Methods of Distributed and Nonlinear Process Control: Structuredness, Optimality and Intelligence

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Wentao Tang

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Advised by Prodromos Daoutidis

May, 2020

Acknowledgement

Before meeting with my advisor Prof. Prodromos Daoutidis, I was not determined to do a Ph.D. in process control or systems engineering; after working with him, I could not imagine what if I had chosen a different path. The important decision that I made in the October of 2015 turned out to be most correct, that is, to follow an advisor who has the charm to always motivate me to explore the unexploited lands, overcome the obstacles and make achievements. I cherish the freedom of choosing different problems of interest and the flexibility of time managing in our research group. Apart from the knowledge of process control, I have learned a lot from Prof. Daoutidis and will try my best to keep learning, about the skills to write, present and teach, the pursuit of knowledge and methods with both elegance of fundamental theory and a promise of practical applicability, and the pride and fervor for the undertaken works. "*The high hill is to be looked up to; the great road is to be travelled on.*"[1] I want to express my deepest gratitude to him.

Prof. Qi Zhang always gives me very good suggestions and enlightenment with his abundant knowledge in optimization since he joined our department. It was a joy to consult him for fresh ideas and interchange our views on a wide variety of things. Qi is not only an intelligent researcher, a thoughtful teacher, but also a generous friend. I still owe him a countless number of meals so far.

I would like to thank my great colleagues: Nahla Alamoodi, Andrew Allman, Davood Babaei Pourkargar, Pedro Constantino, Udit Gupta, Victoria Jones, Lixia Kang, Shaaz Khatib, Jingjun Liu, Abdulla Malek, Ilias Mitrai, Nitish Mittal, Manjiri Moharir, Matthew Palys, Hanchu Wang, and Michael Zachar, with whom sharing the office space and discussing the research projects have always been very pleasing. Former members Seongmin Heo, Sujit Jogwar and Srinivas Rangarajan laid important foundations for a part of my research, and the outstanding works of Michael Baldea, Panagiotis Christofides, and Aditya Kumar introduced me to the history of our group's research when I was starting my work. I have learned many skills and great ideas from my colleagues, and our collaborations turned out to be very fruitful as reflected by our coauthored papers. I am also glad that some of my ideas will continued to be developed by younger members of our group.

"*I am not one who was born in the possession of knowledge; I am one who is fond of antiquity, and earnest in seeking it there*".[2] Thanks to the teachings of all the professors

[1] *Classic of Poetry: Minor Odes of Kingdom: Che Xia* (《詩經·小雅·車舝》).
[2] *Analects (of Confucius): Shu Er* (《論語·述而》).

that have delivered the courses in chemical engineering, control, and optimization that I have taken during the first two years before my preliminary exam, especially Professors Mihailo Jovanović, Andrew Lamperski, Zhi-Quan Luo and Shuzhong Zhang, I have gained the fundamentals of these disciplines to start my research. I would not be able to list the names of all the authors of the books, chapters, papers and tutorial materials that I have read, whose works have refreshed my understanding of the relevant areas as well as my own research over and over again, and the friends in the process systems engineering and control areas that I have talked to from time to time. During the painful processes of submitting and revising papers, I have encountered some sharp but anonymous reviewers who gave very critical but constructive comments. Their opinions forced me to enhance my understanding of many relevant problems.

Of course, this Ph.D. is built upon my previous 16 years of study. I feel obligated to express my gratitude to the education received from my great teachers. Ms. Zhu Qiaohua (朱巧華) in Lengshuitan Fourth Elementary School taught me Mathematics and told me right from wrong during the pupil years. In Nanya Middle school, Mr. Li Weiqing (李偉清) was my first formal English teacher who developed a great interest in this language. The History classes by Mr. Yin Chi'e (尹馳鍔) taught me to understand things always in combination with their history. In Yali High School, Mr. Zhang Liu's (張鎏) Mathematics was my most enjoyable course, and I became an engineering student largely due to his encouragement. Mr. Zhou Shuqiao's (周樹喬) Chinese class made me a fan of ancient literature and linguistics, and Mr. Li Yun's (李贇) Philosophy left me a deep impression of his dialectic and abstract lines of thinking. At Tsinghua University, Profs. Wentao Zhu (朱文濤), Diannan Lu (盧滇楠) and Lixin Yu (余立新) delivered great lectures in Physical Chemistry, Chemical Engineering Thermodynamics, and Principles of Chemical Engineering, respectively. When doing my second degree in Mathematics, the rigor of Prof. Tianquan Chen (陳天權)'s Real Analysis and the elegance of Profs. Meirong Zhang (章梅榮) and Huaiyu Jian's (簡懷玉) Ordinary and Partial Differential Equations attracted me a lot. I learned Russian as a Secondary Foreign Language from Prof. Min Zhang (張敏) for a year, which turned out to be helpful when I dive into the classical control theorists' papers.

Doing a Ph.D. in this frigid and strange land is not an easy journey. The doctors and nurses at Boynton Clinics, the unknown citizens who helped me to move my car out of snows, and the chestnut chicken and grilled fish at Tea House Restaurant all made me much warmer through the long-lasting winters. The help from my friends largely relieved my difficulties during my first semesters in Minnesota and after my little car went missing.

I am grateful to the fate that has brought my beloved wife Yixin Ye and me together. She has always been the one that cheers me up in sadness, accompanies me in difficulties, encourages me in frustrations, and shares every little mood – be it proud, delight, upset or anger – that floats in my mind from time to time. For so many times in the flight to Pittsburgh, I looked through the clouds, and saw a little rainbow above the ripples of Lake Michigan reflecting the sun's beams; I realized that she was awaiting at the destination. There might be more rivers and oceans for us to travel across, more streets and valleys for us to walk through, more gains and losses coming into our lives, but the meaning of this journey is but a journey together with her. Due to the very apparent reason, I am returning home less frequently, but my families' unwavering love can always be felt. Born in an ordinary family, I understand that it was uneasy for my parents to raise me up and find me good education faraway from home for years. Knowing that they are well has always been the biggest motivation for my hardworking, although not being able to do much for them is in turn a source of guilt. And when missing my small hometown in southwestern Hunan, I would read in my heart a poem of it:[3]

High upon Mount Jiuyi, white clouds diffuse;	九嶷山上白雲飛，
Down to the green hills, by wind descended the Princesses.	帝子乘風下翠微。
They sparkled the wild bamboos with their tears so profuse,	斑竹一枝千滴淚，
when the clouds turned rosy behind their layers of dresses.	紅霞萬朵百重衣。
Waves of Dongting surge against the overwhelming snows;	洞庭波湧連天雪，
Folks of Long Isle chant to the rhyme of earth-shaking poems.	長島人歌動地詩。
And when I was lost in this broad and untrammeled dream,	我欲因之夢寥廓，
the land of hibiscus is covered with the morning sun's gleam.	芙蓉國裏盡朝暉。

[3]Mao Zedong, *A Rhyme of Qilü: Reply to a Friend* (毛澤東《七律·答友人》), 1961.

To my wife and my parents

Abstract

Chemical processes are intrinsically nonlinear and often integrated into large-scale networks, which are difficult to control effectively. The traditional challenges faced by process control, as well as the modern vision of transitioning industries into a smart manufacturing paradigm, requires the instillation of new perspectives and application of new methods to the control of chemical processes. The goal is to realize highly automated, efficient, well-performing and flexible control strategies for nonlinear, interconnected and uncertain systems. Motivated by this, in this thesis, the following three important aspects (objectives) for contemporary process control – structuredness, optimality, and intelligence – are discussed in the corresponding three parts.

1. For the control of process networks in a structured and distributed manner, a network-theoretic perspective is introduced, which suggests to find a decomposition of the problem according to the block structures in the network. Such a perspective is examined by sparse optimal control of Laplacian network dynamics. Community detection-based methods are proposed for input–output bipartite and variable-constraint network representations and applied to a benchmark chemical process.

2. For the optimality of control, we first derive a computationally efficient algorithm for nonconvex constrained distributed optimization with theoretically provable convergence properties – ELLADA, which is applied to distributed nonlinear model predictive control of a benchmark process system. We derive bilevel optimization formulations for the Lyapunov stability analysis of nonlinear systems, and stochastic optimization for optimally designing the Lyapunov function, which can be further integrated with the optimal process design problem.

3. Towards a more intelligent diagram of process control, we first investigate an advantageous Lie-Sobolev nonlinear system identification scheme and its effect on nonlinear model-based control. For model-free data-driven control, we discuss a distributed implementation of the adaptive dynamic programming idea. For chemical processes where states are mostly unmeasurable, dissipativity learning control (DLC) is proposed as a suitable framework of input–output data-driven control, and applied to several nonlinear processes. Its theoretical foundations are also discussed.

Contents

List of Tables

List of Figures

Chapter 1
Introduction

In the South grows the tall trees, that one cannot shelter under.	南有喬木，不可休息。
By River Han are the wandering ladies, that one cannot chase after.	漢有游女，不可求思。
So broad is Han, that one cannot dive across.	漢之廣矣，不可泳思。
So long is Jiang (Yangtze), that one cannot raft along.	江之永矣，不可方思。

"*Classic of Poetry: Odes of Southern Zhou: Han Guang*" (excerpt), before 6th century B.C.
《詩經·周南·漢廣》

The history of using control mechanisms for industrial purposes dates back to Watt's steam engine, and for chemical engineering, the research on automatic control has also lasted for more than a century. It is not until the time around the Second World War that the control theory is equipped with its basic tools and established as a science.[1] With the scientific foundations of chemical engineering laid down by the pioneering works in process modeling and dynamic analysis, control theory was appreciated by chemical engineers in operating chemical plants, and process control became a core discipline of chemical engineering [377]. With the rapid growth in computational power brought by the development of digital computers, process control constitutes a part of process systems engineering (PSE) together with process simulation, optimization, etc. [395]. Nowadays, there is already a sufficiently rich set of theories in control, and the industrial practice of process control has also experienced many renovations, from traditional proportional-integral-differential controllers to dynamic matrix control (DMC) [71], model predictive control (MPC) [344] and nonlinear control strategies [30].

Chemical processes are intrinsically nonlinear (or even highly nonlinear) dynamical systems and tend to be large-scale and multi-scale due to the tight couplings by mass and energy integration. Uncertainties may exist both in the underlying models, originating from simplified or inaccurate kinetics and thermodynamics, and in exogenous disturbances. The control of chemical processes also should typically be carried out without violation of operational constraints and should be favorable with respect to economic criteria. It was pointed out since the 1970s that these challenges have left an apparent gap between the research and practice of process control [116, 378]. This gap has stimulated the emergence and development of many different streams of process

[1] "*Conflict between governments with the use of force greatly accelerated the development of … this new science, so important to modern warfare.*" Hsue-Shen Tsien, Preface to *Engineering Cybernetics*, 1954, McGraw-Hill.

control theories, such as robust control after 1980s [275], geometric nonlinear control after 1990s [204, 205], and MPC [344] in the recent two decades.

The increase in the challenges on process control is an ongoing trend, as the process systems are more and more integrated and intensified, the process technologies are more and more innovative, and the economic environment of process operations is more fluctuating. On the other hand, the developments in the information, communication and computation technologies provide the vision of a new paradigm of smart manufacturing [78] and even a new industrial revolution involving artificial intelligence (AI), Internet of Things (IoT), and quantum computing.[2] It is desirable that the process control system can serve the chemical plants in a way that is (i) more distributed, structural and flexible but coordinated, (ii) economically favorable and well tuned but computationally efficient, (iii) self-adaptive and self-learning but preserving the basic performances of control such as stability.

Based on this background, in this thesis, three aspects that are of interest to the current process control theory will be discussed, namely *structuredness*, *optimality*, *intelligence*. Next, let us discuss these three aspects in detail.

1.1 Structuredness

It is well known that chemical processes, even as small as a single unit, have interactions among multiple inputs and outputs. To pursue higher efficiency of utilizing materials and energy, chemical processes are nowadays designed to be large-scale and integrated systems, which is a rule rather than exception [18]. Pairing the inputs and outputs into weakly coupled single loops for subsequent decentralized (PID) controller tuning, known as *control structure design*, has been a distinctive problem of process control since the 1960s [354]. Such pairing is based on the interaction analysis between the inputs and outputs using the information of process dynamics (transfer functions) [258, 134]. After the 1970s, the development of efficient graph-theoretic algorithms encouraged electrical engineers to decompose nonlinear systems, represented as graphs, to design decentralized control [380]. The design of plant-wide control structures has been an

[2]"*The possibilities of billions of people connected by mobile devices, with unprecedented processing power, storage capacity, and access to knowledge, are unlimited. And these possibilities will be multiplied by emerging technology breakthroughs in fields such as artificial intelligence, robotics, the Internet of Things, autonomous vehicles, 3-D printing, nanotechnology, biotechnology, materials science, energy storage, and quantum computing.*" Klaus Schwab. *The fourth industrial revolution: What it means, how to respond*, 2015, World Economic Forum.

extensively considered, yet not fully solved problem [394, 386]. The dynamics of chemical processes are too complex to be captured by only a few quantitative indices, and hence the guidelines for the control structure design are usually heuristic [243]. The underlying graph topology of processes is also usually so connected that a decomposition to smaller components is infeasible.

The quest for flexible and efficient algorithms for the control of large-scale interconnected systems has resulted in the emergence of *distributed control* [73, 11], and with the prevalence of MPC in process control, distributed MPC relying on *distributed optimization* algorithms [44]. That is, the control or its underlying optimization problems are solved by multiple agents (solvers) in coordination. Each agent is in charge of one of the topologically distributed subsystems, and through necessary information exchange, the agents coordinate their control actions either autonomously or depending on a superposed coordinator. Therefore, the problem of interest, which will be the focus of study in Part I, is stated as:

Question 1.1. *Given the dynamic model of a chemical process, how to find a desirable decomposition for its distributed control?*

The prospective here is that *network theory* can be utilized to generate such decompositions. This new science, rapidly developing after the millennium, offers new insights into and tools for macroscopic and statistical properties and rules unexplored by classical graph theory [21]. By representing the structure of the system of interest as possibly complex graphs (networks), we can exploit the tools in network science to detect the *community structures* in the networks for designing the control architectures. The community structures are sub-networks with dense interconnections inside but loose interconnections between. An illustration is given in Fig. 1.1. This pattern allows the resulting distributed control algorithms to concentrate the complexities inside the agents (corresponding to communities) and require low effort in the coordination.

With the network-theoretic perspective, the following two smaller questions are then needed to be answered for Question 1.1.

- Generally for networks, what kind of role do communities play in certain properties of control, and vice versa, does the requirement of such properties of control inevitably results in community structures in the organization of networks?

- For chemical process systems, how to represent them as networks by defining nodes, edges and meaningful interaction measures, and how to decompose them

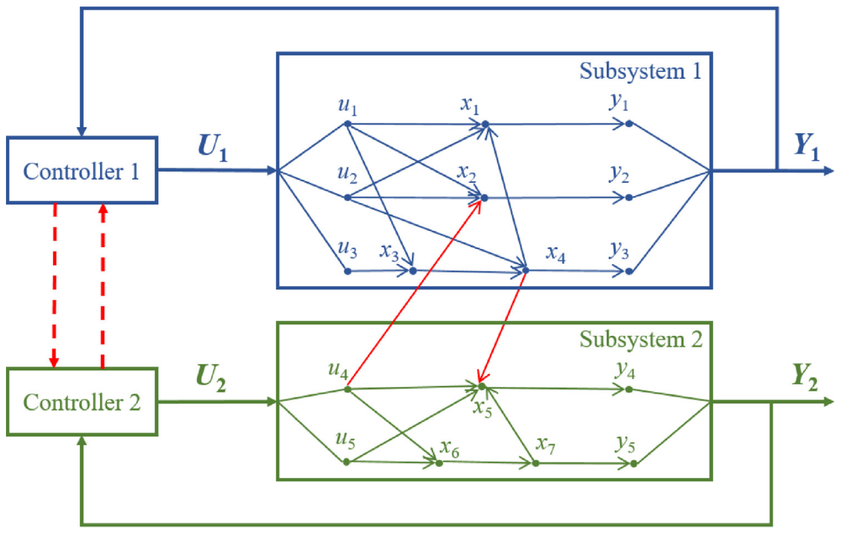


Figure 1.1: Network decomposition of a control problem into a distributed architecture

through community detection?

The contributions here include:

- In Chapter 2, by investigating a prototypical Laplacian dynamics on networks under sparse feedback control, it is pointed out that community structures play a crucial role in lowering the total control cost when the controller sparsity or complexity contributes a significant part of the cost.
- In Chapter 3, by simultaneously varying the network topology and the sparse feedback controller in Laplacian dynamics, it is observed that when taking into consideration the cost of both controller and network sparsity, the optimal network topology turned into one that has community structures.
- In Chapter 4, an input–output bipartite network representation is constructed based on short-time interaction measures among the inputs, states and outputs. Community detection in this bipartite graph generates both topologically distributed and physically meaningful decompositions.
- In Chapter 5, the input–output interaction measures are defined by the relative time-averaged gain array (RTAGA), which captures the output responses over the inputs in a user-defined time scale. Simulation on a reactor-separator process is performed.

- In Chapter 6, the control problem is decomposed in its optimization formulation in the context of MPC. The optimization problem is first represented as a bipartite graph of its variables and constraints, and then partitioned according to the variable-constraint communities.

Remark 1.1. *Based on the work of Chapter 2, a more detailed study was carried out in [69], where the centrality properties of networks are also found to play an important role in highly sparse control. The elucidation of the role of network structures in control helps to understand the ubiquitous existence and evolutionary origin of such structures in the biological world. This was illustrated with a genetic algorithm-based evolutionary experiments of networks under sparse optimal control in [68].*

Remark 1.2. *The proposed approaches of decomposing control systems through community detection were developed on the basis of the hierarchical clustering-based approaches [153, 152]. Other community detection-based approaches were proposed in parallel with or after the publication of the papers pertinent to this thesis [180, 179, 329, 469]. Extensions to PDE systems were made in [267, 269], and applications and case studies were carried out in [327, 328, 268]. Extensions to pure optimization problems (mixed-integer programming problems, such as those originating from process design rather than process control) were considered in [2, 3]. A software implementation, named DeCODe (detection of communities for optimization decomposition), was developed in these works and is undergoing continual improvement.*

1.2 Optimality

The pursuit of optimality has become the most distinctive feature of the field of PSE (e.g., [135]). The optimal control theory dates back to Bellman's pioneering work on dynamic programming [28] and Pontryagin's minimum principle in the 1950s [324]. In view of the practical difficulty of analytically solving the optimal control for nonlinear systems, approximation approches have been proposed, which can be classified into two types. In the first type, the partial differential equations (PDEs) governing the optimal control law (policy function) and control cost (value function), namely the Hamilton-Jacobi-Bellman (HJB) equations, are solved under approximate models [360], which typically remains difficult to solve especially for highly nonlinear and non-polynomial systems[3]. In the second type, the control inputs are implicitly determined by solving an approximated optimal control problem in a finite future horizon. The latter approach, known as MPC, has been widely recognized as a powerful process control strategy, especially for processes with operational constraints. The development in the optimization algorithms and solvers within the PSE community [37], as well as the proposal of its variant, economic MPC, as an integration of real-time optimization and process control [98], have largely promoted its popularity.

The instillation of MPC into process control problems to convert them into optimization problems, as a standard mindset, helps to reduce the conceptual complexity and cumbersomeness of control theory. However, it would be simplistic to believe that MPC might be omnipotent for nonlinear control. With MPC, the control problems are forced to bear the complexity of optimization techniques, while optimization per se can not replace the analysis of dynamic properties such as stability and robustness. The direct transplantation of optimal control is not free from criticisms (see, e.g., Foss' perspectives in 1973 [116]). After many years of development in MPC, it is still worth noting the following two issues.

- The solution of optimal control problems may not be easy. Despite the progress in nonlinear programming (NLP), the size of MPC problems may still grow considerably large in the recently popular stochastic and robust MPC problems [261]. Especially, the distributed MPC of nonlinear chemical processes necessarily results in nonconvex NLP problems, for which the properties and potential of distributed optimization algorithms are yet unexplored to a large extent.

- The MPC as an approximated optimal control strategy does not necessarily guarantee stability, unless the objective function and constraints are well tuned such that an accordingly derived Lyapunov function satisfies feasibility assumptions [256]. However, such assumptions are difficult to test prior to the execution of MPC. Although a Lyapunov stability condition may be explicitly imposed on MPC as a constraint, it is difficult to find its optimal design [234]. Moreover, the suboptimality of MPC with respect to the truly optimal control is difficult to assess.

Hence the problem of interest is:

[3]Although in a model-based setting such a PDE solution is difficult, one can design data-driven control strategies, known as approximate dynamic programming [330] or reinforcement learning [401], by simultaneous adjusting the approximated policy and value functions with updated observations.

Question 1.2. *How should difficult distributed MPC problems be solved, and how should Lyapunov stability analysis and Lyapunov function design be conducted, for nonlinear systems?*

Part II of the thesis includes the results that answer Question 1.2. For the solution algorithm of distributed nonlinear MPC, in Chapter 7, a new algorithm called ELLADA is developed. It is based on a modification of the classical ADMM algorithm [44] that involves a double-layer structure to guarantee convergence to a KKT point of the monolithic problem under mild assumptions, an acceleration scheme to reduce the number of iterations, and an approximate NLP solution technique to save the computational time of each subproblem. The conceptual derivation of the algorithm is presented, and detailed proofs of the relevant propositions are given in the appendices. Numerical studies are performed to test the convergence property and computational performance of the ELLADA algorithm on a benchmark quadruple tank process. The application to a large-scale plantwide process [233] is undergoing research and is to be reported in the near future. For the Lyapunov stability analysis and Lyapunov function design, the following three works are included.

- In Chapter 8, given any Lyapunov function, bilevel programming formulations are derived for estimating the rate of decay of the Lyapunov function and estimating the domain of attraction of the Lyapunov function. The typical way of treating bilevel programming problems, namely converting into mixed-integer nonlinear programs (MINLP) and then calling global solvers, is presented.

