] F. Deroo, S. Hirche, and B. D. O. Anderson. “Spatial decay analysis in interconnected dynamical systems using vector Lyapunov functions.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 3654–3660.

[28] K. J. Aström and R. M. Murray. *Feedback systems: An introduction for Scientists and Engineers*. Princeton University Press, 2010.

[29] A. M. Lyapunov. “The general problem of the stability of motion.” PhD thesis. Kharkov University, 1892.

[30] H. K. Khalil. *Nonlinear Systems*. 3rd ed. Prentice hall, 2002.

[31] E. Kaszkurewicz and A. Bhaya. *Matrix Diagonal Stability in Systems and Computation*. Springer, 2000.

[32] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási. “Controllability of complex networks.” In: *Nature* 473.7346 (2011), pp. 167–173.

[33] H. J. Sussmann and J. C. Willems. “300 years of optimal control: from the brachystochrone to the maximum principle.” In: *IEEE Control Systems* 17.3 (1997), pp. 32–44.

[34] B. D. O. Anderson and J. B. Moore. *Linear optimal control*. Vol. 197. Prentice-Hall Englewood Cliffs, NJ, 1971.

[35] B. D. O. Anderson and J. B. Moore. *Optimal control: Linear quadratic methods*. Courier Dover Publications, 2007.

[36] L. Bakule. “Decentralized control: an overview.” In: *Annual reviews in control* 32 (2008), pp. 87–98.

[37] H. S. Witsenhausen. “A counterexample in stochastic optimum control.” In: *SIAM Journal on Control* 6 (1968), pp. 131–147.

[38] Y.-C. Ho and K.-C. Chu. “Team decision theory and information structures in optimal control problems–Part I.” In: *IEEE Transactions on Automatic Control* 17 (1972), pp. 15–22.

[39] M. Rotkowitz and S. Lall. “A characterization of convex problems in decentralized Control.” In: *IEEE Transactions on Automatic Control* 51.2 (2006), pp. 274–286.

[40] L. Lessard and S. Lall. “Quadratic invariance is necessary and sufficient for convexity.” In: *Proc. American Control Conference (ACC)*. 2011, pp. 5360–5362.

[41] J. Swigart and S. Lall. “Optimal synthesis and explicit state-space solution for a decentralized two-player linear-quadratic regulator.” In: *Proc. 49th IEEE Conference on Decision and Control (CDC)*. 2010, pp. 132–137.

[42] P. Shah and P. A. Parrilo. “H2-optimal decentralized control over posets: a state space solution for state-feedback.” In: *Proc. 49th IEEE Conference on Decision and Control (CDC)*. 2010, pp. 6722–6727.

[43] P. Shah and P. A. Parrilo. “An optimal controller architecture for poset-causal systems.” In: *Proc. 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. 2011, pp. 5522–5528.

[44] C. Langbort, R.S. Chandra, and R. D’Andrea. “Distributed control design for systems interconnected over an arbitrary graph.” In: *IEEE Transactions on Automatic Control* 49.9 (2004), pp. 1502–1519.

[45] A. S. M. Vamsi and N. Elia. “Optimal realizable networked controllers for networked systems.” In: *Proc. American Control Conference (ACC)*. 2011, pp. 336–341.

[46] P. Massioni and M. Verhaegen. “Distributed control for identical dynamically coupled systems: a decomposition approach.” In: *IEEE Transactions on Automatic Control* 54 (2009), pp. 124–135.

[47] F. Borrelli and T. Keviczky. “Distributed LQR design for identical dynamically decoupled systems.” In: *IEEE Transactions on Automatic Control* 53.8 (2008), pp. 1901–1912.

[48] M. Fardad, F. Lin, and M. R. Jovanovic. “On the dual decomposition of linear quadratic optimal control problems for vehicular formations.” In: *Proc. 49th IEEE Conference on Decision and Control (CDC)*. 2010, pp. 6287–6292.