- In Chapter 9, given the required rate of decay and domain of attraction, stochastic programming formulations are derived for optimizing the Lyapunov function in a parameterized form. ADMM algorithm as a distributed optimization algorithm and stochastic gradient descent algorithm are used to solve the stochastic programming problems.

- Based on the Lyapunov analysis and design, in Chapter 10, the concept of Lyapunov dynamic flexibility of nonlinear chemical processes is introduced, and formulations for Lyapunov flexibility analysis and optimal flexible process design are given. An iterative correction factor-based algorithm is proposed for practical solution of these problems.

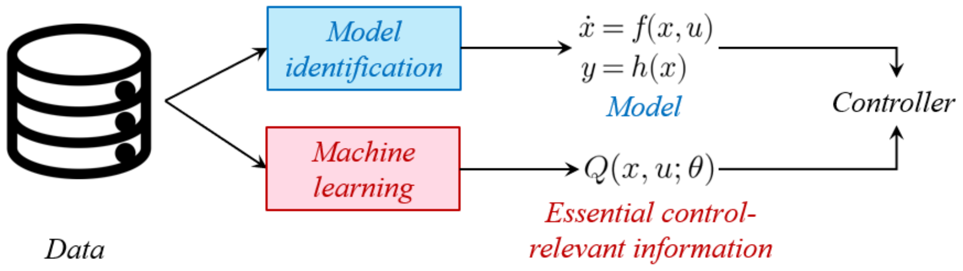


Figure 1.2: Model-based and model-free control driven by data.

1.3 Intelligence

While the development of process control algorithms typically relies on a dynamic model of the system, the models of chemical processes are rarely available in certain, fully accurate and constant forms. Models need to be identified or partially identified from the observations, or *data*, from the plant during either offline test or online production mode. Methods for model identification have been well studied [238], although a generic and efficient method for nonlinear systems, especially when simultaneously combined with the task of state observation, and when structural error exist so that the convergence to accurate values can not be established, is lacking. The ideas of adaptive control and dual control were proposed in the 1960s to allow the control and parameter estimation to proceed simultaneously or even synergistically [111], although their practice in process control is very limited so far. The incorporation of machine learning techniques in model identification has been considered as a resort to handle complex systems more recently [367].

Data can also drive the process control in another way – model-free control. An illustration of the distinction between model-based and model-free control is given in Fig. 1.2. In model-free control, instead of identifying a model that fully describes the dynamic behavior of the object, one exploits learning techniques to find only some essentially control-relevant information from the data. This dates back to the empirical PID tuning rule of Ziegler and Nichols in 1942, where PID parameters are tuned by the frequency and time constants obtained from a step test [490]. For linear systems, iterative feedback tuning (IFT) [158] and virtual reference feedback tuning (VRFT) approaches were proposed to obtain PID controllers that shape the closed-loop response [51]. More generally for nonlinear systems, adaptive (approximate) dynamic programming, often also named reinforcement learning in artificial intelligence contexts, can be used to train the controller based on the optimality principle, which has become more popular in the recent years motivated by the development in deep neural networks [376].

Despite the above-mentioned developments, it is believed that the exploitation of data in process control will continue to develop in theory and receive wider applications in practice, especially with the rapid growth and common acceptance of machine learning. It is noteworthy that the following two issues, although often neglected, are always important.

- While it is always possible, in principle, to use more complex models or model-free descriptions of data, such as deep neural networks, to achieve higher accuracy, this does not necessarily imply that such accurate but complex descriptions are easy to handle (or "invert") when using them for control purposes, e.g., as a part of the nonlinear MPC problem to be solved in real time. Although success may be made in small-scale and relatively simple processes, it is likely that the extension of complex descriptions will suffer from the curse of dimensionality, for which the identification and the utilization of such will result in a significant waste of computational resources.

- While the pursuit of a model or data description that matches the data points or trajectories themselves is desirable, it does not necessarily imply that such a pristine objective always directly leads to good performance in control. For control purposes, it should often be worth accounting for some *control-relevant information* explicitly. Such control-relevant information provides more precise and generalizable estimates or constraints about the manifold on which the system evolves under its nonlinear dynamics.

Therefore, the information to be inferred from the data needs to be essentially relevant to control. On one hand, this allows a circumvention of the need of identifying a complete dynamic model or redundant dimensions in the descriptions of the dynamics. On the other hand, the explicit and direct consideration of control-relevant information helps to improve the performance of data-driven control due to better generalizability. Hence the question of interest here is stated as follows.

Question 1.3. *What is the essential control-relevant information and how to efficiently learn such information from data?*

The author's efforts on data-driven control are included in Part III of the thesis.

- In Chapter 11, the model identification problem is considered. It is proposed that for nonlinear control strategies, the identification procedure should not only consider the direct values of the model functions, but also in a Lie-Sobolev manner by explicitly accounting for *Lie derivatives* as the essential control-relevant information. Case studies are carried out on a non-ideal reactor and a reaction system with complex reaction mechanisms.

- In Chapter 12, model-free control using adaptive dynamic programming is considered, where the *optimal control law and cost functions* as the essential information are regressed from trajectory samples. Distributed optimization algorithms are introduced to solve the problem, and the modifications for the circumstance under input constraints are discussed. This approach applies to relatively simple and state-observable systems.

- In Chapter 13, an input–output data-driven model-free control strategy is proposed in view of the fact that the states may not be observable for chemical processes. The method, called dissipativity learning control (DLC), is based on learning the *dissipativity property* as the essential information from the trajectory samples of the inputs and outputs. DLC is applied to the regulating control of a polymerization reactor and the tracking control of a gas-phase reactor in periodical modes.

- In Chapter 14, the DLC framework is further investigated. Under more standardized procedures, the effect of statistical errors in sampling and inference on the resulting controller synthesis is examined, which provides theoretical foundations for this input–output data-driven control strategy. The advantage of DLC in comparison with a basic linear system identification-based optimal control strategy is demonstrated with a two-phase reactor example.

1.4 Summary

This Ph.D. thesis collects most of my research works aiming at shrinking the gap between the currently existing methods and the outlook of more structured, optimal, and intelligent solutions to the control of chemical systems. The contents of most of the following chapters are based on the author's written journal and conference papers (Chapter 2 – [409], Chapter 3 – [405], Chapter 4 – [408], Chapter 5 – [417], Chapter 6 – [404], Chapter 8 – [410], Chapter 10 – [415], Chapter 12 – [407], Chapter 13 – [411]) with necessary minor revisions. Chapters 7, 9, 11, 14 are not yet published, in which Chapter 7 is shared on arXiv [416]. The introduction and conclusion chapters use materials from the author's contributions in the review and perspective papers [77, 412, 76].

Part I
Structuredness: Exploiting Network Topology to Design Control Architectures

I am to build beneath the water my bride room,	築室兮水中，
and thatch it with an aquatic lotus roof.	葺之兮荷蓋。
With a purple murex dresser and calamus walls,	蓀壁兮紫壇，
I spread scented pepper plants to decorate the halls.	播芳椒兮成堂。
Beams of orchid, pillars of cassia,	桂棟兮蘭橑，
lintels of magnolia, alcoves of angelica.	辛夷楣兮葯房。
Curtains woven from creeping figs knotted,	罔薜荔兮爲帷，
and screens of cymbidiums that have blossomed.	擗蕙櫋兮既張。

Qu Yuan, "*Nine Songs: Goddess of River Xiang*" (excerpt), 3rd century B.C.
屈原《九歌·湘夫人》

Chapter 2
The Role of Community Structures in Sparse Feedback Control

2.1 Introduction

Large-scale and complex networks are ubiquitous in electric, traffic, biomolecular, economic and social systems [58]. Such networks also arise in industrial systems such as chemical and energy plants from the use of mass and energy integration [18]. Due to the prohibitive cost of establishing and maintaining a fully centralized control for large-scale systems, structured control schemes with a trade-off between control performance and sparsity [183], including decentralized control and distributed control, have become the status quo of research activities as well as industrial applications [11, 52]. Extensive research in the late 1970s and early 1980s proposed system decomposition methods for decentralized control according to the special hierarchical structure of strongly connected components in the structural graph [49, 434, 321]. However, the effect of network topology on the control performance, and the exploitation of network topology to assist the control design, remain largely unexplored problems.

In the recently emerging discipline of network science, studies have focused on the choice of minimum driving nodes to guarantee structural controllability and the relation between structural controllability and network topological features [236, 237, 309, 399]. Structural controllability analysis has been applied to the pinning synchronization of networks [390, 471]. Although these studies have offered abundant insights into the control of complex networks, the analysis is restricted to structural concepts, and important practical issues including the design of decentralized or distributed control algorithms, and the corresponding quantitative control performance, have not been addressed. With this background, in this work we aim to explore the relation between *network topology* and *control performance*.

Modular networks refer to networks composed of node subsets, called communities, such that the interconnections between nodes in different communities are much sparser than the interconnections between nodes in the same communities [114, 294]. In fact,

©2019 IEEE. Reprinted, with permission, from W. Tang, P. Daoutidis. The role of community structures in sparse feedback control. *2018 Annual American Control Conference (ACC)*, (June 2018). `DOI:10.23919/ACC.2018.8431002`.

modularity in the interaction patterns of system variables has always been the underlying and implicit principle guiding the design of decentralized and block-decentralized control strategies in process systems [258, 192, 74]. The fact that modular networks are ubiquitous in the biological world also suggests that network modularity may have a strong correlation with control performance. While it has been evidently shown that modularity does not contribute to the structural controllability of networks [325], the effect of modularity on network control has not been quantified.

In this work, we investigate the control performance of modular networks under sparsity-promoting control [228] and structured optimal control [226]. For a generic network control problem, we show that as the cost of feedback channels increases, the sparsity of feedback control is gradually promoted while the communities in the network are preserved. A comparison in control performance and total control cost of networks with different modularities is also performed, and shows a reduction of control cost in modular networks when the cost of feedback channels becomes significant. Subsequently, we focus on decentralized control architectures, and for modular networks, we reveal the correlation between the control performance and the similarity of the decentralized control architecture to the communities. Through these studies, we claim that the community structures play an important role as the "core" of feedback control.

In the last part of the chapter we posit that the findings of our study, specifically that modular networks are advantageous over non-modular ones in the total control cost under significant feedback channel cost, provide a justification for the emergence of modularity in the evolution of biological networks. We also point out that our conclusions offer a justification for decomposing a complex network for decentralized or distributed control in the framework of community detection, which can be followed for the design of control architectures for complex process systems.

2.2 Preliminaries

2.2.1 Modularity

Many real-world networks are modular networks that can be divided into subsets of nodes, called communities, such that the interconnections between nodes in the same communities are much denser than the interconnections between nodes in different communities. The difference between inter-community and intra-community interconnections is quantified by *modularity*, first defined by Newman and Girvan in [296]. For an undirected unipartite network containing nodes $1, 2, \ldots, n$, the modularity Q is defined as

$$Q = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{a_{ij}}{m} - \frac{k_i k_j}{m^2} \right) \delta(C_i, C_j). \tag{2.1}$$

In the above expression, a_{ij} is the (i, j)-th element of the adjacency matrix, i.e. $a_{ij} = 1$ if there is an edge between node i and j, and $a_{ij} = 0$ otherwise. k_i and k_j are the degrees of nodes i and j, respectively, and m is the total number of edges

$$k_i = \sum_{j=1}^{n} a_{ij}, \quad m = \frac{1}{2} \sum_{i=1}^{n} k_i = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}. \tag{2.2}$$

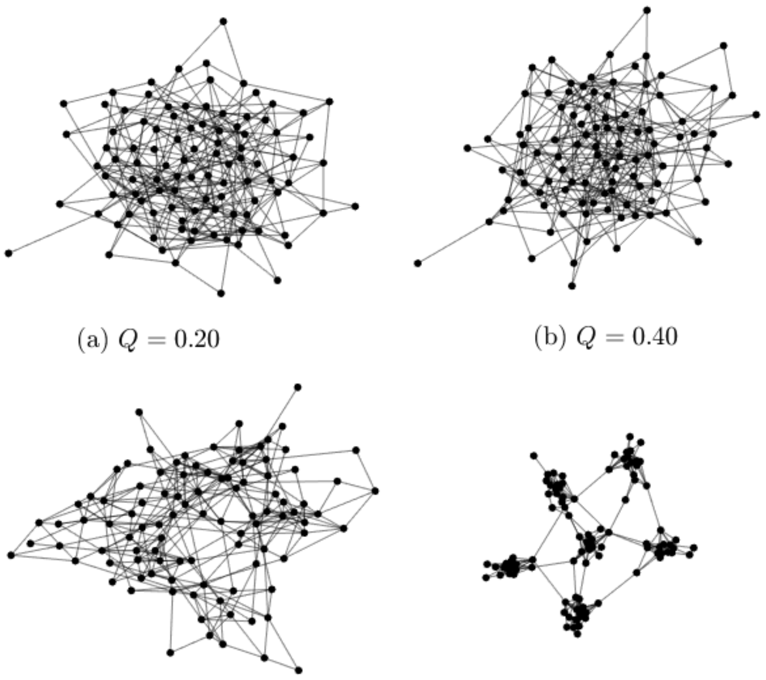
C_i and C_j are the indices of the communities to which i and j belong, respectively. $\delta(C_i, C_j) = 1$ if $C_i = C_j$, i.e. nodes i and j are in the same community, and $\delta(C_i, C_j) = 0$ otherwise. a_{ij}/m is the existing edge fraction between i and j, and $k_i k_j / m^2$ is the expected edge fraction connecting i and j when the edges are randomly redistributed while preserving the degree of each node. The modularity Q is the intra-community sum of these differences.

The community affiliation of each node $C_1, C_2, \ldots, C_n$ can in principle be determined by maximizing Q over all partitions of nodes. The computational intractability of this global modularity maximization problem necessitates efficient approximate algorithms. Here we use the Louvain algorithm (also known as the fast unfolding algorithm) [39] which is known to be the most efficient one for modularity maximization. The details of this algorithm are omitted for brevity. [1]

2.2.2 Network generation

For the purpose of this study, we generate generic networks with any desired modularity by simulated annealing [196]. The process is initiated by generating an Erdős-Rényi random graph where a given number of edges are randomly distributed. In this work, we fix the node number as 100 and the edge number as 300, i.e., the average degree of a node is $\bar{k} = 6$. At each iteration we randomly choose two edges and rewire them while keeping the degree of the incident nodes unchanged. The change in the distance to the

[1] A Matlab implementation for Louvain algorithm is available at `http://perso.uclouvain.be/vincent.blondel/research/louvain.html`.


(a) $Q = 0.20$ (b) $Q = 0.40$

(c) $Q = 0.60$ (d) $Q = 0.78$

Figure 2.1: Networks of different modularities.

desired index value is then calculated:

$$\Delta = |Q_{\text{rewired}} - Q_{\text{desired}}| - |Q_{\text{previous}} - Q_{\text{desired}}|. \tag{2.3}$$

The probability for substituting the previous network with the rewired new network is $\exp(-\beta\Delta)$ if $\Delta > 0$ and 1 if $\Delta \leq 0$. The annealing temperature parameter β is gradually increased during the iterations. We adjust the parameters, including the initial annealing temperature, temperature increment at each iteration, and number of iterations, to produce a series of networks with a range of modularity as wide as possible. Fig. 2.1 shows some networks generated with different modularities following this procedure.

2.2.3 Sparsity-promoting control

The Laplacian dynamics is the most common form of dynamics associated with networks and has been extensively studied in the context of synchronization or consensus formation [307, 418]. In this system, each node corresponds to a state x_i, whose value is affected by the discrepancies with the states at adjacent nodes, a local manipulated input u_i and a local exogenous disturbance d_i:

$$\dot{x}_i(t) = \sum_{j=1}^{n} a_{ij}(x_i(t) - x_j(t)) + u_i(t) + d_i(t), \tag{2.4}$$

or in a compact form

$$\dot{x} = -Lx + u + d \tag{2.5}$$

in which L is the Laplacian matrix of the network satisfying $L_{ij} = k_i$ if $i = j$, $L_{ij} = -1$ if $a_{ij} = 1$ and, $L_{ij} = 0$ otherwise.

For the feedback control law $u = -Fx$ where F is the feedback gain matrix, the closed-loop system becomes

$$\dot{x} = -(L + F)x + d. \tag{2.6}$$

The corresponding control performance is characterized by the $\mathcal{H}_2$ norm of the performance output $z = [R_1^{1/2}x; R_2^{1/2}u]$:

$$J(F) = \text{trace}(P) \tag{2.7}$$

where $P \succeq 0$ is dependent on F through the following algebraic Lyapunov equation, in which we assume R_1 and R_2 are equal to identity for simplicity

$$-(L + F)P - P(L + F)^{\top} + (R_1 + F^{\top} R_2 F) = 0 \tag{2.8}$$

A small $J(F)$ indicates a small impact of the disturbance on the states and manipulated inputs, and hence good disturbance rejection performance. Therefore $J(F)$ is considered as a cost function of the feedback gain to be minimized.

However, minimization of $J(F)$ in general gives a centralized control with almost all elements of F nonzero, i.e. feedback channels between almost all input-state pairs. This is impractical for large-scale networks since the cost of maintaining feedback channels should be taken into consideration. Here we use the formulation of [228] to promote the sparsity of F so that a tradeoff can be reached. The cost of the feedback channels is $\gamma\text{card}(F)$, in which the coefficient γ represents the cost of a single feedback channel, and $\text{card}(F)$ is the number of nonzero gains in F. The optimization problem is then transformed into the following form

$$\min \;\; J(F) + \gamma \mathrm{card}(G) \quad \text{s.t.} \;\; F - G = 0 \tag{2.9}$$

and solved using an alternative direction method of multipliers (ADMM) involving iterative F- and G-optimization.[2]

2.3 Results

2.3.1 Sparsity-promoting control of modular networks

Fig. 2.2 shows the change of the feedback gains as the cost parameter γ increases for a highly modular network with modularity $Q = 0.78$ and average degree $\bar{k} = 6$. The adjacency matrix in Fig. 2.2(a) clearly exhibits a modular structure as the nonzero adjacency elements are more densely concentrated in 6 diagonal square blocks.

When γ is small, the feedback channels do not contribute a significant cost. For example, when $\gamma = 10^{-6}$ (10^{-5}), employing all 10^4 channels only costs 0.01 (0.1), far less than the $\mathcal{H}_2$ norm value $J = 14.9$. Outside of the 6 diagonal blocks where the nonzero adjacency matrix elements are concentrated, most of the gain elements are nonzero (see Figs. 2.2(b) and 2.2(c)), which indicates that the feedback control is shaped in a centralized pattern with dense inter-community feedback gains. On the other hand, the feedback gains within the communities, and especially those on the main diagonal line, i.e. the self-feedback gains, have larger values (as visualized with deeper green colors), which suggests that intra-community feedback plays a more important role than inter-community feedback.

As γ increases from 10^{-6} to 10^{-2}, the number of feedback channels used to control the system gradually decreases to 1/5 of the total number (see Figs. 2.2(b)–(f)). Under the influence of the increasing channel cost, the feedback channels between communities are sacrificed while those inside the communities are preserved. The more interactions exist between communities, the slower the inter-community feedback gains are discarded, for example see the feedback between the 1st and the 6th community. With the increase of γ, the control performance is maintained at almost the same level as that of the centralized control (J increases from 14.90 to 14.93).

[2] A Matlab software for solving sparsity-promoting control problem with ADMM is available at `http://people.ece.umn.edu/users/mihailo/software/lqrsp/`.

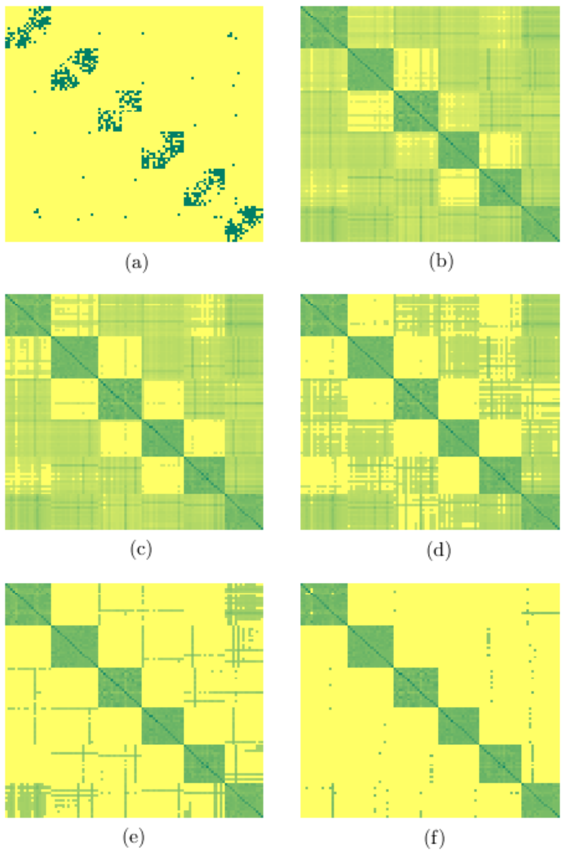


Figure 2.2: Visualization of (a) the adjacency matrix and feedback gains at (b) $\gamma = 10^{-6}$, (c) $\gamma = 10^{-5}$, (d) $\gamma = 10^{-4}$, (e) $\gamma = 10^{-3}$, and (f) $\gamma = 10^{-2}$. A yellow pixel represents a zero element. The pixel color approaches green when the gain becomes larger.

These results demonstrate that the communities in modular networks play the role of "cores" that concentrate dense feedback channels with large feedback gains, which are important for maintaining control performance and are not to be compromised under increasing feedback channel cost.

2.3.2 Sparsity-promoting control of networks with different modularities

We now investigate how the difference in modularity in networks influences the control performance. We first point out that network modularity does not lead to better

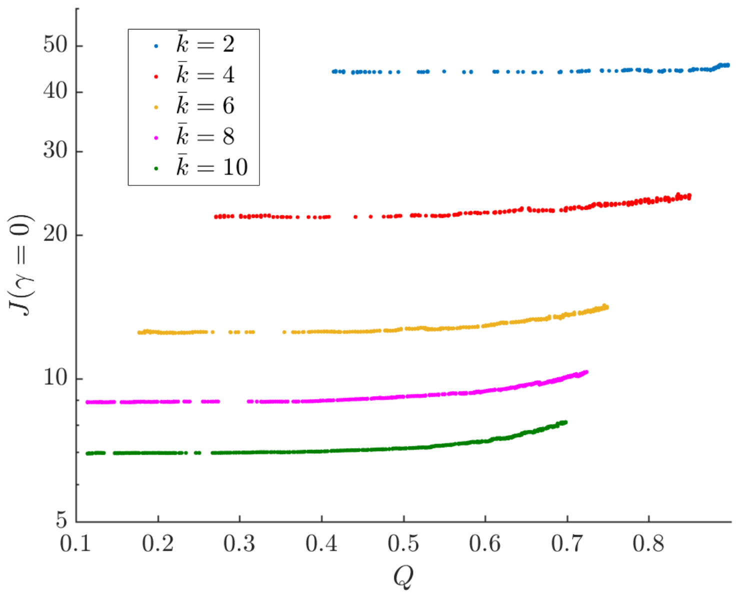


Figure 2.3: Performance of centralized control ($\gamma = 0$).

performance if the control architecture is centralized (see Fig. 2.3). In fact, as the modularity increases, the $\mathcal{H}_2$ norm of the associated Laplacian dynamics slightly increases. In other words, modular networks are not advantageous in the control cost over non-modular ones, if the cost of feedback channels is not considered. Instead, as shown in Fig. 2.3, the control performance is mainly determined by the average degree (or the total edge number) of the network, which affects the self-stabilizability of the Laplacian dynamics.

We proceed to compare networks with different modularities under sparsity-promoting control. Networks with average degree $\bar{k} = 6$ are chosen for the comparison. Fig. 2.4 shows the changes of control performance, number of feedback channels, and the total control cost as γ increases for 6 networks with different modularities. For γ between 10^{-4} and 10^{-2}, the control performance is almost constant for each network, with only very small increase (see Fig. 2.4(a)), while the feedback gains become sparser (see Fig. 2.4(b)). This result indicates that for any of network studied, regardless of its modularity, only a portion of the total feedback channels is necessary for maintaining good control performance. Therefore, the sparsity-promoting control framework can be applied to networks, whether modular or non-modular, to obtain a feedback control that is much sparser than the centralized control but has good performance.

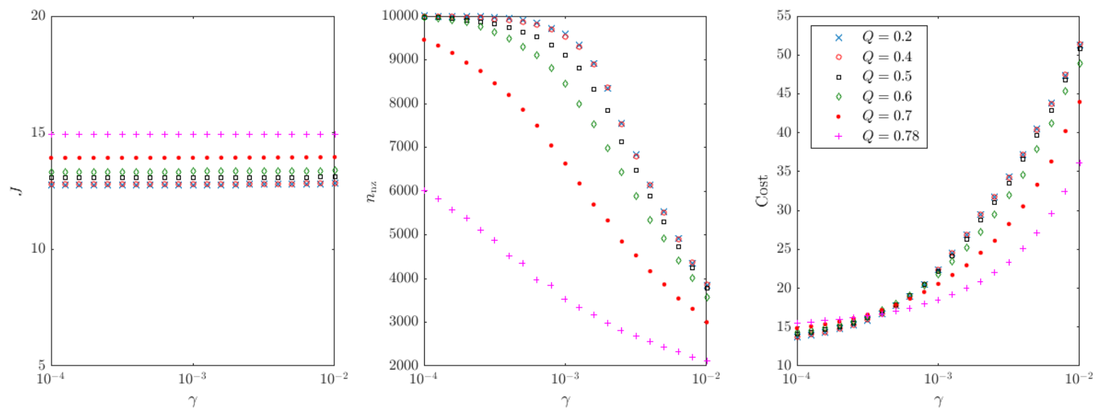


Figure 2.4: The change of (a) the control performance, (b) number of nonzero feedback gains, and (c) total control cost with the feedback channel cost γ for networks with different modularities.

However, for networks with different modularities, the promotion of sparsity proceeds at different rates. At any same cost parameter γ, a much smaller portion of feedback channels is used for networks with larger modularities than for networks with smaller modularities. The reduction of feedback channels helps modular networks to reach a lower total control cost as the cost of feedback channels becomes significant (see Fig. 2.4(c)). For example when $\gamma = 10^{-3}$, the network with $Q = 0.78$ uses about 3500 feedback channels while the one with $Q = 0.20$ uses about 9500. The reduction in total control cost becomes larger as γ increases.

When γ exceeds the order-of-magnitude of 10^{-2}, the cost of the feedback channels exceeds the cost related to the control performance, and dominates the total control cost. A portion of intra-community channels has to be sacrificed for sparsity, which results in control performances significantly worse than those under centralized control. These plots are omitted here due to space constraints. These results suggest that community-based feedback control is the control of maximum "allowed" sparsity, beyond which the sparsity can not be further promoted with good performance.

2.3.3 Structured optimal control under different decentralized architectures

In the sparsity-promoting control studied above, the feedback channels are concentrated yet not completely restricted in the communities. We now consider what we

Table 2.1: Performance deterioration and number of feedback channels for community-decentralized control

Q	0.20	0.30	0.40	0.50	0.60	0.70	0.78
$p(\%)$	1.43	1.73	2.00	1.76	1.49	0.93	0.28
n_{nz}	2640	2160	1670	1710	1682	1688	1674

term *community-decentralized control*, i.e. block-decentralized control such that nonzero feedback gains are constrained to exist only between nodes in the same community. The motivation for this is the fact that decentralized control is the most common practice for large-scale systems. The performance that the community-decentralized control can achieve is found by solving the following optimization problem:

$$\min \ \text{trace}(P) \quad \text{s.t. } (2.8), \ \ P \succeq 0, \ \ F \circ \mathcal{S} = 0 \tag{2.10}$$

in which $\circ$ represents element-by-element product, and $\mathcal{S}$ is the binary matrix corresponding to the prescribed structure of F. $\mathcal{S}_{ij} = 1$ if F_{ij} is restricted to be 0, and $\mathcal{S}_{ij} = 0$ otherwise. The optimization problem (2.10) is solved by an augmented Lagrangian algorithm proposed in [226].

Table 2.1 shows the performance of community-decentralized control, as characterized by the performance deterioration ratio $p = (J_{\text{community}} - J_{\text{centralized}})/J_{\text{centralized}} \times 100\%$, and the number of nonzero feedback gains n_{nz}. It can be seen that for networks with different modularities, using community-decentralized control generally maintains good control performance (within 2% of the centralized control performance), although at lower modularities, the deterioration ratio is slightly larger and more feedback channels are used. In other words, as long as the feedback control architecture is designed by using the communities, by optimally tuning the feedback gains, the control performance approaches that of centralized control.

2.4 Conclusion and Discussion

Through the study of sparsity-promoting control of modular networks, we found that communities in the network act as the "core" of feedback control in the following sense: (i) With an increase in feedback channel cost, intra-community feedback channels are preserved in priority. (ii) The control performance of community-decentralized control is close to that of a centralized architecture. (iii) Compared to non-modular networks, modular networks have a faster sparsity promotion rate, which results in a reduction in total control cost. Of course, these results are obtained for a specific network dynamics, and the measures that quantify the control performance and the control cost are also quite simplified. Yet the results document a meaningful, fundamental connection between network topology and control performance.

2.4.1 The evolution of modularity in biological networks

Modular networks including metabolic, protein interaction, and transcription regulation networks, are common in the biological world [114]. Many different hypotheses for the evolutionary origins of modularity have been proposed without a consensus so far [190, 438, 206, 64, 259]. To our best knowledge, the reason for the emergence of modularity in the biological networks during evolution has never been explicitly addressed within a control perspective.

It is seen from our results that modular networks are not advantageous over non-modular networks under centralized control. However, when the feedback control channels contribute a significant part of the cost, modular networks lead to largely reduced cost and become advantageous in control (see Fig. 2.4). This is achieved by discarding most of the inter-community channels and retaining the intra-community feedback channels. Even when the control is forced to be decentralized (compartmentalized), the communities in the network offer a natural architecture of community-decentralized control, whose performance is almost as good as that of a centralized architecture.

It is reasonable to hypothesize that the unit cost of the feedback channels, γ, is correlated to the magnitude and the frequency of the control signals. In biological networks, the control signals are generated by the underlying molecular mechanisms (enzyme synthesis, protein interactions, activity regulation), which consume materials and energy. The more intense is the exogenous disturbance to the network, the more expensive is the cost to maintain the feedback channels. Our conclusion that modular networks are advantageous over non-modular ones at significantly large γ but disadvantageous at low γ is in accordance with the systems biologists' observation that the modularity is selected in frequently varying environments but anti-selected in constant environments (see [190, 64] for example).

To exclude the possibility that the reduction of control cost is caused by some unobserved change of another topological feature, we conducted a simulated annealing experiment which reconfigures the network to minimize the total control cost given

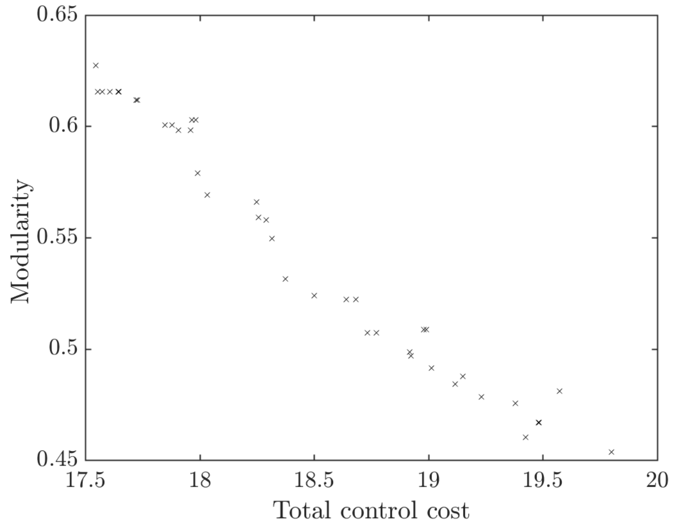


Figure 2.5: The correlation between total control cost and modularity.

$\gamma = 10^{-2}$. To reduce the computational burden, we adjusted the number of nodes to be 40 and the average degree as $\bar{k} = 3$. Fig. 2.5 shows that the decrease in the total control cost is accompanied by an increase in network modularity, albeit with some oscillations. This corroborates the negative correlation between control cost and network modularity under the requirement of sparsity. Since the simulated annealing involves a recursion of rewiring and replacing the old network by the new one with a probability, similar to the procedure of mutation and selection in the biological evolution, the simulated annealing in concept simulates the evolutionary emergence of modularity in networks.

2.4.2 Design of architectures for process control

Model predictive control (MPC) has been one of the major topics in recent studies in process control, and has been widely applied in industrial practice [274, 337, 454, 254]. Due to the high computational cost of a centralized MPC architecture, distributed MPC has recently attracted a lot of attention [362, 63]. It is generally agreed that the decomposition has an impact on the computational performance of distributed MPC and an optimal decomposition should be found by analyzing the interactions in process systems. However, for the majority of works in this field, the decomposition is chosen intuitively.

The systematic decomposition for distributed MPC has only been recently investigated. Community detection methods have been proposed for input-output bipartite graphs where the edges are weighted by interaction measures appropriately defined [406, 408, 417] and system digraphs [180] to obtain decompositions of process networks. Case studies with a specific process have shown a significant reduction in the computational cost resulting by these decompositions while retaining a satisfactory control performance [327, 328].

The findings in this chapter serve as a conceptual justification for decomposing the network for control in the framework of community detection. The communities in the network correspond to the minimal subsystems which can be controlled separately while maintaining control performance, and the decomposition according to the communities achieves a good trade-off between control performance and computational cost.

Chapter 3
Optimal Network Topology under Sparse Feedback Control of Laplacian Networks

3.1 Introduction

Large scale connected systems with complex functionalities, typically referred to as *networks* [295], are ubiquitous in biology, society and engineered systems. The emergence of network science provides a wealth of methods of analyzing topological features of given networks, e.g., distribution and correlation of node degrees [444, 22, 291], modularity (the existence of node communities that are well-connected within but less connected across communities) [296], and core-periphery structures (the existence of a small portion of nodes well connected to all nodes) [351]. Based on the structural analysis of networks, the effects of topological features on the network functionalities, such as resilience from attacks [364] and isolation of epidemics [441], have been addressed.

Control is an important aspect of networks. The feedback control mechanisms widely existing in biomolecular pathways [172] and industrial plants [18] are crucial to their normal functionality or operability. In the recent years, many works have focused on specific forms of dynamics on networks, such as consensus dynamics [347, 471, 9], in which the goal is to achieve the synchronization among network nodes (e.g., unmanned vehicles). Studies of the effect of network topology on control have mainly focused on choosing the minimum number of driver nodes to guarantee structural controllability [236, 399]. From this point of view, however, real biological networks do not appear to have evolved towards more controllable structures [325]. More recent research has considered some control energy-related metrics [460, 315] and evaluated the role of topological features in minimizing such metrics. However, these controllability or control energy metrics correspond to open-loop rather than feedback controllers. Moreover, due to the limitations of communication between the sensors and actuators on networks, real-world feedback controllers need to be *sparse*, as in the context of decentralized and distributed control in the engineering fields [73, 16].

In our recent works, we have aimed at understanding the effect of network topology on control taking into account the sparsity of feedback controllers. In [409], we adopted

©2019 Elsevier. Reprinted, with permission, from W. Tang, P. H. Constantino, P. Daoutidis. Optimal sparse network topology under sparse control in Laplacian networks. *IFAC-PapersOnLine* (Volume 52, Issue 20, December 2019). `DOI:10.1016/j.ifacol.2019.12.193`.

the framework of [228] to reach a trade-off between control performance and controller sparsity, and revealed the role of network modularity in promoting the controller sparsity while preserving good control performance in Laplacian networks. In [69], a larger ensemble of Laplacian networks with diverse topological features were examined, and it was shown that modular networks and highly hierarchical networks become advantageous over other networks under moderate and high controller sparsity, respectively, in the sense of a total control cost including the control performance and the controller sparsity. In [68], such a total control cost was used as a fitness function and a genetic algorithm was used to drive the network evolution; the emergence of modularity was once again observed in this evolutionary process.

In this work, we extend our previous works to incorporate the network topology (represented by the Laplacian matrix) as a variable to be optimized, and *directly solve for the optimal network topology and optimal controller* with a trade-off between control performance, network sparsity and controller sparsity. The formulation is similar to that of [228] and that of [83] for finding a sparse representation of uncontrolled Laplacian dynamics. Two trade-off parameters, standing for the cost of network edges and feedback channels, are used to account for the sparsity of the network and controller. As these parameters increase, the sparsity of the network and the controller is promoted, resulting in optimal network topologies with modular and hierarchical topological features. These features are similar to ones that have been observed in biological networks. Motivated by this, we discuss how the topology optimization procedure mimics a control cost-driven evolution process. Also, for engineering networks (e.g., vehicle platoons or chemical processes), our results suggest some principles of network topology design and retrofit and controller synthesis.

3.2 Method

We consider undirected weighted networks. Denote by A the adjacency matrix, whose (i, j)-th entry $a_{ij} > 0$ represents the weight of the edge between node i and node j in the network. Laplacian dynamics is a common form of dynamics associated with networks, where each node corresponds to a state x_i, whose time rate of change is affected by the differences with its neighboring nodes, a local control input u_i and a exogenous disturbance d_i:

$$\dot{x}_i(t) = -\sum_{j=1}^{n} a_{ij}(x_i(t) - x_j(t)) + u_i(t) + d_i(t). \tag{3.1}$$

For any node i, let k_i be its degree, namely the sum of the weights of all edges incident to it, i.e., $k_i = \sum_j a_{ij}$. Let L be the Laplacian matrix of the network. Then its elements satisfy $l_{ij} = k_i$ if $i = j$ and $l_{ij} = -a_{ij}$ if $i \neq j$. Then (3.1) can be written as

$$\dot{x} = -Lx + u + d \tag{3.2}$$

where u and d are the vectors of control inputs and disturbances, respectively. Given a feedback gain matrix F such that $u = -Fx$, with the (i, j)-th element f_{ij} representing the feedback gain from x_j to u_i, the control performance considered by [228] is the $\mathcal{H}_2$-norm of the system with d as inputs and $z = [R_1^{1/2}x; R_2^{1/2}u]$ as outputs:

$$J(L, F) = \text{trace}(P). \tag{3.3}$$

Here P is a positive definite matrix dependent on F through the following algebraic Lyapunov equation

$$-(L + F)P - P(L + F)^\top + (R_1 + F^\top R_2 F) = 0. \tag{3.4}$$

We assume that the weighting matrices R_1 and R_2 for the states and control inputs in z are equal to the unit matrix for simplicity. A small value of $J(L, F)$ indicates good performance of disturbance attenuation. Therefore, $J(L, F)$ is considered as a cost function of the feedback gain to be minimized.

The minimization of $J(L, F)$ alone usually does not give a Laplacian matrix of a sparse network and the feedback gain matrix of a sparse controller. To account for network and controller sparsity, we add two regularization terms $\gamma_L\text{card}(L)$ and $\gamma_F\text{card}(F)$ to the $\mathcal{H}_2$ norm for minimization, where card is the cardinality function counting the number of nonzero entries in matrices, and γ_F and γ_L are the parameters standing for the cost of each feedback channel and network edge, respectively. As parameters γ_F and γ_L increase, the relative importance of controller and network sparsity over control performance in the objective function increases, thus promoting the adoption of sparser controllers and network topologies. The resulting optimization problem is expressed as

$$\begin{aligned} &\min\ J(L,F)+\gamma_L\mathrm{card}(L)+\gamma_F\mathrm{card}(F) \\ &\text{s.t. } l_{ij}=l_{ji}\le 0,\ \forall i,j;\quad l_{ii}=k_i,\ \sum_j l_{ij}=0,\ \forall i. \end{aligned} \tag{3.5}$$

The constraints on the elements of L require that L be a Laplacian matrix of some network with the degree of each node fixed (otherwise the degrees can be infinitely large to minimize the $J(L,F)$). The degrees of the nodes can be assigned by a graph generation model, such as Erdős-Rényi and Barabási-Albert [295], which also generates an initial guess of the network topology.

[228] solved the above problem for the case of fixed L (given network) to obtain a so-called sparsity-promoting controller using an alternative direction method of multipliers (ADMM) algorithm. Here we solve (3.5) following a similar procedure. The variables F and L are first duplicated with equality constraints ($L=\bar{L}$, $F=\bar{F}$), and an augmented Lagrangian is constructed:

$$\begin{aligned} \mathcal{L}(L,F,\bar{L},\bar{F}) =&\bar{J}(L,F)+\gamma_L\mathrm{card}(\bar{L})+\gamma_F\mathrm{card}(\bar{F}) \\ &+\langle\Lambda_L,L-\bar{L}\rangle+\frac{\rho_L}{2}\|L-\bar{L}\|^2+\langle\Lambda_F,F-\bar{F}\rangle+\frac{\rho_F}{2}\|F-\bar{F}\|^2, \end{aligned} \tag{3.6}$$

where $\bar{J}(L,F)=J(L,F)$ if L satisfies the constraints in (3.5) and $\bar{J}(L,F)=\infty$ otherwise, Λ_F and Λ_L are dual matrices, ρ_F and ρ_L are penalty parameters, the inner product $\langle\cdot,\cdot\rangle$ between symmetric matrices is the trace of their product, and $\|\cdot\|$ stands for the Frobenius norm. The arguments are updated in a block coordinate descent manner followed by dual updates iteratively until convergence, i.e., in each iteration, the following steps are executed:

$$\begin{aligned} &(L,F):=\arg\min_{L,F}\mathcal{L}(L,F,\bar{L},\bar{F}) \\ &\bar{L}:=\arg\min_{\bar{L}}\mathcal{L}(L,F,\bar{L},\bar{F}),\quad \bar{F}:=\arg\min_{\bar{F}}\mathcal{L}(L,F,\bar{L},\bar{F}) \\ &\Lambda_L:=\Lambda_L+\rho_L(L-\bar{L}),\quad \Lambda_F:=\Lambda_F+\rho_F(F-\bar{F}). \end{aligned} \tag{3.7}$$

The updates of the duplicated variables $\bar{L}$ and $\bar{F}$ can be solved analytically. One can show that

$$\bar{l}_{ij} = \begin{cases} 0, & l_{ij} + \lambda_{L,ij}/\rho_L \geq -\sqrt{2\gamma_L/\rho_L} \\ l_{ij} + \lambda_{L,ij}/\rho_L, & \text{otherwise} \end{cases} \tag{3.8}$$

where $\bar{l}_{ij}$ and $\lambda_{L,ij}$ is the (i,j)-th entry of L and Λ_L, respectively. A similar formula holds for the $\bar{F}$ entries with the subscripts L substituted by F. The minimization of (L,F) does not possess a closed-form expression and L is subject to the constraints in (3.5). To simplify the (L,F)-minimization step, we approximate the nonlinear objective function $\mathcal{L}$ with its linearization at the previous value of (L,F), denoted by (L_0,F_0). Specifically, let $\tilde{L} = \bar{L} - (1/\rho_L)\Lambda_L$ and $\tilde{F} = \bar{F} - (1/\rho_F)\Lambda_F$. Then we have

$$\begin{aligned} \mathcal{L} =& \bar{J}(L,F) + \frac{\rho_L}{2}\|L-\tilde{L}\|^2 + \frac{\rho_F}{2}\|F-\tilde{F}\|^2 + \text{const.} \\ \approx& \langle \frac{\partial J}{\partial L}\Big|_{(L_0,F_0)}, L - L_0\rangle + \langle \frac{\partial J}{\partial F}\Big|_{(L_0,F_0)}, F - F_0\rangle + \frac{\rho_L}{2}\|L-\tilde{L}\|^2 + \frac{\rho_F}{2}\|F-\tilde{F}\|^2 + \text{const.} \end{aligned} \tag{3.9}$$

The partial derivatives of J are calculated to be

$$\frac{\partial J}{\partial L}\Big|_{(L_0,F_0)} = -2P_0G_0, \;\; \frac{\partial J}{\partial F}\Big|_{(L_0,F_0)} = 2(F_0 - P_0)G_0 \tag{3.10}$$

with the symmetric matrix G_0 being the solution of a Lyapunov equation

$$(L_0 + F_0)G_0 + G_0(L_0+F_0)^\top = I \tag{3.11}$$

and P_0 determined from (3.4) with L and F replaced by L_0 and F_0, respectively. This leads to the approximate optimizers of F in the form of

$$F := \tilde{F} - \frac{2}{\rho_F}(F_0 - P_0)G_0 \tag{3.12}$$

and of L by solving a quadratic programming problem

$$\begin{aligned} L := \arg\min \;& \frac{\rho_L}{2}\|L-\tilde{L}\|^2 - 2\langle P_0G_0, L-\tilde{L}\rangle \\ \text{s.t. } & l_{ij} = l_{ji}(\forall i,j), \;\; l_{ii} = k_i, \;\; \sum_j l_{ij} = 0 \;\; (\forall i). \end{aligned} \tag{3.13}$$

It is not hard to prove that the solution to (3.13) is

$$L = Q - \mathrm{Diag}(\mathrm{diag}(Q)) + \mathrm{Diag}(k) + [\mathrm{Diag}(\mu) - \frac{1}{2}\mu e^\top - \frac{1}{2}e\mu^\top] \tag{3.14}$$

where $Q = \tilde{L} + (1/\rho_L)(P_0G_0 + G_0P_0)$ and

$$\mu = \frac{1}{n-2}[2v - \frac{e^\top v}{n-1}e], \;\; v = Qe - \mathrm{diag}(Q) + k. \tag{3.15}$$

In the above expressions we denote by diag(Q) the vector of diagonal entries of Q, Diag($\cdot$) for any vector the diagonal matrix whose nonzero entries are the components of this vector, e the vector whose components are all equal to 1, and $k = [k_1, k_2, \dots]^\top$ the column vector of node degrees.