[49] F. Lin, M. Fardad, and M. R. Jovanovic. “Design of optimal sparse feedback gains via the alternating direction method of multipliers.” In: *IEEE Transactions on Automatic Control* 58.9 (2013), pp. 2426–2431.

[50] A. Das and F. L. Lewis. “Distributed adaptive control for synchronization of unknown nonlinear networked systems.” In: *Automatica* 46.12 (2010), pp. 2014–2021.

[51] W. Yu et al. “Distributed adaptive control of synchronization in complex networks.” In: *IEEE Transactions on Automatic Control* 57.8 (2012), pp. 2153–2158.

[52] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis. “Reinforcement learning and feedback control: using natural decision methods to design optimal adaptive controllers.” In: *IEEE Control Systems* 32.6 (2012), pp. 76–105.

[53] A. Gusrialdi and S. Hirche. “Performance-oriented communication topology design for large-scale interconnected systems.” In: *Proc. 49th IEEE Conference on Decision and Control (CDC)*. 2010, pp. 5707–5713.

[54] S. Schuler et al. “Controller structure design for decentralized control of coupled higher order subsystems.” In: *Proc. 2nd IFAC Workshop on Distributed Estimation and Control of Networked Systems (NecSys)*. 2010, pp. 269–274.

[55] C. Dwork. “Differential privacy: a survey of results.” In: *Theory and Applications of Models of Computation*. Ed. by M. Agrawal et al. Springer, 2008, pp. 1–19.

[56] S. Han, U. Topcu, and G. J. Pappas. “Differentially private convex optimization with piecewise affine objectives.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 2160–2166.

[57] J. Le Ny and G. J. Pappas. “Differentially private filtering.” In: *IEEE Transactions on Automatic Control* 59.2 (2014), pp. 341–354.

[58] Z. Huang et al. “On the cost of differential privacy in distributed control systems.” In: *Proc. 3rd International Conference on High Confidence Networked Systems (HiCoNS)*. 2014, pp. 105–114.

[59] S. Pequito et al. “Design of communication networks for distributed computation with privacy guarantees.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 1370–1376.

[60] Y. Mo and R. M. Murray. “Privacy preserving average consensus.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 2154–2159.

[61] S. S. Kia, J. Cortés, and S. Martinez. “Dynamic average consensus under limited control authority and privacy requirements.” In: *International Journal of Robust and Nonlinear Control* 25 (2015), pp. 1941–1966.

[62] J. Yao and P. Venkitasubramaniam. “The privacy analysis of battery control mechanisms in demand response: revealing state approach and rate distortion bounds.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 1377–1382.

[63] J. Giraldo et al. “Delay and sampling independence of a consensus algorithm and its application to smart grid privacy.” In: *Proc. 53rd IEEE Conference on Decision and Control (CDC)*. 2014, pp. 1389–1394.

[64] J. Le Ny and G. J. Pappas. “Privacy-preserving release of aggregate dynamic models.” In: *Proc. 2nd international conference on High Confidence Networked Systems (HiCoNS)*. 2013, pp. 49–56.

[65] N. Martins. “Efficient eigenvalue and frequency response methods applied to power system small-signal stability studies.” In: *IEEE Transactions on Power Systems* (1986), pp. 217–224.

[66] Y. Nesterov. “Smooth minimization of non-smooth functions.” In: *Mathematical Programming* 103.1 (2005), pp. 127–152.

[67] I. Necoara and J. Suykens. “Application of a smoothing technique to decomposition in convex optimization.” In: *IEEE Transactions on Automatic Control* 53.11 (2008), pp. 2674–2679.

[68] M. Meinel, M. Ulbrich, and S. Albrecht. “A class of distributed optimization methods with event-triggered communication.” In: *Computational Optimization and Applications* 57.3 (2014), pp. 517–553.

[69] D. Carlson, D. Hershkowitz, and D. Shasha. “Block diagonal semistability factors and Lyapunov semistability of block triangular matrices.” In: *Linear Algebra and its Applications* 172 (1992), 1–25.