3.3 Results

We first generate an unweighted Erdős-Rényi random network of $N = 100$ nodes with a connection probability $p = 0.06$, which is shown in Fig. 3.1 and has an average degree of $\bar{k} = 5.96$. It does not exhibit any significant topological characteristics. If the cost of feedback channels $\gamma_F = 0$, under the given topology, the minimization of the $\mathcal{H}_2$ norm $J(L, F)$ leads to a feedback gain matrix F where all the elements are nonzero, i.e., the best-performing controller is non-sparse. If the cost of network edges γ_L is also 0, starting from the current random topology, the network will be optimized towards a fully-connected structure (i.e., a complete graph) in which the weight of each edge takes a small positive value. This is due to the fact that when the mutual influences between the nodes are small and widely distributed, the dissemination of exogenous disturbances along the edges becomes weak and can be well suppressed by small feedback control signals. However, such non-sparse controller and non-sparse network topology are undesirable in that they require a heavy load of feedback information transfer between all the states and all the inputs, and a dense pattern of physical interactions between all the nodes on the network. Whenever it is reasonable to consider the feedback channels and network edges as associated with an extra cost, the optimal network topology and feedback controller need to be solved with a trade-off between the control performance and sparsity in the form of (3.5).

Here we solve problem (3.5) according to the linearized ADMM algorithm described in the previous section, with the node degrees determined by a random network of

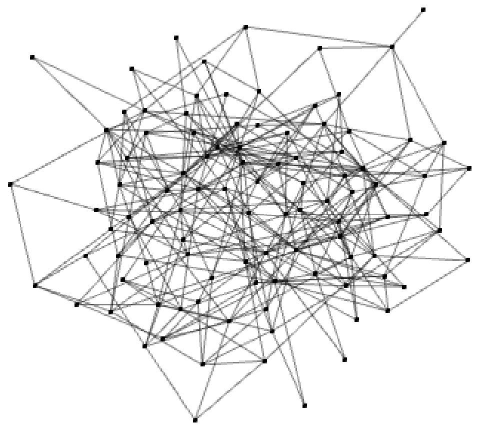


Figure 3.1: A random Erdős-Rényi network.

$N = 100$ and $p = 0.06$. The best achievable control performance for such a random network is $J \approx 10$. Hence we choose the range of the cost of feedback channel γ_F and the cost of network edge γ_L within the range of $[10^{-4}, 10^{-2}]$, such that under low values of γ parameters, the densest matrices of F and L ($\text{card}(F) = \text{card}(L) = 10^4$) result in costs of feedback channels and network edges in the order of $O(1)$ (much lower than J), and under high values of the γ parameters, such dense matrices F and L result in costs around $O(10^2)$ (much higher than J), thus promoting sparse L and F matrices. In the ADMM algorithm, the two penalty parameters are empirically chosen as $\rho_L = \max(3^{[\log_{10}(\gamma_L)]+3}, 10^{-3}/\gamma_L)$ ($[\cdot]$ is the roundoff integer) and $\rho_F = 10$ to improve the convergence of the algorithm.

First we consider the case where γ_L is small, so that the network topology remains an almost complete graph with small edge weights with almost all node pairs, and γ_F is varied. As γ_F increases from 10^{-4} to 10^{-2}, the optimized feedback gain matrices with increasing sparsity are visualized in Fig. 3.2. It is observed that the controller sparsity is promoted by discarding the feedback channels between nodes of low degrees while keeping the channels from high-degree nodes to the rest of the nodes. The high-degree nodes, by their interaction with more neighboring nodes, better dissipate the effect of disturbances. Hence the information of high-degree states is more valuable as a source of computing the control signals.

Next, consider the case where γ_F is small, so that a dense controller can always be employed, and γ_L is varied. The optimized network topology under different values of γ_L is illustrated in Fig. 3.3. With increasing promotion of the network sparsity, an

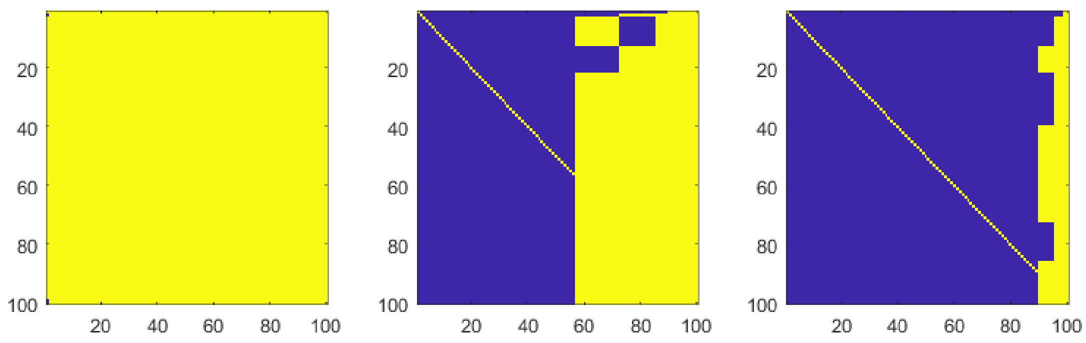


Figure 3.2: Optimal feedback gains under $\gamma_F = 10^{-4}$, 10^{-3} and 10^{-2} (from left to right) and $\gamma_L = 10^{-4}$. A blue pixel stands for 0 and a yellow pixel corresponds to a nonzero feedback gain. The number of the nodes are sorted in the ascending order of their degrees.

increasing number of nodes escapes from the dense component and remains connected to it with fewer edges. When γ_L reaches a high value, a *modular* feature emerges on the optimal network topology. As observed from the last subplot of Fig. 3.3, the majority of the nodes belong to 3 groups (called communities). The connections inside the communities are dense, but the communities are mutually disconnected, except that all of them are linked to a small number of high-degree nodes. The community detection procedure of [296] confirms the existence of 3 communities (apart from the 6 nodes adjacent only to the high-degree nodes), as marked by the node colors in Fig. 3.3, and a modularity level of 0.3557. The nodes that interconnect the otherwise disconnected components of the network are said to have high *betweenness centrality*, in the sense that they act as "hubs" that must be passed through on the shortest path between nodes of different communities. They can also be considered as the shared parts of overlapping communities, or "kernels" that form a *hierarchy* with the ordinary community nodes.

The adjacency matrix of the modular network and its optimized feedback gain matrix are shown in Fig. 3.4. By arranging the node indices according to their community affiliations, the adjacency matrix exhibits a block diagonal pattern. Comparing the feedback gain matrix to the adjacency matrix, we observe that despite under a sufficiently low feedback channel cost γ_F, the sparsity of controller is automatically promoted to a certain extent (with $\text{card}(F)/N^2 = 0.3435$). The nonzero controller gains exist mainly between the nodes within the same communities, and between the first two communities with strong interactions through the high-degree nodes.

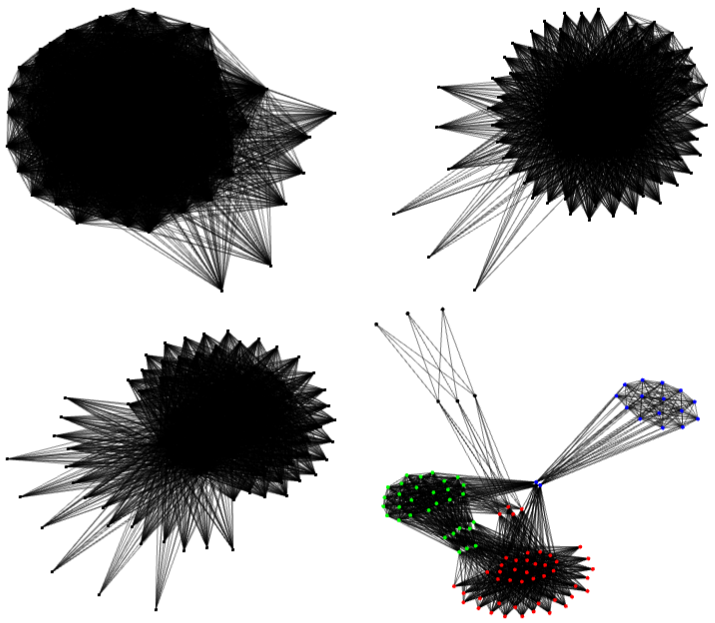


Figure 3.3: Optimal network topology under $\gamma_L = 10^{-4}$ (upper left), 10^{-3} (upper right), $10^{-2.5}$ (lower left) and 10^{-2} (lower right) and $\gamma_F = 10^{-4}$. The community structures in the last network are marked by different colors of nodes.

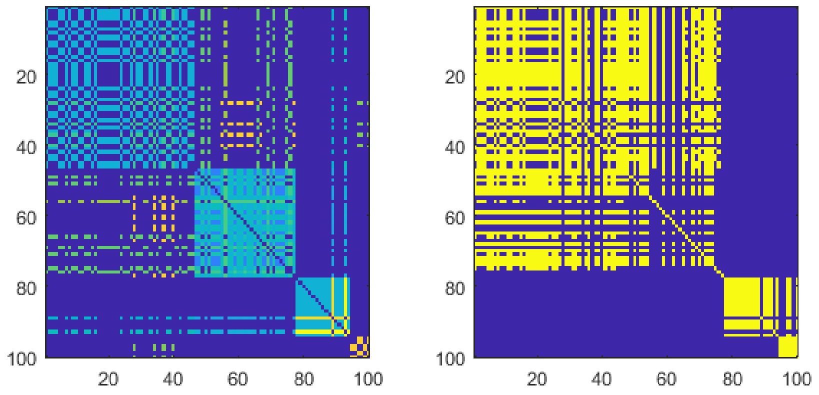


Figure 3.4: Adjacency matrix of the optimized network topology where the pixel color ranging from blue to yellow stands for edge weight in $[0, 1]$ (left), and feedback gain matrix (right) where yellow pixels stands for nonzero gains and blue ones stands for zeros, obtained under $\gamma_L = 10^{-2}$ and $\gamma_F = 10^{-4}$.

To summarize, γ_F increases the controller sparsity by preferentially preserving the feedback channels from high-degree nodes, and γ_L promotes the network sparsity by encouraging the emergence of modularity and hierarchy. We finally investigate the case with the values of both γ_L and γ_F to be high enough to make both the network topology and feedback controller sparse. For $\gamma_L = 10^{-2}$ and γ_F increased from 10^{-4} to 10^{-2}, the obtained feedback matrices are visualized in Fig. 3.5. With the increase of γ_F, the controller sparsity is further improved both inside the communities and across communities. As a result, most feedback channels become concentrated inside the communities. The inter-community channels are gradually removed with the exception of those linked to the high-degree nodes bridging the communities. Such an architecture is typically referred to as a *distributed* and *hierarchical* one. Specifically, in distributed control [73], the system is considered as comprising of several distributed subsystems and the control mechanisms exist only locally inside the systems; in hierarchical control [362], the controller allows the subsystems to communicate with a coordinator on a higher level. The distributed/hierarchical control architecture is in accordance with the network topology, as the community structures and high-degree, high centrality nodes play the roles of distributed subsystems and coordinators, respectively.

3.4 Discussions

Through the optimization of network topology together with its feedback controller, we have highlighted the importance of topological features including modularity and hierarchy. These features are common in real-world networks, such as in biological ones capturing protein interactions and gene regulation [342, 184, 114]. In fact, many works by systems and computational biologists have attempted to understand the evolutionary origins of topological features of biological networks (see, e.g., [190, 64, 260]), especially modularity and hierarchy. However, the importance of network topology on the cost of control in terms of control performance and sparsity has not been explicitly considered. On the other hand, the control mechanisms in biological networks are distributed and hierarchical (see, e.g., [89]), while the relation between the control architecture and the topology of the network under control has attracted little attention.

The optimization procedure as described in Section 3.2 can be interpreted within an *evolutionary perspective* as a deterministic emulation of the evolution of network topology driven by the total cost as specified by the objective function in (3.5). The optimization is performed with iterative updates, during which the intermediate solution at each iteration possesses similarities (inherited traits) from the previous iteration as

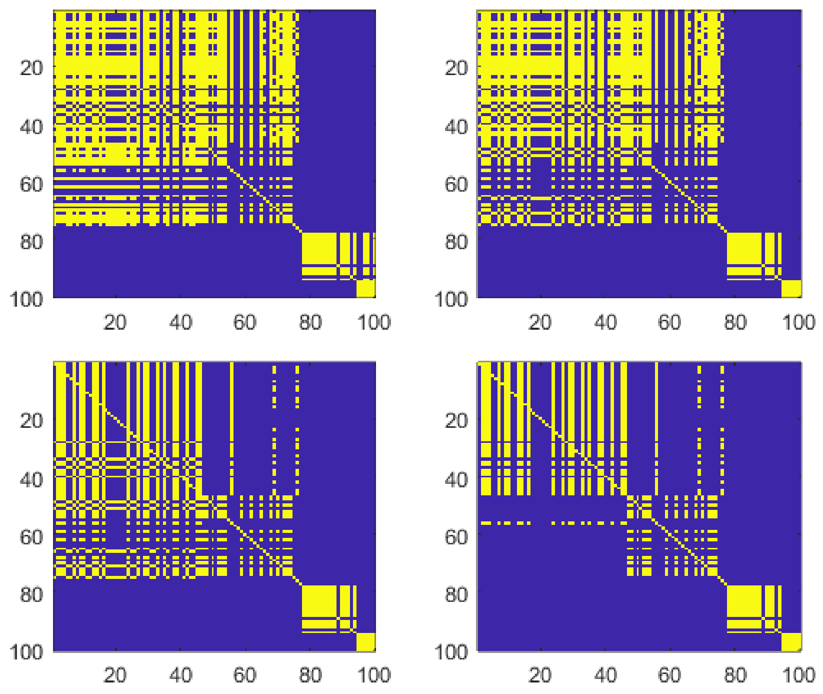


Figure 3.5: The optimal feedback gain matrices under $\gamma_F = 10^{-4}$ (upper left), 10^{-3} (upper right), $10^{-2.5}$ (lower left) and 10^{-2} (lower right) under $\gamma_L = 10^{-2}$. Blue and yellow pixels stands for zero and nonzero entries, respectively.

well as differences (directed mutations) to be accumulated throughout the iterations (generations). The triggering mechanism of such directed mutations lies in (3.9), where the $\partial J/\partial L$ and $\partial J/\partial F$ drives the network to seek a new topology and a new controller that potentially give a better control performance, while the $\|L - \tilde{L}\|^2$ and $\|F - \tilde{F}\|^2$ terms adjust the update direction to account for sparsity requirements. The penalty parameters ρ_F and ρ_L act as resistance forces that determine the rate of evolution. The dual matrices Λ_F and Λ_L serve as a posteriori fitness indicators assessing whether the performance-sparsity trade-off has been well achieved, whose values will be used in the next iterations and influence the subsequent evolution.

Therefore, this study, together with our previous works [409, 69, 68], proposes a new hypothesis that the topological features widely existing in biological networks, including modularity and hierarchy, together with their control (regulatory) mechanisms, *are the consequences of an evolution process driven by control performance* (i.e., disturbance rejection or stability) *and sparsity* (i.e., complexity of the system and its control mechanisms). During the evolution, the degrees of the nodes are considered as fixed at the values of the initial network, thus preserving the degree distribution in the generative model of the network.

The simultaneous optimization of network topology and controller also provides principles on the design and retrofit of networked engineering systems, such as unmanned vehicle networks or integrated and intensified chemical processes, that are restricted by sparsity of the network and controller. Specifically, with the distributed and hierarchical control architectures widely applied to industrial systems [362, 63], we suggest that network systems should be designed with modular features on the majority of low-degree nodes and a hierarchical feature between the high-degree nodes and low-degree nodes. On the other hand, given such a networked system, distributed/hierarchical control architectures should be designed based on the analysis of the underlying network topology, which has been addressed by the authors in the context of controlling chemical plants [77].

Remark 3.1. *We note that in the majority of relevant studies where a desirable network topology needs to be constructed, simulated annealing has been adopted as a standard metaheuristic approach to the optimization of networks due to their combinatorial nature. In simulated annealing, the edges are repeatedly rewired in a trial-and-error manner and accepted with a probability based on the effect of such rewiring on the objective function. The computational inefficiency of such a procedure largely prohibited the application to networks of moderately large sizes. In this study, we avoid this issue by considering the network as weighted and represented by a Laplacian matrix of continuous-valued entries, and regularizing the number of network edges by a cost γ_L in optimization.*

3.5 Conclusion

In this work, we proposed an optimization formulation for directly finding the optimal network topology with a trade-off between control performance, feedback controller sparsity and network sparsity, which is solved via a linearized ADMM algorithm. By optimizing such sparse networks under sparse controllers, networks with modular and hierarchical topological features have been obtained, with a distributed and hierarchical control architecture. These results are in accordance with the observations in real-world biological networks, and allow the interpretation that these topological features have resulted from a control performance and sparsity-driven evolution process. In the future work, it is worth investigating how such conclusions can be extended to more general linear, nonlinear, or multivariable node dynamics, and systems where the feedback channels are restricted by a communication graph structure [308, 227].

Chapter 4
Network Decomposition for Distributed Control by Community Detection in Input–Output Bipartite Graphs

4.1 Introduction

Large-scale systems with complex interconnections, including biomolecular reactions [398], electric grids [103], cyber-physical systems [194], and chemical processes [18], typically exhibit a networked structure. In recent years, a large volume of literature has emerged on the control and control-relevant analysis of network systems [15, 262, 143]. Distributed control plays a central role in the control of such systems [11]. It is based on a decomposition of a large-scale network into constituent subsystems, each controlled by a local controller with some degree of communication between controllers. Distributed control has been pursued in the context of typical network control problems such as consensus and formation control [307, 418, 306], for which there are well-identified agents to be coordinated. For more general linear network systems under linear control, significant effort has been put on the design of well-posed and stabilizing distributed control laws [73, 210], robust stability analysis [6, 488] and optimal control [277, 228].

A related direction of research of particular importance to process systems is the design of distributed model predictive control (MPC) schemes (see e.g. [362, 63, 288]). Such controllers retain the inherent advantages of MPC (e.g. flexibility of using different control objective functions, potential of optimizing process economics, and direct handling of constraints) while addressing its key limitation when applied to large-scale nonlinear systems, namely the prohibitive cost of the repeated on-line solution of the underlying dynamic optimization problem. In distributed MPC, the plant is decomposed into constituent subsystems, and the optimal control problem is solved in a distributed manner, with some information sharing between the controllers [232, 109, 422]. In tightly interconnected plants with an underlying networked structure the decomposition of the system into the distributed architecture is both important and difficult to determine due to the strong and often hidden interactions among the process variables. It is well recognized that the optimal decomposition for nonlinear distributed MPC, i.e., the optimal allocation of the process variables into the distributed control subsystems in order to reduce the computational effort without compromising control performance, is an open and challenging problem [362, 63].

Large-scale system decomposition has a long history, especially in the context of decentralized control. Early efforts seek to detect the underlying acyclic pattern and permute the system into a hierarchy of subsystems, so that the subsystems become controllable in sequence [393, 380]. This decomposition is achieved through graph-theoretic search of strongly connected states [265, 434] or acyclic input-output reachable systems [321], with sufficiently small elements in the coefficient matrix neglected [372]. Later studies [478, 381] considered the situation when the system has a small number of complicating components between otherwise separable subsystems and thus exhibit bordered block diagonal interaction pattern. However, the special graph structures required for these rearrangements are usually restrictive for process systems, due to the complex interactions between the system states and the existence of recycle structures.

On a different vein, several papers considered the design of decentralized control architecture as an optimal graph partitioning problem. In [146], the number of interconnecting edges is to be minimized, with size restrictions in the corresponding subsystems. In [303, 304, 101], algorithms to partition networks into subnetworks of approximately equal sizes with minimized interconnections were developed. The work in [25] sought to partition the graph with a trade-off among subsystem sizes, interconnections, pair-wise distances and information relevance. The work in [186] proposed using a system digraph representation with sensitivity weights with a merging scheme based on LQG regulator metrics. In [456], a computationally expensive genetic algorithm was applied to optimize the effect of decompositions on the control performance for linear systems. Several other papers have considered application-specific decompositions [53, 197, 284].