[70] A. Berman, F. Goldberg, and R. Shorten. “Comments on Lyapunov α-stability with some extensions.” In: *Variational and Optimal Control Problems on Unbounded Domains*. American Mathematical Society, 2014, pp. 19–29.

[71] P. Moylan and D. Hill. “Stability criteria for large-scale systems.” In: *IEEE Transactions on Automatic Control* 23.2 (1978), pp. 143–149.

[72] M. Vidyasagar. *Input-Output Analysis of Large-Scale Interconnected Systems*. Lecture Notes in Control and Information Sciences. Springer, 1981.

[73] R. H. Gielen and M. Lazar. "Non-conservative dissipativity and small-gain conditions for stability analysis of interconnected systems." In: *Proc. 51st IEEE Conference on Decision and Control (CDC)*. 2012, pp. 4187–4192.

[74] A. Rantzer. "Distributed control of positive systems." In: *Proc. 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. 2011, pp. 6608–6611.

[75] R. Pates and G. Vinnicombe. "Stability certificates for networks of heterogeneous linear systems." In: *Proc. 51st IEEE Conference on Decision and Control (CDC)*. 2012, 6915–6920.

[76] S. Z. Khong and A. Rantzer. "Scalable stability conditions for heterogeneous networks via integral quadratic constraints." In: *Proc. European Control Conference (ECC)*. 2014, 2863–2867.

[77] R. P. Mason and A. Papachristodoulou. "Chordal sparsity, decomposing SDPs and the Lyapunov equation." In: *Proc. American Control Conference (ACC)*. 2014, pp. 531–537.

[78] J. Nocedal and S. J. Wright. *Numerical Optimization*. 2nd ed. Springer New York, 2006, pp. 497–528.

[79] I. Necoara and V. Nedelcu. "Rate analysis of inexact dual first order methods. Application to dual decomposition." In: *IEEE Transactions on Automatic Control* 49.5 (2014), 1232–1243.

[80] R. Olfati-Saber, J. A. Fax, and R. M. Murray. "Consensus and cooperation in networked multi-agent systems." In: *Proceedings of the IEEE* 95 (2007), pp. 215–233.

[81] I. Necoara and V. Nedelcu. "On linear convergence of a distributed dual gradient algorithm for linearly constrained separable convex problems." In: *Automatica* 55 (2014), pp. 209–216.

[82] S. Kim et al. "Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion." In: *Mathematical Programming* 129 (2011), pp. 33–68.

[83] J. L. Gross and J. Yellen. *Handbook of Graph Theory*. CRC press, 2004.

[84] J. Löfberg. "YALMIP: a toolbox for modeling and optimization in MATLAB." In: *Proc. CACSD Conference*. 2004. URL: `http://users.isy.liu.se/johanl/yalmip`.

[85] E. Camponogara et al. "Distributed model predictive control." In: *IEEE Control Systems* 22 (2002), pp. 44–52.

[86] R. Christie. *IEEE 30 Bus Test Case*. `http://www.ee.washington.edu//research/pstca/pf30/pg_tca30bus.htm/`. 1993.

[87] D. Hershkowitz. "Recent directions in matrix stability." In: *Linear Algebra and its Applications* 171 (1992), pp. 161–186.

[88] G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory: A Convex Approach*. Vol. 36. Springer New York, 2000.

[89] H. Sandberg and R. M. Murray. "Model reduction of interconnected linear systems using structured gramians." In: *Proc. 17th IFAC World Congress*. 2008, pp. 8725–8730.

[90] A. Tahbaz-Salehi and A. Jadbabaie. "A one-parameter family of distributed consensus algorithms with boundary: from shortest paths to mean hitting times." In: *Proc. 45th IEEE Conference on Decision and Control (CDC)*. 2006, pp. 4664–4669.

[91] P. Giselsson and A. Rantzer. "Distributed model predictive control with suboptimality and stability guarantees." In: *Proc. 49th IEEE Conference on Decision and Control (CDC)*. 2010, pp. 7272–7277.