From the perspective of network theory, network decomposition can be considered as a *community detection* problem. Community detection aims to divide the vertices of a graph into subsets (communities), such that the links inside communities significantly exceed the links between communities [114, 294]. The use of community detection in the context of control-relevant decomposition has only recently been pursued. A hierarchical agglomerative clustering approach has been proposed for the generation of input-output clusters with strong intra-cluster interactions (in an input-output connectivity sense) and weak inter-cluster coupling; agglomerative clustering [153] and divisive clustering approaches have been followed for this purpose [152] to produce candidate structures with different extents of decentralization. Graph theoretic formulations of these methods and a-posteriori assessment of optimality of the resulting clusters using modularity were proposed in [188], and extended to PDE-governed systems [267]. Recently, community detection in the system digraph based on modularity maximization was proposed to generate well-decoupled subsystems containing inputs, states and outputs [180]. This approach is based on pure connectivity considerations.

We note that pure variable connectivity such as the one used in [180] does not account for the intensity of input-output interactions. On the other hand, classical interaction measures based primarily on relative gain array [258, 192] do not account for the network topology. A combination of connectivity and response sensitivities (the coefficients of the linearized system), in the form of a relative sensitivity array (RSA), was recently proposed [468] and used in the context of input-output pairing, yet not in the context of network decomposition for distributed control.

In this work, we define a new interaction measure combining connectivity and response sensitivities, which we term *input-output affinity*, and use it as the edge weight in an input-output bipartite graph. We establish that input-output affinities capture the intensity of short-time interactions and hence are suitable measures for decomposing networks for the application of nonlinear model predictive control, where, due to the need for repeated solution of dynamic optimization problems the prediction horizon over which the optimization problem is solved is usually short. To account for the network topology in the decomposition we focus on the problem of input-output partitioning such that intra-subsystem input-output interactions are significantly stronger than inter-subsystem interactions. This is formulated as a community detection problem on the weighted input-output bipartite graph. The method of [23], which allows forming communities from both independent vertex sets (containing the inputs and outputs in our case) of a bipartite graph using modularity maximization, is adopted for the solution of this problem. A chemical process network example is used to illustrate the application of the proposed method.

4.2 Barber's Method for Community Detection

The problem of community detection in networks has been studied extensively in network science (see [115] for a recent review). Among the numerous algorithms of community detection, modularity-based methods stay as the mainstream, which consider the community detection problem as the maximization of a quality function, called *modularity*, defined for any partition of the nodes of the network. Modularity captures the difference between the number of intra-community edges in the network and its expected value in a randomized counterpart, thus characterizing the statistical significance of the existence of communities in the network [127]. The global optimization of modularity over all possible partitions is an NP-hard problem [46]. Approximate algorithms have been adopted instead to make the modularity maximization problem computationally tractable. Newman's spectral method for unipartite networks [293] performs the community detection by recursive bisections, in which each bisection is determined by the spectral decomposition of a modularity matrix and an additional adjustment (fine-tuning) step. Barber's method [23] is an extension of Newman's spectral method to bipartite graphs, aiming at detecting communities comprising of vertices from both independent vertex sets of such networks. This is the method that will be used in this chapter. The procedure is described below.

Consider a bipartite graph in which $U = \{u_i | i = 1, \ldots, p\}$ and $Y = \{y_j | j = 1, \ldots, l\}$ are two independent sets of vertices such that an edge exists between a vertex in U and another vertex in Y. For unweighted graphs, let a_{ij} be the (i, j)-th element of the bipartite adjacency matrix, such that $a_{ij} = 1$ if there is an edge between u_i and y_j, and $a_{ij} = 0$ otherwise. Let: $\beta(u_i)$ be the degree of u_i, namely the number of edges linking u_i and a vertex in Y; $\beta(y_j)$ be the degree of y_j, namely the number of edges linking a vertex in U and y_j; and β be the total number of edges, i.e.,

$$\beta(u_i) = \sum_{j=1}^{l} a_{ij}, \quad \beta(y_j) = \sum_{i=1}^{p} a_{ij}, \quad \beta = \sum_{i=1}^{p} \sum_{j=1}^{l} a_{ij}. \tag{4.1}$$

As there are β edges in total, of which $\beta(u_i)$ are incident with u_i, the likelihood for an edge to be incident with u_i is $\beta(u_i)/\beta$. Similarly, the likelihood for an edge to be incident with y_j is $\beta(y_j)/\beta$. The product $\beta(u_i)\beta(y_j)/\beta^2$ can then be viewed as the likelihood for an edge to exist or the expected fraction of edges between u_i and y_j. The difference between the fraction of edges between u_i and y_j and its expected value,

$$b_{ij} = \frac{a_{ij}}{\beta} - \frac{\beta(u_i)\beta(y_j)}{\beta^2} = \frac{a_{ij}}{\sum_{i'=1}^{p} \sum_{j'=1}^{l} a_{i'j'}} - \frac{\sum_{i'=1}^{p} a_{i'j} \sum_{j'=1}^{l} a_{ij'}}{\left(\sum_{i'=1}^{p} \sum_{j'=1}^{l} a_{i'j'}\right)^2} \tag{4.2}$$

is a modularity measure for a pair (u_i, y_j) that captures the extent that the relation between u_i and y_j is closer than randomly expected. For a weighted graph where a_{ij} is a positive real number, b_{ij} can be defined in the same way as above.

"Good" communities comprising of vertices from U and Y are those whose members u_i and y_j have a positive, large b_{ij}. For a partition $\mathcal{P}$ of $U \cup Y$ into disjoint communities $C_k, k = 1, 2, \ldots$, its modularity $Q(\mathcal{P})$ is defined as the sum of all intra-community b_{ij}:

$$Q(\mathcal{P}) = \sum_{C_k \in \mathcal{P}} \left(\sum_{y_j \in C_k} \sum_{u_i \in C_k} b_{ij} \right). \tag{4.3}$$

The optimal partition of a bipartite graph can be addressed through recursive bisections. The bisection of a subset of vertices $C \subseteq (U \cup Y)$ is chosen by maximizing the bisection modularity increase ΔQ. Suppose there are p_C vertices in $C \cap U$ and l_C vertices in $C \cap Y$. Define $\mathbf{B}_C \in \mathbb{R}^{p_C \times l_C}$ as the modularity matrix of C, formed by the b_{ij} for $u_i, y_j \in C$. Let $\mathbf{s} \in \mathbb{R}^{p_C}$ and $\mathbf{t} \in \mathbb{R}^{l_C}$ be the vectors indicating the decision of allocating vertices in C into two subsets C_1 and C_2

$$s_i = \begin{cases} 1, & u_i \in C_1 \\ -1, & u_i \in C_2 \end{cases}, \quad t_j = \begin{cases} 1, & y_j \in C_1 \\ -1, & y_j \in C_2 \end{cases}. \tag{4.4}$$

The modularity change after the bisection can be expressed as [293]

$$\Delta Q = \frac{1}{2}(\mathbf{s}^\top \mathbf{B}_C \mathbf{t} - \mathbf{1}^\top \mathbf{B}_C \mathbf{1}). \tag{4.5}$$

The second term $\mathbf{1}^\top \mathbf{B}_C \mathbf{1}$ is fixed, and can be viewed as a modularity loss when C is collapsed and its elements are not grouped. The first term $\mathbf{s}^\top \mathbf{B}_C \mathbf{t}$ can be viewed as a modularity gain when all members of C are reorganized into C_1 and C_2 according to $\mathbf{s}$ and $\mathbf{t}$. This term is to be maximized by a proper choice of $\mathbf{s}$ and $\mathbf{t}$. For the trivial choice $\mathbf{s} = \mathbf{t} = \pm\mathbf{1}$, $\Delta Q = 0$. When this trivial choice is optimal, C should not be divided.

The maximization of $G = \mathbf{s}^\top \mathbf{B}_C \mathbf{t}$ is naturally associated with the largest singular value of $\mathbf{B}_C$. Consider the singular value decomposition of $\mathbf{B}_C$

$$\mathbf{B}_C = \sum_{a=1}^{\min\{p_C, l_C\}} \sigma_a \mathbf{u}_a \mathbf{v}_a^\top, \tag{4.6}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$ are the singular values, and $\mathbf{u}_a \in \mathbb{R}^{p_c}$ and $\mathbf{v}_a \in \mathbb{R}^{l_c}$ are mutually orthogonal groups of unit vectors. Then

$$\mathbf{s}^\top \mathbf{B}_C \mathbf{t} = \sum_{a=1}^{\min\{p_C, l_C\}} \sigma_a (\mathbf{s} \cdot \mathbf{u}_a)(\mathbf{t} \cdot \mathbf{v}_a). \tag{4.7}$$

Note that $\mathbf{s}$ and $\mathbf{t}$ have fixed norms and restricted orientations since their dimensions are fixed and their components are ± 1. Suppose that $\mathbf{s}$ and $\mathbf{t}$ are allowed to point at any direction on the spheres prescribed by their norms. Then $\mathbf{s}^\top \mathbf{B}_C \mathbf{t}$ is maximized when $\mathbf{s}$ and $\mathbf{t}$ are parallel to $\mathbf{u}_1$ and $\mathbf{v}_1$, the vectors associated with the maximum singular value of $\mathbf{B}_C$, σ_1. Under the restriction on the orientations, choosing $\mathbf{s}$ that is "the most parallel" to $\mathbf{u}_1$ and the $\mathbf{t}$ that is "the most parallel" to $\mathbf{v}_1$, an approximately optimal solution can be obtained as

$$\mathbf{s} = \operatorname{sign}(\mathbf{u_1}), \quad \mathbf{t} = \operatorname{sign}(\mathbf{v_1}), \tag{4.8}$$

i.e., $s_i = 1$ if $u_{1,i} > 0$ and -1 otherwise, and $t_i = 1$ if $v_{1,j} > 0$ and -1 otherwise.

Since (4.8) is an approximation, a fine-tuning step is needed. After obtaining the bisection according to (4.8), we scan every element in C_1 (and C_2) to examine whether moving it into C_2 (C_1) (i.e., flip the sign of each component of $\mathbf{s}$ and $\mathbf{t}$) increases modularity. The move with the largest modularity increase is carried out. We continue scanning and moving elements until no further increase is available. In this way, a bisection with a maximal ΔQ is obtained. If the maximal ΔQ is positive, then C is divided according to the current $\mathbf{s}$ and $\mathbf{t}$ by (4.4). Recursive bisections are performed until further bisection does not yield any increase in modularity.

4.3 Community Detection in Input–Output Bipartite Graphs

4.3.1 Problem formulation

We consider nonlinear systems in the control-affine form

$$\dot{x} = f(x) + g(x)u, \quad y = h(x) \tag{4.9}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^p$, $y(t) \in \mathbb{R}^l$. $f(x) \in \mathbb{R}^n, g(x) \in \mathbb{R}^{n\times p}$ and $h(x) \in \mathbb{R}^l$ are smooth functions. x is the vector of state variables whose evolution is dependent on the manipulated inputs u, and y represents the outputs of the system that need to be controlled. Let $\bar{x}$ be the operational steady state at which the states should be controlled at. The equations are transformed such that $f(\bar{x}) = 0$, i.e., $\bar{x}$ is an equilibrium point under zero input. We denote by L_f the Lie derivative in the direction of vector field $f(x)$, and by r_{ij} the relative degree between u_i and y_j, i.e., the smallest positive integer such that $L_{g_i}L_f^{r_{ij}-1}h_j(x) \neq 0$ or $r_{ij} = \infty$ if such a positive integer does not exist.

Given a system (4.9), we aim to construct a bipartite graph of inputs and output, i.e., a bipartite graph with $U = \{u_i | i = 1, \ldots, p\}$ and $Y = \{y_j | j = 1, \ldots, l\}$. We assign a weight to each edge to account for the interaction of inputs and outputs, and specifically, to capture the short-time response of outputs over the inputs.

4.3.2 Weighted system digraph

For the system (4.9), a digraph can be derived (see e.g. [74]) as a directed graph with vertices representing the components of u, x, y and edges from u_i to x_k if $g_{ki} \not\equiv 0$, from x_k to x_l if $\partial f_l/\partial x_k \not\equiv 0$, and from x_l to y_j if $\partial h_j/\partial x_l \not\equiv 0$. It was proved in [74] that $r_{ij} = l_{ij} - 1$, where l_{ij} is the length of the shortest path from u_i to y_j in the unweighted system digraph. This establishes relative degree as a measure of dynamic interaction between u_i and y_j, which captures the sluggishness of the response of y_j to a change in u_i. Specifically, when u_i is a unit step input and other inputs are kept at 0, r_{ij} is the lowest order of nonzero time derivative of y_j at the initial time [74]:

$$\left.\frac{d^{r_{ij}} y_j}{dt^{r_{ij}}}\right|_{t=0} = L_{g_i} L_f^{r_{ij}-1} h_j(\bar{x}). \tag{4.10}$$

In what follows we define a weighted system digraph where a weight is assigned on each edge to capture the intensity of the initial response. We begin by considering the response sensitivities for each edge in the system digraph, defined as the corresponding coefficients in the linearized system around the operation point, $\bar{x}$,

$$S(u_i, x_k) = g_{ki}(\bar{x}),\ \ S(x_k, x_l) = \partial f_l(\bar{x})/\partial x_k,\ \ S(x_l, y_j) = \partial h_j(\bar{x})/\partial x_l. \tag{4.11}$$

Assumption 4.1. *We assume that if $g_{ki} \not\equiv 0$ ($\partial f_l/\partial x_k \not\equiv 0$, $\partial h_j/\partial x_l \not\equiv 0$), the corresponding sensitivity $S(u_i, x_k) \neq 0$ ($S(x_k, x_l) \neq 0$, $S(x_l, y_j) \neq 0$) at $\bar{x}$. In other words, there is no edge with sensitivity 0.*

We now define the edge weights in the equation graph as

$$\begin{aligned} w(u_i, x_k) &= 1 - \log |S(u_i, x_k)|, \;\; w(x_k, x_l) = 1 - \log |S(x_k, x_l)|, \\ w(x_l, y_j) &= - \log |S(x_l, y_j)|. \end{aligned} \tag{4.12}$$

The weights comprise of two terms. The first is due to the dynamic effect of the first variable on the second, assigned to be 1 when there is a differential equation involved (such as in input-state edges or state-state edges), and 0 when there is an algebraic relation between the variables (state-output edges). The second term accounts for the response sensitivity. The use of $-\log$ makes the weights negatively correlated to $|S|$ and closer to a notion of distance, and relates the multiplication of sensitivity coefficients with the summation of weights, thus allowing expressing the composite effect of an input on an output through input-output paths in an additive way.

Specifically, let P denote an input-output path and $e \in P$ the edges in the path. The length of the path $L(P)$ is defined as

$$L(P) = \sum_{e \in P} w(e). \tag{4.13}$$

Suppose that this path passes through $N(P)$ state variables $x_1, x_2, \ldots, x_{N(P)}$. Then, given the definitions (4.12),

$$L(P) = N(P) - \log \prod_{e \in P} |S(e)| = N(P) - \log \left| g_{1i} \frac{\partial f_2}{\partial x_1} \cdots \frac{\partial f_{N(P)}}{\partial x_{N(P)-1}} \frac{\partial h_j}{\partial x_{N(P)}} (\bar{x}) \right|. \tag{4.14}$$

This path length accounts for both the number of states in an input-output path and the corresponding sensitivities. We now proceed to establish the relation between the path length derived here and the short-time output response.

Theorem 4.1. *Let $\mu(P)$ denote the number of negative sensitivities on the path P. Let $x(t=0) = \bar{x}$. When u_i is subject to a small step input, $u_i(t) = \epsilon H(t)$, where $|\epsilon| \ll 1$, $H(\cdot)$ is the unit step function, and all other inputs are kept at 0, then for $t \in [0, \delta)$, where δ is sufficiently small,*

$$y_j(t) = h_j(\bar{x}) + \epsilon \sum_{P \in \mathcal{W}_{ij}} \frac{(10t)^{N(P)}}{N(P)!} (-1)^{\mu(P)} 10^{-L(P)} + O(\epsilon^2) \tag{4.15}$$

where $\mathcal{W}_{ij}$ is the collection of all paths from u_i to y_j.

Proof. Consider the Fliess expansion [173] which gives the response of y_j in a short time interval $t \in [0, \delta)$, assuming analyticity for the vector fields and scalar functions in (4.9):

$$y_j = h_j(\bar{x}) + \sum_{N=0}^{\infty} \sum_{i_0,\ldots,i_N=0}^{n} L_{g_{i_0}} \ldots L_{g_{i_N}} h_j(\bar{x}) \int_0^t d\xi_{i_N} \ldots d\xi_{i_0} \tag{4.16}$$

wherer $g_0 = f, \xi_0 = t, \xi_i = \int_0^t u_i(\tau)d\tau$, and $\int_0^t d\xi_{i_N} \ldots d\xi_{i_0} = \int_0^t d\xi_{i_N}(\tau) \int_0^\tau d\xi_{i_{N-1}} \ldots d\xi_{i_0}$. When $u_i = \epsilon H(t)$ and all other inputs are 0, only the terms involving f, g_i and u_i can be nonzero. When u_i appears multiple times in the iterated integrals, the corresponding terms will be $O(\epsilon^k), k \geq 2$. Lie derivatives involving only f are 0 as $f(\bar{x}) = 0$. So we consider only the terms involving g_i once and f. Since $f(\bar{x}) = 0$, only such terms for which f is not on the first position can be nonzero. Based on these arguments, the expansion in (4.16) takes the form

$$\begin{aligned} y_j &= h_j(\bar{x}) + \sum_{N=0}^{\infty} L_{g_i} L_f^N h_j(\bar{x}) \int_0^t d^N\xi_0 d\xi_i + O(\epsilon^2) \\ &= h_j(\bar{x}) + \sum_{N=0}^{\infty} L_{g_i} L_f^N h_j(\bar{x}) \frac{\epsilon t^{N+1}}{(N+1)!} + O(\epsilon^2) \\ &= h_j(\bar{x}) + \epsilon \sum_{N=1}^{\infty} \frac{t^N}{N!} L_{g_i} L_f^{N-1} h_j(0) + O(\epsilon^2). \end{aligned} \tag{4.17}$$

By definition,

$$\begin{aligned} L_{g_i} L_f^{N-1} h_j(\bar{x}) &= \underbrace{\frac{\partial}{\partial x}\left[\frac{\partial}{\partial x}\left[\cdots \frac{\partial}{\partial x}\left[\frac{\partial h_j}{\partial x} f\right] f \cdots\right] f\right]}_{N \text{ brackets in total}} g_i \bigg|_{\bar{x}} \\ &= \sum_{k_1,\ldots,k_N=1}^{n} \frac{\partial}{\partial x_{k_1}}\left[\frac{\partial}{\partial x_{k_2}} \cdots \left[\frac{\partial h_j}{\partial x_{k_N}} f_{k_N}\right] \cdots f_{k_2}\right] g_{k_1 i} \bigg|_{\bar{x}}. \end{aligned} \tag{4.18}$$

Since $f_k(\bar{x}) = 0$, $k = 1, \ldots, n$, a nonzero term is obtained only when all partial differential operators are imposed on the f components. Then

$$\begin{aligned} L_{g_i} L_f^{N-1} h_j(\bar{x}) &= \sum_{k_1,\ldots,k_N=1}^{n} \frac{\partial h_j}{\partial x_{k_N}}(\bar{x}) \frac{\partial f_{k_N}}{\partial x_{k_{N-1}}}(\bar{x}) \cdots \frac{\partial f_{k_2}}{\partial x_{k_1}}(\bar{x}) g_{k_1 i}(\bar{x}) \\ &= \sum_{k_1,\ldots,k_N=1}^{n} S(u_i, x_{k_1}) S(x_{k_1}, x_{k_2}) \ldots S(x_{k_N}, y_j). \end{aligned} \tag{4.19}$$

Every nonzero term under summation corresponds to a path from u_i to y_j with N state variables. Thus,

$$L_{g_i} L_f^{N-1} h_j(\bar{x}) = \sum_{P \in \mathcal{W}_{ij}, N(P)=N} \prod_{e \in P} S(e). \tag{4.20}$$

Then (4.17) becomes

$$\begin{aligned} y_j &= h_j(\bar{x}) + \epsilon \sum_{N=1}^{\infty} \sum_{P \in \mathcal{W}_{ij}, N(P)=N} \frac{t^N}{N!} \prod_{e \in P} S(e) + O(\epsilon^2) \\ &= h_j(\bar{x}) + \epsilon \sum_{P \in \mathcal{W}_{ij}} \frac{t^{N(P)}}{N(P)!} \prod_{e \in P} S(e) + O(\epsilon^2). \end{aligned} \tag{4.21}$$

From (4.14) we have

$$N(P) - L(P) = \log \prod_{e \in P} |S(e)| = \log \left[(-1)^{\mu(P)} \prod_{e \in P} S(e) \right] \tag{4.22}$$

and hence

$$\prod_{e \in P} S(e) = (-1)^{\mu(P)} 10^{N(P)-L(P)}. \tag{4.23}$$

Substituting (4.23) into (4.21), we obtain the result in (4.24). □

The short-time response of any output over any input in (4.15) is expressed as a summation of the contributions of each input-output path, which is dominated by the term corresponding to the shortest input-output path.

4.3.3 Shortest path length, distance and affinity

Let SP_{ij} denote the shortest path between u_i and y_j, and L_{ij} its length. If the shortest path in the weighted graph is identical to the shortest path in the unweighted graph, then $N(SP_{ij}) = r_{ij}$ and the expression in (4.14) becomes

$$L_{ij} = r_{ij} - \log \prod_{e \in SP_{ij}} |S(e)|. \tag{4.24}$$

When the contribution of the response sensitivities, $\prod_{e \in SP_{ij}} |S(e)|$ is small ($\ll 1$), L_{ij} exceeds r_{ij}; when the contribution of the response sensitivities is large ($\gg 1$), L_{ij} is smaller than r_{ij}; and when the product of the response sensitivities is close to 1,

$L_{ij} \approx r_{ij}$. Thus the shortest path length in the weighted system digraph is an extension of the relative degree, by modifying it according to the magnitude of the response sensitivities.

Noting that the scalings of inputs and outputs influence the path lengths L_{ij} by changing the sensitivity coefficients, we define a distance between u_i and y_j as

$$d_{ij} = L_{ij} - \min_{1 \le i' \le p, 1 \le j' \le l} L_{i'j'} + 1 \tag{4.25}$$

such that the smallest distance among all input-output pairs (u_i, y_j) is fixed at 1. A scaling of all inputs or all outputs by the same constant does not affect d_{ij}, since it causes the same additional term on the length of every path (4.14). Finally, to capture the desired property that strongly interacting input-output variables should be topologically close, we now define the *affinity* between u_i and y_j as

$$a_{ij} = 1/d_{ij} \tag{4.26}$$

with $a_{ij} \in [0, 1]$, resembling the elements of an adjacency matrix. This will now be used as a weight function in the input-output bipartite graph.

4.3.4 Modularity maximization

Motivated by the modularity measure b_{ij} derived in (4.2), we define an analogous modularity measure for the input-output bipartite graph

$$b_{ij} = \frac{a_{ij}}{\sum_{i'=1}^{p} \sum_{j'=1}^{l} a_{i'j'}} - \frac{\sum_{i'=1}^{p} a_{i'j} \sum_{j'=1}^{l} a_{ij'}}{\left(\sum_{i'=1}^{p} \sum_{j'=1}^{l} a_{i'j'}\right)^2}. \tag{4.27}$$

For a partition $\mathcal{P}$ of the input and output set $U \cup Y = \{u_1, \dots, u_p, y_1, \dots, y_l\}$, we can define its modularity as in (4.3). The recursive bisections and fine-tuning steps have already been described in Section 4.2. In summary, the decomposition is performed according to the following procedure:

1. Construct the weighted system digraph.
   (a) Take all inputs, states and outputs as vertices.
   (b) Evaluate sensitivities according to (4.11).
   (c) Draw and weight edges between vertices according to (4.12).

2. Construct the input-output bipartite graph.
    (a) Take all inputs and outputs as vertices.
    (b) Find the input-output shortest paths and their lengths L_{ij}.
    (c) Calculate distances using (4.25) and affinities using (4.26).
    (d) Calculate modularity measures with (4.27).
3. Perform community detection.
    (a) Initialize by assigning all vertices in a single community and label it as "decomposable".
    (b) For every "decomposable" community C:
        i. Obtain singular value decomposition of the modularity matrix (4.6).
        ii. Obtain approximated bisection with (4.4).
        iii. Fine-tune the bisection by recursively flipping the signs of each element of $\mathbf{s}, \mathbf{t}$, checking the modularity change, and perform the one with largest modularity increase.
        iv. Compute the modularity change (4.5). If $\Delta Q > 0$, execute the bisection, otherwise label the community as "indecomposable".
    (c) If there is now no "decomposable" community, the procedure is terminated. Otherwise return to 3(b).

The shortest path between any input and any output can be found using Dijkstra's algorithm with complexity of $O(E \log V)$ [447], and the complexity of the community detection procedure is $O(V^2 \log V)$ [293], where V and E are the number of vertices and nodes in the network, respectively. The overall procedure, dominated by the shortest path searches and community detection, is thus of polynomial complexity. We note that the network decomposition should be executed off-line and repeated only when there is a change in the system design.

The following input-reachability property can be stated for the resulting decomposition.

Theorem 4.2. *Let $\mathcal{P} = \{C_k | k = 1, \ldots, K\}$ be the obtained partition. If $\forall y_j \in Y, \exists u_i \in U$, s.t. $a_{ij} \neq 0$, then $\forall C_k \in \mathcal{P}$, $\forall y_j \in Y \cap C_k$, $\exists u_i \in U \cap C_k$, s.t. $a_{ij} \neq 0$.*

Proof. First, we assert that the sum of intra-community modularities is guaranteed to be non-negative by fine-tuning:

$$\sum_{u_i \in C_k} b_{ij} \geq 0, \quad \forall C_k \in \mathcal{P} \text{ and } \forall y_j \in C_k. \tag{4.28}$$

Assume that the first violation of (4.28) takes place at the bisection of $C_1 \cup C_2$ into C_1 and C_2 after fine-tuning. Let $\exists y_j \in C_1$ satisfy

$$\sum_{u_i \in C_1} b_{ij} < 0. \tag{4.29}$$

Because this is the first violation of (4.28), $C_1 \cup C_2$ still satisfies (4.28), i.e.,

$$\sum_{u_i \in C_1 \cup C_2} b_{ij} = \sum_{u_i \in C_1} b_{ij} + \sum_{u_i \in C_2} b_{ij} \geq 0. \tag{4.30}$$

Then it follows from (4.29) and (4.30) that

$$\sum_{u_i \in C_2} b_{ij} > 0. \tag{4.31}$$

If y_j is moved from C_1 into C_2, the modularity change is

$$\Delta Q = \sum_{u_i \in C_2} b_{ij} - \sum_{u_i \in C_1} b_{ij} > 0. \tag{4.32}$$

However, the fine-tuning procedure guarantees that moving any vertex from C_1 into C_2 does not result in modularity increase. This is a contradiction. Thus the assertion (4.28) is proved.

Now we prove the proposition by contradiction. Assume $\exists y_j \in C_k, \forall u_i \in C_k, a_{ij} = 0$. Then

$$b_{ij} = -k_i d_j / m^2 < 0 \quad \Rightarrow \quad \sum_{u_i \in C_k} b_{ij} < 0. \tag{4.33}$$

This contradicts the assertion (4.28) proved above. □

As $a_{ij} = 0$ is equivalent to $d_{ij} = \infty$ and $L_{ij} = \infty$, and hence there is no path from u_i to y_j, the proposition is interpreted as that every output is input-reachable within the same community.

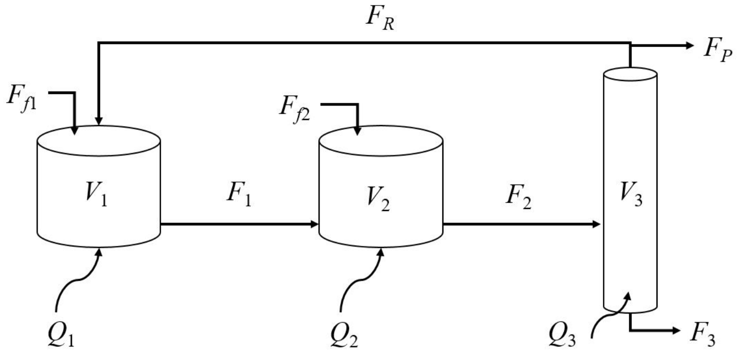


Figure 4.1: Reactor-separator process.

4.4 Chemical Process Network Example

Now we consider a reactor-separator process, which has been widely used as a benchmark for studying distributed model predictive control in the literature [234, 397, 421]. In this process, first-order reactions $\mathrm{A}\xrightarrow{1}\mathrm{B}\xrightarrow{2}\mathrm{C}$ take place in two continuously stirred tank reactors (CSTRs) in series followed by a separator which produces a vapor phase recycled and a liquid phase withdrawn as the product. The second CSTR has a fresh feed stream (see Fig. 4.1). B is the desired product. Detailed description and dynamical equations involving 12 states (x_{A1}, x_{B1}, T_1, V_1, x_{A2}, x_{B2}, T_2, V_2, x_{A3}, x_{B3}, T_3, V_3, labeled as x_1 to x_{12}, respectively) can be found in Appendix A. The operating conditions and parameters are listed in Table A.1. F, Q, T, V, x stand for flow rates, heat exchange rates, temperatures, holdup volumes, and molar fractions, respectively. m, C_p, k, E, ΔH, E, α, ρ denote molality, heat capacity, pre-exponential coefficient for reaction rate, enthalpy change, activation energy, relative volatility and density respectively. Subscripts A, B, C correspond to chemical species, $1, 2, 3$ to reactor and separator units, and f, R to the feed and recycle streams, respectively.

The manipulated variables are the feed flow rates ($u_1 = F_{f1}$, $u_2 = F_{f2}$), inventory flow rates ($u_3 = F_1$, $u_4 = F_2$, $u_5 = F_3$), recycle rate ($u_6 = F_R$) and the heat exchange rates ($u_7 = Q_1$, $u_8 = Q_2$, $u_9 = Q_3$). The controlled variables include the liquid holdup volumes ($y_1 = T_1$, $y_2 = T_2$, $y_3 = T_3$), temperatures ($y_4 = T_1$, $y_5 = T_2$, $y_6 = T_3$), and the concentrations of B ($y_7 = x_{B1}$, $y_8 = x_{B2}$, $y_9 = x_{B3}$). Fractional deviations from steady state values are used to define the dimensionless flow rates, heat exchange rates, holdup volumes and concentrations. The dimensionless temperature is defined as the deviation

Table 4.1: Input-output affinities in the reactor-separator process

a_{ij}	V_1	V_2	V_3	T_1	T_2	T_3	x_{B1}	x_{B2}	x_{B3}
F_{f1}	0.462	0	0	0.566	0.350	0.241	0.492	0.321	0.218
F_{f2}	0	0.578	0	0.232	0.667	0.357	0.212	0.578	0.311
F_1	0.719	0.918	0	0.293	0.383	0.256	0.322	0.366	0.237
F_2	0	1.000	0.769	0.270	0.323	0.454	0.307	0.360	0.568
F_3	0	0	0.573	0.183	0.153	0.253	0.140	0.122	0.103
F_R	0.665	0	0.665	0.284	0.217	0.270	0.498	0.324	0.565
Q_1	0	0	0	0.475	0.313	0.223	0.263	0.205	0.157
Q_2	0	0	0	0.213	0.528	0.313	0.156	0.231	0.172
Q_3	0	0	0	0.270	0.209	0.454	0.185	0.154	0.126

[1]Numbers in red, blue, green and magenta correspond to the four subsystems.

from the steady state over 40 K (assumed to be the allowed deviation of temperature from the steady state).

The weighted system digraph is constructed as in Fig. 4.2. By searching the shortest paths between the inputs and the outputs, the input-output affinities are calculated and shown in Table 4.1. Fig. 4.3 describes the procedure of community detection involving bisections and fine-tuning. The system is ultimately decomposed into 4 communities. The intra-community affinities in the final decomposition generally have significantly higher values than the inter-community ones (see Table 4.1), indicating a significant difference between intra-subsystem and inter-subsystem short-time interactions. According to the decomposition, the temperatures and intermediate concentrations are controlled by the heat transfer and feed flow rates; feed flow rates are chosen because of the low temperature of the feed streams, which results in significant effects on the reactor temperatures as the feed flow rates change. The liquid levels of the reactors are controlled by the flow rate from reactor 1 to reactor 2, F_1. The liquid level of the separator and the final product concentration are controlled by the flow rates related to the separator.

The distribution of the resulting subsystems is shown in Fig. 4.4. All the subsystems are located at a single or adjacent process units. Subsystem 1 is located in the first reactor; subsystem 2 includes the second reactor and the separator; subsystem 3 includes both reactors; subsystem 4 is located at the separator. Subsystems 1 and 2 are associated with the energy balance of the system and the reaction kinetics (the change in feed flow rates influences the product concentration), and subsystems 3 and 4 are

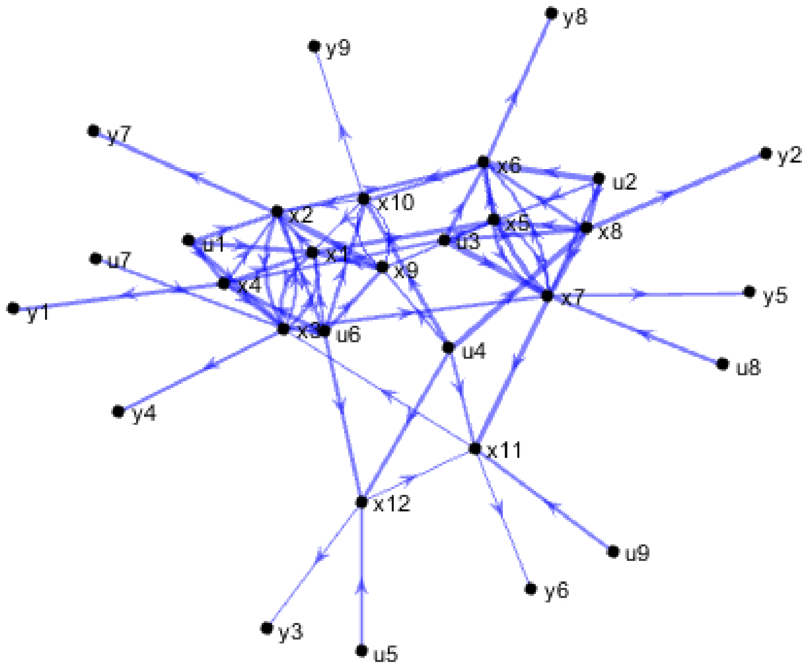


Figure 4.2: The weighted system digraph of the reactor-separator process. The line width is proportional to the edge weights.

associated with the mass balances. Hence the community detection proposed here generates a decomposition that combines spatial distribution and underlying physical laws, i.e., mass and energy balances. This should be contrasted with other decompositions possible for this system. One is a *unit-based* decomposition that contains 3 subsystems corresponding to the 3 units in the process, which can be generated either by community detection [180] or a minimally coupled equal-size partition [303] in the unweighted system digraph. The other is an intuitive *mass and energy balance-based* decomposition in line with [234], obtained by grouping heat exchange rates with temperatures, feed and recycle rates with product concentrations, and inventory flow rates with holdup volumes.

A detailed comparison of different decompositions for this system in terms of computational cost and control performance has been performed in [326, 327]. Specifically, a non-cooperative iterative distributed model predictive control scheme was implemented. The control performance was characterized by integrated squares of the error and the control signal (ISE + ISC) for output regulation. A sequential quadratic programming

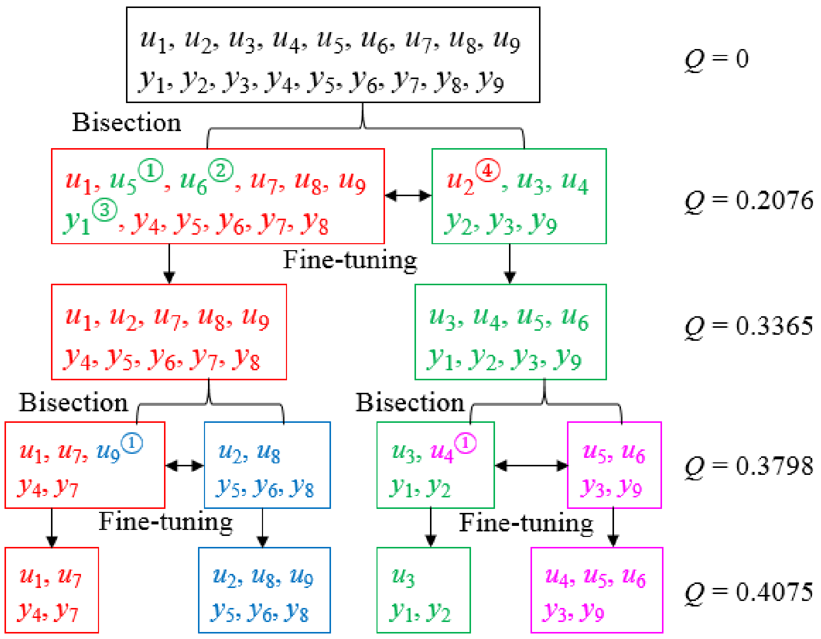


Figure 4.3: The procedure of community detection for input-output partitioning. The superscripts in circle represent the order of moving the nodes to the other community in the corresponding fine-tuning step.

(SQP) solver was used for the solution of the optimal control problems. (See the aforementioned references for details.) In [327], the decomposition proposed in the present chapter was shown to result in a shorter computational time (3.8 min, 16% of that of centralized MPC), compared to the unit-based (4.7 min) and intuitive decompositions (4.3 min). The control performance of the proposed decomposition (ISE + ISC $= 6.99 \times 10^3$) is slightly inferior to that of the unit-based decomposition (6.11×10^3), yet still close to that of centralized MPC (5.65×10^3), and significantly better than the one under the intuitive decomposition (1.04×10^4).

Note that, as expected for a nonlinear system, the decomposition will in general depend on the operating steady state. For example, if the feed stream is preheated to a temperature close to the temperature of the process, the process will not be heated or cooled significantly by a change in the feed flow rate. For the case where $T_0 = 432.0$ K (close to $T_1 = 432.4$ K and $T_2 = 427.1$ K), the method generates a decomposition where the temperatures are controlled only by heat exchange rates, and the feed streams are used to control the product concentrations (see Table 4.2).

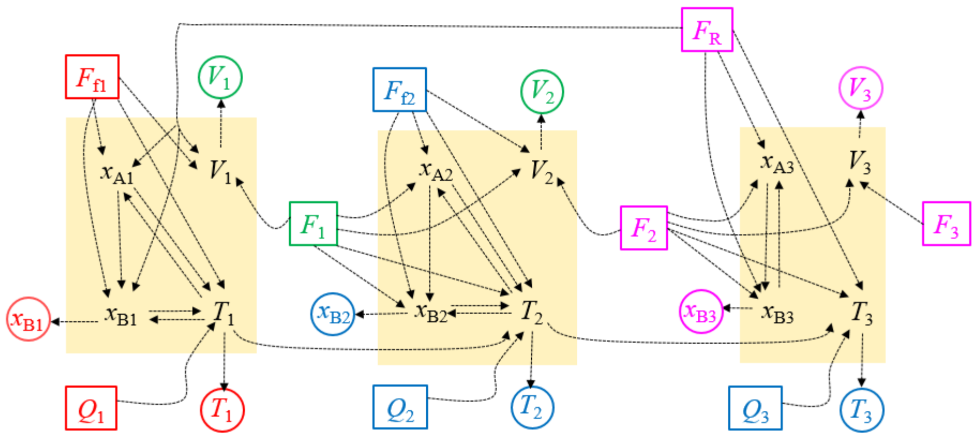


Figure 4.4: The distribution of subsystems in the process network. The system digraph (dashed) is embedded in the flowsheet (solid). The inputs and the outputs are boxed and circled respectively, and colored in red, blue, green and magenta for the four communities.

Table 4.2: Input-output communities at different feed temperatures

Subsystem	Inputs	Outputs
1	F_{f1}, F_{f2}, Q_1	x_{B2}, T_1, x_{B1}
2	F_{f2}, Q_1, Q_2, Q_3	x_{B2}, T_1, T_2, T_3
3	F_1	V_1, V_2
4	F_2, F_3, F_R	V_3, x_{B3}

[2]The blue, red and black are under $T_0 = 359.1$ K, $T_0 = 432.0$ K and both, respectively.

4.5 Conclusions

In this chapter we have proposed a method of input-output partitioning to design a distributed (model predictive) control architecture for a class of nonlinear control affine network systems. The method is based on community detection in input-output bipartite graphs weighted by affinities. The definition of affinity incorporates relative degrees (which capture the structure of interconnections on the system digraph) and response sensitivities (the coefficients of the linearized equations). Barber's modularity maximization method through recursive bisections is employed to identify the optimal partition. The resulting input-output communities are located within adjacent physical units, containing outputs reachable from intra-community inputs, while input-output interactions inside the communities over short time scales are stronger than inter-community interactions. The resulting decompositions thus appear well-suited for the application of nonlinear distributed model predictive controllers with short prediction horizon. However, the proposed approach does not address the effect of input constraints or uncertainty on the decomposition, or the impact of decomposition on closed-loop stability. These will be subjects of future research.

Chapter 5
Relative Time Averaged Gain Array for Distributed Control-Oriented Network Decomposition

5.1 Introduction

Mass and energy integration, which has become a rule rather than an exception in modern chemical and energy plants [18], results in large-scale and complex networks. The control of complex networks is difficult. Recent research has focused on distributed control strategies [11], as a middle ground between centralized and decentralized control approaches. Distributed control assumes the decomposition of a large-scale network into constituent subsystems, with the corresponding controllers coordinated through some level of information sharing. Regarding the control of process systems, distributed model predictive control (MPC), where the optimal control problems are solved using distributed optimization algorithms, has been recognized as a useful approach due to its capability of addressing process constraints and optimizing process economics while reducing the computational complexity (see [362, 63, 288] for reviews and references).

In order to implement any distributed control scheme, a decomposition of the network must be predetermined. Large-scale system decomposition utilizing the underlying network topology in the context of decentralized control has a long history. These efforts are based on the detection of strongly connected subnetworks [49, 434], and rearranging these subnetworks in a hierarchy [379], so that by stabilizing the subnetworks, the stability of the entire system can be guaranteed. With given inputs and outputs, an algorithm was proposed to decompose the network into subsystems satisfying input- and output-reachability properties, with the links between the subsystems exhibiting a hierarchical pattern [321]. Nested ε-decomposition [372] was proposed to neglect small elements in the coefficient matrix. However, the special network structures required for these methods are usually restrictive mainly due to the existence of recycles and strong connectivity in process systems. Neglecting the intensity of interactions, Hangos and Tuza [146] formulated the network decomposition as an optimization problem to minimize the interconnecting edges with size restrictions in the corresponding subsystems (similar formulations are also found in [303] as well as in [40] on fault detection system decomposition). These methods, however, involve computationally demanding and nonscalable integer programming formulations, and lack a detailed analysis of interactions within the system which favors the resulting control performance. Evidently, the systematic decomposition of a network system into a distributed architecture remains an open problem.

5.1.1 Relative gain analysis

Instead of analyzing the underlying graph representing the detailed state-space model, for process control, the classical input-output pairing/partitioning problems based on input-output models are usually considered for designing controllers. The input-output partitioning aims at dividing the input and output variables into groups such that the response characteristics between different groups are well decoupled, and designing the decentralized/block decentralized control structure for the system. The input-output pairing problem has been addressed extensively during a half century mostly in the framework of relative gain analysis (see [258, 386, 192] for reviews) together with the Niederliński index [298], which guarantees decentralized integral controllability [470], and has been widely applied in the practice of plant-wide control.