[92] P. Giselsson et al. "Accelerated gradient methods and dual decomposition in distributed model predictive control." In: *Automatica* 49.3 (2013), pp. 829–833.

[93] R. Scattolini. "Architectures for distributed and hierarchical model predictive control – a review." In: *Journal of Process Control* 19.5 (2009), pp. 723–731.

[94] I. Necoara, V. Nedelcu, and I. Dumitrache. "Parallel and distributed optimization methods for estimation and control in networks." In: *Journal of Process Control* 21.5 (2011), pp. 756–766.

[95] C. Langbort and J. Delvenne. "Distributed design methods for linear quadratic control and their limitations." In: *IEEE Transactions on Automatic Control* 55.9 (2010), pp. 2085–2093.

[96] F. Farokhi, C. Langbort, and K. H. Johansson. "Optimal structured static state-feedback control design with limited model information for fully-actuated systems." In: *Automatica* 49.2 (2013), pp. 326–337.

[97] F. Farokhi and K. H. Johansson. "Optimal control design under limited model information for discrete-time linear systems with stochastically-varying parameters." In: *IEEE Transactions on Automatic Control* 60.3 (2015), pp. 684–699.

[98] K. Martensson and A. Rantzer. "A scalable method for continuous-time distributed control synthesis." In: *Proc. American Control Conference (ACC)*. 2012, pp. 6308–6313.

[99] I. Shames and M. Cantoni. "Constrained linear quadratic control in networks with limited model-information sharing." In: *21st International Symposium on Mathematical Theory of Networks and Systems (MTNS)*. 2014, pp. 29–34.

[100] A. A. Alam, A. Gattami, and K. H. Johansson. "Suboptimal decentralized controller design for chain structures: Applications to vehicle formations." In: *Proc. 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. 2011, pp. 6894–6900.

[101] H. Chen and F. Allgöwer. "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability." In: *Automatica* 34.10 (1998), pp. 1205–1217.

[102] R. R. Bitmead, M. Gevers, and V. Wertz. *Adaptive Optimal Control: The Thinking Man's GPC*. Prentice Hall, 1990.

[103] R. M. Hermans. "Distributed control of deregulated electrical power networks." PhD thesis. Ph. D. thesis, Eindhoven University of Technology, The Netherlands, 2012.

[104] R. V. Bobiti, R. H. Gielen, and M. Lazar. "Non-conservative and tractable stability tests for general linear interconnected systems with an application to power systems." In: *Proc. 4th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys)*. Vol. 4. 2013, pp. 152–159.

[105] J. Barzilai and J. M. Borwein. "Two-point step size gradient methods." In: *IMAJ of Numerical Analysis* 8 (1988), pp. 141–148.

[106] R. Fletcher. "On the Barzilai-Borwein method." In: *Optimization and Control with Applications*. Ed. by L. L. Qi, K. Teo, and X. Yang. Vol. 96. Applied Optimization. Springer US, 2005, pp. 235–256.

[107] G.C. Calafiore, L. Carlone, and M. Wei. "A distributed gradient method for localization of formations using relative range measurements." In: *Proc. IEEE International Symposium on Computer-Aided Control System Design (CACSD)*. 2010, pp. 1146–1151.

[108] L. Xiao, S. Boyd, and S. Lall. "A scheme for robust distributed sensor fusion based on average consensus." In: *Proc. Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*. 2005, pp. 63–70.

[109] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley Online Library, 2005.

[110] E. Kofman and S. Junco. "Quantized-state systems: a DEVS approach for continuous system simulation." In: *Transactions of the Society for Modeling and Simulation International* 18.3 (2001), pp. 123–132.

[111] G. Migoni, E. Kofman, and F. Cellier. "Quantization-based new integration methods for stiff ordinary differential equations." In: *Simulation* 88.4 (2011), pp. 387–407.

[112] P. Tabuada. "Event-triggered real-time scheduling of stabilizing control tasks." In: *IEEE Transactions on Automatic Control* 52.9 (2007), pp. 1680–1685.