The classical relative gain array (RGA) [47] is defined as the element-by-element product of the steady state gain array and its transposed inverse (or pseudo-inverse for non-square systems [345, 54]):

$$\Lambda = G(0) \circ G(0)^{-\top}, \tag{5.1}$$

where $G(s)$ is the transfer function matrix and hence $G(0)$ represents the steady state gains obtained for $s = 0$. Any element of the RGA can be interpreted as the ratio of open-loop and closed-loop steady state gains. Hence if the (i, j)-th element of Λ, λ_{ij}, is close to 1, then the SISO loop comprising of the i-th output y_i and the j-th input u_j is well decoupled from other SISO loops, and y_i and u_j should be paired. Integer programming formulations were proposed for optimally pairing the inputs and outputs [199, 200]. When SISO control is not sufficient for decoupling input-output interactions, block-decentralized configurations can be generated by the block RGA (BRGA) [248, 189] method by trial-and-error combinations of the input and output variables. The RGA method has also been extended to unstable plants with poles on the right half complex plane [168] and integrators [451, 12].

The output response over an input is intrinsically dependent on the input frequencies. By replacing $G(0)$ with $G(j\omega)$ which shows the responses of the outputs over sinusoidal inputs, the frequency-dependent dynamical RGA (DRGA) is obtained [450, 424]. In the effective RGA (ERGA) [457], the bandwidth (crossover frequency) ω_{ij}^{c} obtained from frequency analysis was used as a factor to modify the steady state gain by replacing $g_{ij}(0)$ with $g_{ij}(0)\omega_{ij}^{c}$. In the effective relative energy array (EREA) [272], the interaction measure was modified in the form of $g_{ij}^{2}(0)\omega_{ij}^{c}$, which is related to the $\mathcal{H}_2$-norm of transfer functions.

To account for the response in finite time in addition to the steady state or ultimate response, the steady state gain was replaced by a finite time average $\int_0^{\theta} y_i(t)dt$ to obtain the relative dynamic array (RDA) [450] and by $\frac{1}{\theta-d_{ij}}\int_{d_{ij}}^{\theta} y_i(t)dt$ to obtain the average relative gain array (AGRA) [120], where d_{ij} is the time delay between u_j and y_i. In the relative normalized gain array (RNGA) proposed [149], the interaction measure, the normalized gain, is defined as the ratio of the steady state gain over the average residence time, $\tau_{\mathrm{AR}}^{ij} := \int_0^{\infty}[1-y_i(t)/y_i(\infty)]dt$. The average residence time can be viewed as the time scale for y_i to reach its steady state, and hence the normalized gain is a "velocity" of the response of y_i.

The analyses of dynamic behavior through relative gain interaction measures gives overall evaluations of the effects of the inputs on the outputs, and thus provide useful tools for control structure design. The selection of input and output variables is also facilitated by the information of RGA singular values [134, 385]. However, since every input affects its nearby outputs first and then the outputs faraway with increasing time, the paired inputs and outputs can be distant on the network, and the paths from inputs to outputs in different subsystems may not be disjoint, causing undesirable interplays in the closed-loop system, although superficially well-decoupled with a relative gain characterization. Therefore, the relative gain analysis alone is not sufficient for the network decomposition for distributed control architecture design without taking the network topology into consideration.

5.1.2 Relative degree and sensitivity analysis

The relative degree between an input and an output [74], defined as the lowest order of the time derivative of the output that involves the input, reflects the sluggishness of the output response over the input, and is equal to the number of states that the shortest input-output path must pass through on the system digraph [346], thus characterizing the input-output closeness in the network. Schné and Hangos [363] proposed the relative degree-based SISO control configuration design by performing weighted bipartite matching based on relative degree. Control configurations with different extents of decentralization based merely on relative degrees were produced through hierarchical agglomerative clustering [153, 188] or hierarchical divisive clustering [152].

Since the relative degree is a structural measure and does not contain any information related to the magnitude of the output response over the input, the sensitivity coefficient, i.e., the corresponding lowest order nonzero initial time derivative, which has been used as the supplemental information to relative degree for input selection in economic MPC [97] and sensor arrangement [467], should also be considered for input-output partitioning. Tang and Daoutidis [408] defined an input-output interaction measure incorporating both relative degree and sensitivity information by the shortest path length on a weighted system digraph, and applied community detection algorithm [23] in the input-output bipartite graph to generate an optimal partition. Recently, Yin and Liu [468] proposed an input-output pairing framework called relative sensitivity array (RSA). In this approach, the inputs and outputs are first separated into several groups according to their relative degrees. For the inputs and outputs in the group corresponding to the relative degree r, the array of sensitivities S_r gives a relative sensitivity array $\Psi_r = S_r \circ S_r^{-\top}$. An input and an output with a corresponding RSA element close to 1 are suggested to be paired. The pairing is performed in the ascending order of relative degrees, thus generating input-output pairs with small relative degrees and well-decoupled sensitivities.

The analysis of relative degrees and sensitivities generates distributed control architectures, where each subsystem includes inputs and outputs closely connected in the network. The underlying reason is that relative degrees and sensitivities give the order and magnitude of the lowest order term of the response, which can be dominating locally within a sufficiently short time interval. However, since no information on the response after this infinitesimal time is contained, the subsystems are not guaranteed to be dynamically well decoupled. Therefore it is necessary to find a middle ground between the descriptions of the sufficiently short and sufficiently long time responses.

5.1.3 This work

In this work, we propose a new input-output interaction measure, $G(1/\tau)$, namely the relative time-averaged gain array (RTAGA), as a middle ground to unify the relative gain analysis and relative degree and sensitivity analysis, to characterize input-output interactions within the closed-loop time scale τ.

The remainder of this chapter is organized as follows. Section 5.2 addresses the concept of RTAGA. In Subsection 5.2.1, we establish that $G(1/\tau)$ is a weighted average of the output responses over the inputs within a time scale of τ, and hence can be used as an input-output interaction measure. The resulting RTAGA $\Lambda(1/\tau)$ embodies the coupling of inputs and outputs within the time scale of τ. In Subsection 5.2.2, we prove that the RTAGA approaches the steady state RGA and RSA as $\tau \to \infty$ and $\tau \to 0$ respectively. In Subsection 5.2.3 we confer the relative time-averaged gain an interpretation based on signal flow graph. Section 5.3 addresses the application of RTAGA. For large-scale systems, we introduce in Subsection 5.3.1 the application of community detection based on RTAGA for efficient input-output partitioning. In Subsection 5.3.2 we demonstrate that τ should be chosen as the time scale of closed-loop dynamics, and discuss how to determine τ for distributed control. Finally, the RTAGA framework proposed here is examined by an application to the distributed model predictive control of a reactor-separator process in Section 5.4.

5.2 Relative Time-Averaged Gain Array (RTAGA)

We note that the transfer function matrix $G(\cdot)$ requires a variable with a reciprocal time dimension, with G(0) equal to the array of steady state gains for step inputs and $G(j\omega)$ (evaluated on the imaginary axis) reflecting the ultimate response for sinusoidal inputs. We now consider the transfer function evaluated on the positive real axis, $G(1/\tau)$, where $\tau > 0$ has a dimension of time.

5.2.1 Definition and interpretation

We consider a linearized system

$$\dot{x} = Ax + Bu, \quad y = Cx. \tag{5.2}$$

where u, x and y are the vectors of inputs, state variables and outputs respectively. Let the system state be at the origin under zero inputs for $t < 0$. After $t = 0$, the j-th input is subject to a unit step input and the other inputs are kept at zero. The response of the i-th output, $y_i(t)$ is then given by its Laplace transform

$$\bar{y}_i(s) = g_{ij}(s)\bar{u}_j(s) = \frac{g_{ij}(s)}{s}. \tag{5.3}$$

Now we evaluate the intensity of the response of y_i for the step change in u_j averaged with an exponential distribution function $f(t;\tau) = \frac{1}{\tau}e^{-t/\tau}$, where τ is a parameter that characterizes the decaying rate of the distribution function. The $f(t;\tau)$ is used as a normalized (i.e., $\int_0^\infty f(t;\tau)dt = 1$) weight function to average the time-domain response. Then the average is expressed as

$$\int_0^\infty y_i(t)f(t;\tau)dt = \frac{1}{\tau}\int_0^\infty y_i(t)e^{-t/\tau}dt = \frac{1}{\tau}\bar{y}_i\left(\frac{1}{\tau}\right). \tag{5.4}$$

According to (5.3), we have

$$\int_0^\infty y_i(t)f(t;\tau)dt = g_{ij}(1/\tau). \tag{5.5}$$

Hence we see that $g_{ij}(1/\tau)$ can be interpreted as the intensity of the response of y_i over the step change in u_j weighted with an exponential distribution that decays at a characteristic time scale of τ. $g_{ij}(1/\tau)$ is the open-loop time-averaged gain of y_i over u_j within the time scale of τ.

Based on the gain array $G(1/\tau) = [g_{ij}(1/\tau)]$, the relative gain array at time scale τ can be calculated as

$$\Lambda(1/\tau) = G(1/\tau) \circ G(1/\tau)^{-\top}. \tag{5.6}$$

Similar to the steady state RGA, the (i, j)-th element of $\Lambda(1/\tau)$, $\lambda_{ij}(1/\tau)$, can be viewed as the ratio of open-loop time-averaged gain and closed-loop time-average gain of y_i over u_j within the time scale of τ. Also similarly, the optimal input-output pairing can be chosen such that the $\Lambda(1/\tau)$ is the closest to the identity matrix using a mathematical programming formulation [201]. The optimal partitioning of inputs and outputs into MIMO subsystems (not restricted to an input-output pairing into SISO subsystems) will be discussed later.

Remark 5.1. *In the steady state RGA analysis, if integrators appear in a column or a* *row, the integrators are simply ignored [12, 169]. By employing RTAGA, the integrator $1/s$ is evaluated at $s = 1/\tau$ as τ, i.e., the total integrated amount during the time scale of interest. Since the multiplication of a row or a column in $G(1/\tau)$ with the same number does not change the resulting RTAGA $\Lambda(1/\tau)$, the integrators can also be directly ignored.*

Remark 5.2. *The bandwidth ω_c has been incorporated into the input-output interaction measure as a multiplying factor [457, 272]. The interaction measure can become unbounded as ω_c increases and thus exaggerating the input-output interactions. By using time-averaged gains $G(1/\tau)$ as the interaction measures, this can be avoided. For example, for first-order processes with fixed steady state gain and bandwidth ω_c, $G(s) = 1/(1 + s/\omega_c)$, $G(1/\tau) = \omega_c\tau/(1 + \omega_c\tau)$ increases with increasing ω_c, but is bounded by the steady state gain of 1. When ω_c is small, $G(1/\tau) \approx \omega_c\tau$ is proportional to ω_c, reducing to the previous methods.*

5.2.2 Time scale limits

When τ becomes large enough, the weight function $f(t;\tau) = \frac{1}{\tau}e^{-t/\tau}$ favors the response at long times, and hence the averaged gains approach the steady state gains. The traditional steady state RGA can be viewed as a limiting case, i.e.,

$$\Lambda(0) = \lim_{\tau\to\infty} \Lambda(1/\tau). \tag{5.7}$$

We now consider the case when τ is sufficiently small.

Lemma 5.1. *When $\tau \to 0$, the open-loop time-averaged gain within the time scale τ can be expressed as*

$$g_{ij}(1/\tau) = s_{ij}\tau^{r_{ij}} + O(\tau^{r_{ij}+1}). \tag{5.8}$$

where r_{ij} is the relative degree between u_j and y_i, i.e., the smallest positive integer such that

$$\left.\frac{d^{r_{ij}}y_i}{dt^{r_{ij}}}\right|_{t=0} \neq 0, \tag{5.9}$$

and s_{ij} is the corresponding sensitivity coefficient

$$s_{ij} = \left.\frac{d^{r_{ij}}y_i}{dt^{r_{ij}}}\right|_{t=0}, \tag{5.10}$$

when u_j is subject to a unit step change and all other inputs are kept at zero.

Proof. According to the definition of time-averaged gain within the time scale of τ, when u_j is subject to a unit step change and all other inputs are zero,

$$g_{ij}(1/\tau) = \frac{1}{\tau}\int_0^\infty y_i(t)e^{-t/\tau}dt = \int_0^\infty y_i(\tau\sigma)e^{-\sigma}d\sigma. \tag{5.11}$$

As a function of τ, for any $r \in \mathbb{N}$, we have

$$\left.\frac{d^r g_{ij}(1/\tau)}{d\tau^r}\right|_{\tau=0} = \int_0^\infty \left.\frac{d^r y_i(t)}{dt^r}\right|_{t=0} \sigma^r e^{-\sigma}d\sigma. \tag{5.12}$$

According to the definitions of relative degree (5.9) and sensitivity (5.10),

$$\left.\frac{d^r g_{ij}(1/\tau)}{d\tau^r}\right|_{\tau=0} = 0, \quad r = 0, 1, \ldots, r_{ij}-1 \tag{5.13}$$

and

$$\left.\frac{d^{r_{ij}} g_{ij}(1/\tau)}{d\tau^{r_{ij}}}\right|_{\tau=0} = \int_0^\infty s_{ij}\sigma^{r_{ij}}e^{-\sigma}d\sigma = s_{ij}r_{ij}! \tag{5.14}$$

which implies the conclusion of the lemma. □

Assumption 5.1. *Assume that the inputs and outputs can be divided into n groups: $\{u_j, y_i | 1 \leq i, j \leq m_1\}$, $\{u_j, y_i | m_1 + 1 \leq i, j \leq m_1 + m_2\}$, ..., $\{u_j, y_i | m_1 + \cdots + m_{n-1} + 1 \leq i, j \leq m_1 + \cdots + m_{n-1} + m_n\}$, such that each input (output) in the r-th group ($1 \leq r \leq n$) has a relative degree no smaller than r with any other output (input). Then according to Lemma 5.1, for any output y_i in the r-th group and any input u_j in the r'-th group, the relative degree between them should be at least $\max(r, r')$, and hence $g_{ij}(1/\tau) \sim O(\tau^{\max(r,r')})$. Therefore $G(1/\tau)$ can be expressed in the following form:*

$$G(1/\tau) = \begin{bmatrix} G_{11} & G_{12} & \cdots & G_{1n} \\ G_{21} & G_{22} & \cdots & G_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ G_{n1} & G_{n2} & \cdots & G_{nn} \end{bmatrix}, \tag{5.15}$$

in which the diagonal blocks $G_{rr} = \tau^r S_r + O(\tau^{r+1})$ and off-diagonal blocks $G_{rr'} \sim O(\tau^{\max(r,r')}), r \neq r'$, where S_r is the array of corresponding sensitivities.

Assumption 5.2. *All the sensitivity matrices $S_1, S_2, \ldots, S_n$ are invertible.*

Theorem 5.1. *Under Assumptions 5.1 and 5.2, the RTAGA $\Lambda(1/\tau) = G(1/\tau) \circ G(1/\tau)^{-\top}$ can be written as*

$$\Lambda(1/\tau) = \begin{bmatrix} \Psi_1 & 0 & \cdots & 0 \\ 0 & \Psi_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Psi_n \end{bmatrix} + O(\tau), \tag{5.16}$$

in which $\Psi_r = S_r \circ S_r^{-\top}$ is called the r-th relative sensitivity array (RSA) [468].

Proof. First we prove the proposition in the presence of only two relative degrees ($n = 2$).

$$G(1/\tau) = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}, \tag{5.17}$$

in which $G_{11} = \tau S_1 + O(\tau^2)$, $G_{22} = \tau^2 S_2 + O(\tau^3)$, $G_{12}, G_{21} \sim O(\tau^2)$, where $S_1, S_2 \sim O(1)$. Then

$$\begin{aligned} (G^{-1}(1/\tau))_{11} &= (G_{11} - G_{12}G_{22}^{-1}G_{21})^{-1} = \tau^{-1}S_1^{-1} + O(1), \\ (G^{-1}(1/\tau))_{12} &= -G_{11}^{-1}G_{12}(G_{22} - G_{21}G_{11}^{-1}G_{12})^{-1} = -\tau^{-1}S_1^{-1}S_{12}S_2^{-1} + O(1) \\ (G^{-1}(1/\tau))_{21} &= -G_{22}^{-1}G_{21}(G_{11} - G_{12}G_{22}^{-1}G_{21})^{-1} = -\tau^{-1}S_2^{-1}S_{21}S_1^{-1} + O(1) \\ (G^{-1}(1/\tau))_{22} &= (G_{22} - G_{21}G_{11}^{-1}G_{12})^{-1} = \tau^{-2}S_2^{-1} + O(\tau^{-1}). \end{aligned} \tag{5.18}$$

Namely,

$$G^{-1}(1/\tau) = \begin{bmatrix} \tau^{-1}S_1^{-1} + O(1) & -\tau^{-1}S_1^{-1}S_{12}S_2^{-1} + O(1) \\ -\tau^{-1}S_2^{-1}S_{21}S_1^{-1} + O(1) & \tau^{-2}S_2^{-1} + O(\tau^{-1}) \end{bmatrix}. \tag{5.19}$$

Hence

$$\begin{aligned} \Lambda(1/\tau) = G(1/\tau) \circ G^{-\top}(1/\tau) &= \begin{bmatrix} S_1 \circ S_1^{-\top} + O(\tau) & O(\tau) \\ O(\tau) & S_2 \circ S_2^{-\top} + O(\tau) \end{bmatrix} \\ &= \begin{bmatrix} \Psi_1 & 0 \\ 0 & \Psi_2 \end{bmatrix} + O(\tau) \quad (\tau \to 0). \end{aligned} \tag{5.20}$$

Next we prove the proposition by mathematical induction. Let the proposition hold for n, then for $n+1$,

$$G(1/\tau) = \begin{bmatrix} G_{11} & \cdots & G_{1n} & G_{1,n+1} \\ \vdots & \ddots & \vdots & \vdots \\ G_{n1} & \cdots & G_{nn} & G_{n,n+1} \\ G_{n+1,1} & \cdots & G_{n+1,n} & G_{n+1,n+1} \end{bmatrix}. \tag{5.21}$$

Denote

$$\begin{gathered} \bar{G}_{nn} = \begin{bmatrix} G_{11} & \cdots & G_{1n} \\ \vdots & \ddots & \vdots \\ G_{n1} & \cdots & G_{nn} \end{bmatrix}, \quad \bar{G}_{n,n+1} = \begin{bmatrix} G_{1,n+1} \\ \vdots \\ G_{n,n+1} \end{bmatrix}, \\ \bar{G}_{n+1,n} = \begin{bmatrix} G_{n+1,1} & \cdots & G_{n+1,n} \end{bmatrix}, \quad \bar{G}_{n+1,n+1} = G_{n+1,n+1}. \end{gathered} \tag{5.22}$$

We have

$$G(1/\tau) = \begin{bmatrix} \bar{G}_{nn} & \bar{G}_{n,n+1} \\ \bar{G}_{n+1,n} & \bar{G}_{n+1,n+1} \end{bmatrix}, \tag{5.23}$$

where $\bar{G}_{n+1,n}$, $\bar{G}_{n,n+1}$ and $\bar{G}_{n+1,n+1}$ are of $O(\tau^{n+1})$, and $\bar{G}_{nn}^{-1} \sim O(\tau^{-n})$. Similar as in (5.18), $G^{-1}(1/\tau)$ can be expressed as

$$G^{-1}(1/\tau) = \begin{bmatrix} (\bar{G}^{-1})_{nn} & (\bar{G}^{-1})_{n,n+1} \\ (\bar{G}^{-1})_{n+1,n} & (\bar{G}^{-1})_{n+1,n+1} \end{bmatrix}, \tag{5.24}$$

where

$$\begin{aligned} (\bar{G}^{-1})_{nn} &= (\bar{G}_{nn} - \bar{G}_{n,n+1}\bar{G}_{n+1,n+1}^{-1}\bar{G}_{n+1,n})^{-1} = \bar{G}_{nn}^{-1} + O(\tau^{-n+1}), \\ (\bar{G}^{-1})_{n,n+1} &= -\bar{G}_{nn}^{-1}\bar{G}_{n,n+1}(\bar{G}_{n+1,n+1} - \bar{G}_{n+1,n}\bar{G}_{nn}^{-1}\bar{G}_{n,n+1})^{-1} \\ &= -\bar{G}_{nn}^{-1}\bar{G}_{n,n+1}\bar{G}_{n+1,n+1}^{-1} + O(\tau^{-n+1}) \\ (\bar{G}^{-1})_{n+1,n} &= -\bar{G}_{n+1,n+1}^{-1}\bar{G}_{n+1,n}(\bar{G}_{nn} - \bar{G}_{n,n+1}\bar{G}_{n+1,n+1}^{-1}\bar{G}_{n+1,n})^{-1} \\ &= -\bar{G}_{n+1,n+1}^{-1}\bar{G}_{n+1,n}\bar{G}_{nn}^{-1} + O(\tau^{-n+1}) \\ (\bar{G}^{-1})_{n+1,n+1} &= (\bar{G}_{n+1,n+1} - \bar{G}_{n+1,n}\bar{G}_{nn}^{-1}\bar{G}_{n,n+1})^{-1} = \bar{G}_{n+1,n+1}^{-1} + O(\tau^{-n}). \end{aligned} \tag{5.25}$$

The dominating terms

$$\begin{aligned} &\bar{G}_{nn}^{-1} \sim O(\tau^{-n}), \quad -\bar{G}_{n+1,n+1}^{-1}\bar{G}_{n+1,n}\bar{G}_{nn}^{-1} \sim O(\tau^{-n}), \\ &-\bar{G}_{nn}^{-1}\bar{G}_{n,n+1}\bar{G}_{n+1,n+1}^{-1} \sim O(\tau^{-n}), \quad \bar{G}_{n+1,n+1}^{-1} \sim O(\tau^{-(n+1)}). \end{aligned} \tag{5.26}$$

Therefore

$$\begin{aligned}\Lambda(1/\tau) &= \begin{bmatrix} \bar{G}_{nn} \circ \bar{G}_{nn}^{-\top} & 0 \\ 0 & \bar{G}_{n+1,n+1} \circ \bar{G}_{n+1,n+1}^{-\top} \end{bmatrix} + O(\tau) \\ &= \begin{bmatrix} \Psi_1 & 0 & \cdots & 0 & 0 \\ 0 & \Psi_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & \cdots & \Psi_n & 0 \\ 0 & 0 & \cdots & 0 & \Psi_{n+1} \end{bmatrix} + O(\tau).\end{aligned} \tag{5.27}$$

Thus the proposition holds for $n+1$. The proposition is thus proved. $\square$

We see from the above theorem that when $\tau \to 0$, the RTAGA within the time scale τ approaches a block-diagonal matrix of relative sensitivity arrays (RSAs). As a consequence, if the time scale of interest is sufficiently short, the input-output pairing or partitioning according to $\Lambda(1/\tau)$ can be performed according to RSAs Ψ_1, Ψ_2, ..., Ψ_n. Therefore Theorem 5.1 serves as a justification of the RSA-based input-output approach [468].