[113] F. L. Lewis. "A survey of linear singular systems." In: *Circuits, Systems, and Signal Processing* 5 (1986), pp. 3–36.

[114] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed computation*. Prentice Hall, 1989.

[115] H. A. Van der Vorst. "Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems." In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (1992), pp. 631–644.

[116] L.T. Yang and R. P. Brent. "The improved BiCGStab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures." In: *Proc. 5th Int. Conference on Algorithms and Architectures for Parallel Proc.* 2002, pp. 324–328.

[117] F. Iavernaro and M. La Scala. "Boundary values methods for time-domain simulation of power system dynamic behavior." In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 45 (1998), pp. 50–63.

[118] D. Bender and A. Laub. "The linear-quadratic optimal regulator for descriptor systems." In: *IEEE Trans. on Automatic Cont.* 32.8 (1987), pp. 672–688.

[119] J. Liu et al. "Joint controller-communication topology design for distributed wide-area damping control of power systems." In: *Proc. 18th IFAC World Congress*. 2011, pp. 519–525.

[120] P. W. Sauer and M. A. Pai. *Power System Dynamics and Stability*. Prentice Hall, 1998.

[121] A. Kiani and A. Annaswamy. "Distributed hierarchical control for renewable energy integration in a Smart Grid." In: *Proc. IEEE PES Innovative Smart Grid Technologies*. 2012, pp. 1–8.

[122] M. W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. John Wiley & Sons, 2006.

[123] D. Williams and O. Khatib. "The virtual linkage: a model for internal forces in multi-grasp manipulation." In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. 1993, pp. 1025–1030.

[124] G. Meurant. "A review on the inverse of symmetric tridiagonal and block tridiagonal matrices." In: *SIAM Journal on Matrix Analysis and Applications* 13.3 (1992), pp. 707–728.

[125] T. Fujimoto, C. Herrero, and A. Villar. "A sensitivity analysis for linear systems involving M-matrices and its application to the Leontif model." In: *Linear Algebra and its Applications* 64 (1985), pp. 85–91.

[126] B. Bamieh et al. "Coherence in large-scale networks: dimension-dependent limitations of local feedback." In: *IEEE Transactions on Automatic Control* 57.9 (2012), pp. 2235–2249.

[127] A. Chapman, E. Schoof, and M. Mesbahi. "Distributed online topology design for disturbance rejection." In: *Proc. 52nd IEEE Conference on Decision and Control (CDC)*. 2013, pp. 817–822.

[128] F. Lin, M. Fardad, and M. R. Jovanovic. “Performance of leader-follower networks in directed trees and lattices.” In: *Proc. 51st IEEE Conference on Decision and Control (CDC)*. 2012, pp. 734–739.

[129] K. Cai, B. D. O. Anderson, and C. Yu. “Local average consensus in distributed measurement of spatial-temporal varying parameters: 1D case.” In: *Proc. 52nd IEEE Conference on Decision and Control (CDC)*. 2013, pp. 2139–2144.

[130] A. Berman and R. J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. SIAM, 1979.

[131] D. D. Siljak. *Large-Scale Dynamic Systems: Stability and Structure*. North-Holland, 1978.

[132] M. Ikeda and D. D. Šiljak. “On decentrally stabilizable large-scale systems.” In: *Automatica* 16.3 (1980), pp. 331–334.

[133] R. Nabben. “Decay rates of the inverse of nonsymmetric tridiagonal and band matrices.” In: *SIAM Journal on Matrix Analysis and Applications* 20.3 (1999), pp. 820–837.

[134] D. P. Bertsekas. *Nonlinear Programming*. 2nd ed. Athena Scientific, Belmont, MA, 1999.

[135] Y.-X. Yuan. “Step-sizes for the gradient method.” In: *Proc. 3rd International Congress of Chinese Mathematicians*. Vol. 42. 2. AMS IP Studies in Advanced Mathematics, 2008, pp. 785–796.

[136] M. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[137] C. Godsil and G. F. Royle. *Algebraic Graph Theory*. Springer Science & Business Media, 2013.