5.2.3 Small and big time scale expansions

Now we assume that matrix A in (5.2) is invertible. Consider the two cases when (i) τ is small enough but finite, and specifically, when $\tau < [\overline{\lambda}(A)]^{-1} = \underline{\lambda}(A^{-1})$; and (ii) τ is large enough but finite, and specifically, when $\tau > [\underline{\lambda}(A)]^{-1} = \overline{\lambda}(A^{-1})$, where $\overline{\lambda}$ and $\underline{\lambda}$ denotes the largest and the smallest eigenvalue modulus, respectively.

In the case (i), any eigenvalues of τA have a modulus smaller than 1, and hence $G(1/\tau)$ can be expanded into a Taylor series:

$$G(1/\tau) = C\left(1/\tau \cdot I - A\right)^{-1} B = C(\tau B) + C(\tau A)(\tau B) + C(\tau A)^2(\tau B) + \ldots, \tag{5.28}$$

whose (i,j)-th element, corresponding to the output y_i and the input u_j, is

$$g_{ij}(1/\tau) = c_i(\tau b_j) + c_i(\tau A)(\tau b_j) + c_i(\tau A)^2(\tau b_j) + \ldots, \tag{5.29}$$

where c_i is the i-th row of C and b_j is the j-th column of B, i.e.,

$$\begin{aligned} g_{ij}(1/\tau) = &\sum_{k=1}^{n} c_{ik}(\tau b_{kj}) + \sum_{k_1,k_2=1}^{n} c_{ik_1}(\tau a_{k_1k_2})(\tau b_{k_2j}) \\ &+ \sum_{k_1,k_2,k_3=1}^{n} c_{ik_1}(\tau a_{k_1k_2})(\tau a_{k_2k_3})(\tau b_{k_3j}) + \dots . \end{aligned} \tag{5.30}$$

Now we confer the interaction measure $g_{ij}(1/\tau)$ with a graph-theoretical interpretation. We construct a signal flow graph [252] for the scaled system (5.31) as the directed graph in which the nodes represent the inputs, states and outputs, and there is an edge from u_j to x_k if $b_{kj} \neq 0$, from x_k to x_l if $a_{lk} \neq 0$, and from x_k to y_i if $C_{ik} \neq 0$. We define the gain of edge u_jx_k by τb_{kj}, the gain of edge x_kx_l by τa_{lk}, and the gain of edge x_ky_i by c_{ik}. The gains of the edges are defined based on the following motivation. By scaling the time variable with τ, the system (5.2) can be written as

$$\frac{dx}{d(t/\tau)} = (\tau A)x + (\tau B)u, \quad y = Cx. \tag{5.31}$$

Each edge gain (τa_{lk}, τb_{kj}, c_{ik}) thus captures the effect of the value of one variable (deviation from origin) on the rate of change, and hence the time-averaged deviation from the origin in a short time scale of τ, of another variable. Note that every term in (5.30) is the product of gains on a path from u_j to y_i, namely the gain of a path from u_j to y_i, which captures the contribution of u_j on the change in y_i through the corresponding path. Each summation in (5.30) include the paths from u_j to y_i of a certain length (containing a certain number of edges). Therefore, $g_{ij}(1/\tau)$ is the total gain from u_j to y_i on the signal flow graph, characterizing the total short-time effect of u_j on y_i.

Similarly for the case (ii). When $\tau > [\underline{\lambda}(A)]^{-1} = \overline{\lambda}(A^{-1})$, any eigenvalue of A^{-1}/τ has a modulus smaller than 1, and we have

$$\begin{aligned} G(1/\tau) &= C\left(\frac{1}{\tau}I - A\right)^{-1} B = -CA^{-1}\left(I - \frac{1}{\tau}A^{-1}\right)^{-1} B \\ &= -CA^{-1}B - \frac{1}{\tau}CA^{-2}B - \frac{1}{\tau^2}CA^{-3}B - \dots \\ &= C(-A^{-1}B) + C(\tau A)^{-1}(-A^{-1}B) + C(\tau A)^{-2}(-A^{-1}B) + \dots . \end{aligned} \tag{5.32}$$

Denote by $\bar{a}_{lk}$ the (l,k)-th element of τA^{-1}, $\bar{b}_{kj}$ the (k,j)-th element of $-A^{-1}B$. Its

(i,j)-th element can be written as

$$g_{ij}(1/\tau) = \sum_{k=1}^{n} c_{ik}\bar{b}_{kj} + \sum_{k_1,k_2=1}^{n} c_{ik_1}\bar{a}_{k_1k_2}\bar{b}_{k_2j} + \sum_{k_1,k_2,k_3=1}^{n} c_{ik_1}\bar{a}_{k_1k_2}\bar{a}_{k_2k_3}\bar{b}_{k_3j} + \dots. \quad (5.33)$$

A graph-theoretical interpretation can be given in a similar fashion, if we define the gains of the edges u_jx_k, x_kx_l, and x_ky_i as $\bar{b}_{kj}$, $\bar{a}_{lk}$ and c_{ik}, respectively. We note that this definition of edge gains is in fact the coefficients in a transformation of the scaled system:

$$x = (\tau A)^{-1}\frac{dx}{d(t/\tau)} + (-A^{-1}B)u, \quad y = Cx. \quad (5.34)$$

The gains $\bar{a}_{lk}$ and $\bar{b}_{kj}$ capture the effect of the average rate of change of x_k within a time scale of τ, and the effect of the value of u_j, on the value that x_k settles on after this time scale. c_{ik} still captures the effect of x_k on y_i. $g_{ij}(1/\tau)$ in (5.33), as the total gain from u_j to y_i on this new signal flow graph, characterizes the total long-time effect of u_j on y_i.

Remark 5.3. *The idea of representing the information of systems by signal flow graphs was mentioned in a work of Johnston [182], where the gains capture the steady state interactions, and the input-output interactions are analyzed by the paths on signal flow graphs. In this subsection we have provided new constructions of signal flow graphs to interpret the time-averaged gain $g_{ij}(1/\tau)$ for both small and large τ values, and hence extended the application of signal flow graphs in the interaction analysis.*

5.3 Control Architecture Design

5.3.1 Network decomposition through community detection in input-output bipartite graph

With the RTAGA $\Lambda(1/\tau)$ characterizing the dynamic response decoupling within the time scale of τ, we consider the problem of optimal input-output partition. Although mathematical programming algorithms for SISO pairing are available [201], MIMO partition is still limited to block RGA [248] which has a combinatorial complexity for large-scale systems. Here we consider the use of community detection [114, 294] in the network analysis, which aims to detect subnetworks (called communities) such that the interconnections inside are significantly denser (stronger) than the interconnections between them. Specifically, we adopt the modularity maximization method, which has been the mainstream of community detection methods, and was used in our previous study [408].

We first define the input-output bipartite graph where each node represents an input or an output, and the edge joining u_j and y_i is given a weight $w_{ij} \in [0,1]$ characterizing the discrepancy between the open-loop and closed-loop time-averaged gain within the time scale τ. Especially when the open-loop and closed-loop gains have different signs, we should avoid grouping the corresponding input and output, and hence we define the weight as 0:

$$w_{ij} = \begin{cases} \lambda_{ij}, & 0 \leq \lambda_{ij} \leq 1 \\ 1/\lambda_{ij}, & \lambda_{ij} > 1 \\ 0, & \lambda_{ij} < 0 \end{cases} . \tag{5.35}$$

For a partition $\mathcal{P}$ of the inputs and outputs into communities $C_1, C_2, \dots$, the modularity of the partition [23] is defined as

$$Q(\mathcal{P}) = \sum_{C_k \in \mathcal{P}} \sum_{u_j, y_i \in C_k} \left[\frac{w_{ij}}{W} - \frac{W_{i\cdot} W_{\cdot j}}{W^2} \right], \tag{5.36}$$

in which

$$W_{i\cdot} = \sum_{j'} w_{ij'}, \ \ W_{\cdot j} = \sum_{i'} w_{i'j}, \ \ W = \sum_{i'} \sum_{j'} w_{i'j'} \tag{5.37}$$

are the degree of (total edge weights incident to) node y_i, degree of node u_j and the total degree of the network, respectively. Thus each term under the summation in (5.36) is the difference between w_{ij}/W, the fraction of edge weights between u_j and y_i, and $W_{i\cdot} W_{\cdot j}/W^2$, the expected fraction of edge weights between u_j and y_i when the edge weights are randomly redistributed while the degrees of all the nodes are kept unchanged. The modularity Q is therefore equal to the difference between the intra-community edge weights and its expected value. A good partition should be such that the intra-community weights significantly exceeds the expected value, and hence has a large Q value.

The community detection is formulated as an optimization problem that seeks to maximize modularity (5.36) over all partitions of nodes. Although this is an NP-hard problem in nature [46], approximate algorithms have been developed in recent years (see

References [114, 115] for comprehensive reviews). In this work, we adopt the Louvain algorithm (fast unfolding algorithm) [39] which is known to be the most computationally efficient so far. This algorithm involves the following steps:

1. Initialize by assigning each node into one community.

2. For each node, calculate the modularity increase when it is tentatively moved from its original community into the community of every other node. Choose the maximum modularity increase and perform the corresponding move.

3. Repeat the step 2 to adjust the commmunity affiliation of the nodes, until the modularity can no more be increased by moving any node between communities.

4. Agglomerate the communities into a single node. For any community, agglomerate all the intra-community edges into a self-loop with the weights summed. For every pair of communities, agglomerate the inter-community edges into a single edge with the weights summed.

5. Repeat steps 2 through 4 until each community is a single node. Then for every node at this stage, all the nodes that has been agglomerated into this node form a community.

5.3.2 Choice of time scale variable τ

To obtain the desirable network decomposition through community detection, we must first choose the time scale parameter τ. Since the $\Lambda(1/\tau)$ only characterizes the effects of the inputs on the outputs within the time scale of τ, it is implicitly assumed that the closed-loop dynamics evolves in a characteristic time scale of τ, i.e., the system should be stabilized with a time constant of approximately τ. Therefore, once the τ is determined, the controller should be designed such that the time constant of closed-loop dynamics, for example, the receding horizon of a model predictive control scheme or the time parameter of a internal model controller, should approximate τ in the sense of order-of-magnitude.

We use the following example [134] to illustrate that the mismatch between time scale parameter τ and closed-loop time constant may deteriorate the control performance. The example can be thus viewed as a proof-of-concept of our claim that τ should be the time constant of closed-loop dynamics. Consider a transfer function matrix

$$G(s) = \begin{bmatrix} \frac{-2}{10s+1} & \frac{1.5}{5s+1} & \frac{1}{s+1} \\ \frac{1.5}{5s+1} & \frac{-1}{s+1} & \frac{2}{10s+1} \\ \frac{1}{s+1} & \frac{2}{10s+1} & \frac{1.5}{5s+1} \end{bmatrix}. \tag{5.38}$$

Varying time scale τ, we find that the optimal pairing based on RTAGA is $y_1 - u_1/y_2 - u_3/y_3 - u_2$ when $\tau > 8$, and $y_1 - u_3/y_2 - u_2/y_3 - u_1$ when $\tau < 8$. For example, when $\tau = 10$,

$$G(1/10) = \begin{bmatrix} -1 & 1 & 0.909 \\ 1 & -0.909 & 1 \\ 0.909 & 1 & 1 \end{bmatrix}, \quad \Lambda(1/10) = \begin{bmatrix} \mathbf{0.549} & -0.026 & 0.477 \\ -0.026 & 0.477 & \mathbf{0.549} \\ 0.477 & \mathbf{0.549} & -0.026 \end{bmatrix}, \tag{5.39}$$

and when $\tau = 1$,

$$G(1/1) = \begin{bmatrix} -0.182 & 0.25 & 0.5 \\ 0.25 & 0.5 & 0.182 \\ 0.5 & 0.182 & 0.25 \end{bmatrix}, \quad \Lambda(1/1) = \begin{bmatrix} 0.157 & 0.039 & \mathbf{0.805} \\ 0.039 & \mathbf{0.805} & 0.157 \\ \mathbf{0.805} & 0.157 & 0.039 \end{bmatrix}. \tag{5.40}$$

The reason for this switch in the optimal pairing is that within the time scale of $\tau = 1$, the transfer functions $\pm 2/(10s + 1)$ with a time constant 10 can not give significant response, while within the time scale of $\tau = 10$, the response will exceed those of other transfer functions since the static gain is larger.

Lee and Edgar [212] studied the response of the closed-loop system with internal model control (IMC)

$$C_{ij}(s) = \frac{h_{ij}(s)}{g_{ij}(s)[1 - h_{ij}(s)]}, \quad h_{ij}(s) = \frac{g_{ij}^{+}(s)}{(\lambda s + 1)^{r_{ij}}}, \tag{5.41}$$

where $g_{ij}^{+}(s)$ is the non-invertible part of $g_{ij}(s)$ and r_{ij} is the relative degree between paired input u_j and output y_i. The closed-loop dynamics has a time scale of λ. When using $\lambda = 10$, the pairing obtained by $\tau = 10$ guarantees the stability while the other pairing destabilizes the system. When using $\lambda = 1$, the pairing obtained by $\tau = 1$ stabilizes the system while the pairing obtained by $\tau = 10$ is destabilizing (see Fig. 5 of [212]). Therefore, the problem of choosing parameter τ for network decomposition is the problem of determining a time constant for closed-loop dynamics.

For large values of τ, the effects of each input traverse on the network from the outputs close to the input to the outputs far from the input, and hence it becomes likely to pair inputs and outputs which are distant, resulting in intertwined input-output paths. Also, the computational burden is large for controllers targeting at longer time response. For small values of τ, the subsystems generated are of smaller sizes and well disjoint. However, since small τ dictates a fast evolution of closed-loop system, the manipulated input must take greedily large values, which may exceed practical constraints, invalidate the linearization of the system, or cause large deviations in the longer time scale that triggers instability.

Hence a trade-off in the choice of τ becomes important. Strictly speaking, this should be a trial-and-error procedure involving the network decomposition by $\Lambda(1/\tau)$ and controller design and simulation. Here we propose to choose the parameter τ according to the following guidelines:

1. The decomposition by $\Lambda(1/\tau')$ is identical to that generated by $\Lambda(1/\tau)$ for any τ' in a neighborhood of τ that is not too small. This guarantees the robustness of the decomposition with respect to the time scale parameter.

2. The decomposition by $\Lambda(1/\tau)$ generate input-output groups that are well separated in the network, in the sense that the inputs and outputs in different subsystems can interact through paths that do not intersect.

3. For the control of a process system, a meaningful time constant of the closed-loop dynamics should be minutes to hours.

4. The closed-loop time constant should not be too short so that the manipulated input values are constrained within reasonable ranges, and the closed-loop system is stable.

5. For model predictive controllers, the receding horizon should not be too long so that the size of the optimization problem can be efficiently solved with a robust solution.

5.4 Case Study

We use a reactor-separator process (see Fig. 4.1), which has been widely used for developing distributed MPC algorithms [234, 397, 421], to examine the quality of the decomposition generated in the RTAGA framework proposed in this work. The process has two CSTRs in series, where reactions A$\xrightarrow{1}$B$\xrightarrow{2}$C take place, followed by a separator which produces a vapor stream recycled and a liquid stream withdrawn as the product. Cold reactant A is fed into both reactors. B is the product of interest. The governing equations of the system can be found in Appendix A. The parameters and operation point is the same as in Table A.1. F denotes flow rate, x mole fraction, V volumetric holdup, T temperature, Q heat exchange rate, k^0 pre-exponential factor of rate constants, E activation energy, ΔH_r heat of reaction, ρ density, C_p heat capacity, α relative volatility and m molality. The inputs include flow rates F_{f1}, F_{f2}, F_1, F_2, F_3, F_R and heat transfer rates Q_1, Q_2, Q_3. The outputs include liquid levels V_1, V_2, V_3, temperatures T_1, T_2, T_3 and desirable product concentrations x_{B1}, x_{B2}, x_{B3} in three units.

5.4.1 Network decomposition

By linearizing the system model, $G(1/\tau)$ and $\Lambda(1/\tau)$ can be calculated. The network decomposition is performed for different time scale parameters τ. Fig. 5.1 shows the change of the number of communities and corresponding modularity obtained by community detection for different time scales. According to the guideline (i), we see that the number of communities is relatively robust on 3 intervals of τ, namely when $0 < \tau < 0.5$ h, 0.6 h $< \tau <$ 1.5 h and 10 h $< \tau <$ 40 h. The decomposition for these three cases are given in Table 5.1.

Within these decompositions, the first and the second decompositions guarantee that the input-output subsystems are separated. For the decomposition obtained for τ exceeding 10 h, Q_1 (Q_2 and Q_3) is paired with T_3 (T_1 and T_2, correspondingly). However, the only state that input Q_1 (Q_2, Q_3) directly affects is T_1 (T_2, T_3), which implies that the input-output paths of different subsystems intersect. Therefore, according to the guideline (ii), the value of τ must be within the range that generates the first and second decompositions in Table 5.1. Also, the guideline (iii) excludes the decomposition for τ longer than 10 hours, and allows only the first and the second decompositions. Easy to see that these two decompositions are different only in that the subsystems II and III in the first decomposition are merged in the second decomposition. Here we take the 8-subsystem and 7-subsystem decompositions as candidates to be further examined.

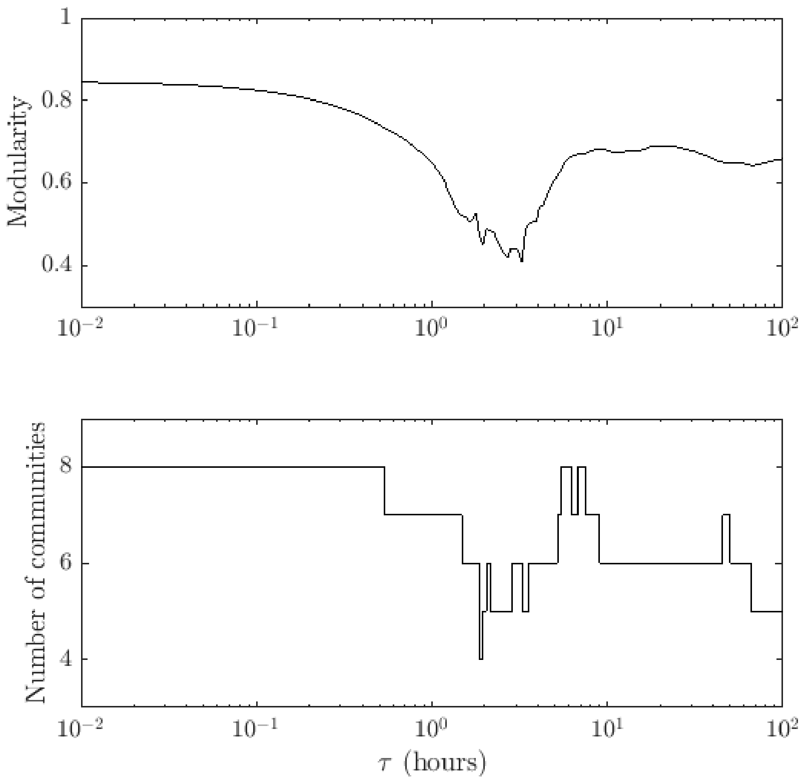


Figure 5.1: Community number and modularity for different time scales.

5.4.2 Distributed MPC simulation

We follow the method of [326, 327] to compare different network decompositions by the resulting control performance and computational time using a noncooperative and iterative distributed MPC scheme [232]. Each controller determines the local manipulated inputs by minimizing locally tailored objective function associated with the predicted future trajectory. At each iteration, the minimizations are performed in parallel using the optimization results obtained at the previous iteration of all controllers (see Fig. 5.2). Specifically, we write the underlying (nonlinear) dynamics in the form of

$$\dot{x} = f(x) + g(x)u, \quad y = h(x). \tag{5.42}$$

We use a backward discretization for the differential equations $\dot{x} = f(x) + g(x)u$, so that the x_l can be expressed explicitly in terms of u_{l-1} and x_{l-1} for $l = k+1, \dots, k+N$. At

Table 5.1: Decompositions in different time scales

τ	Subsyst.	Inputs	Outputs
0–0.5 h	I	F_{f1}, F_R	V_1, x_{B1}
	II	F_{f2}	x_{B2}
	III	F_1	V_2
	IV	F_2	x_{B3}
	V	F_3	V_3
	VI	Q_1	T_1
	VII	Q_2	T_2
	VIII	Q_3	T_3
0.6–1.5 h	I	F_{f1}, F_R	V_1, x_{B1}
	II	F_{f2}, F_1	V_2, x_{B2}
	III	F_2	x_{B3}
	IV	F_3	V_3
	V	Q_1	T_1
	VI	Q_2	T_2
	VII	Q_3	T_3
10–40 h	I	F_{f1}, F_2, F_R	V_1, x_{B1}, x_{B3}
	II	F_{f2}, F_1	V_2, x_{B2}
	III	F_3	V_3
	IV	Q_1	T_3
	V	Q_2	T_1
	VI	Q_3	T_2

time t_k, the i-th controller solves the optimization problem at its c-th iteration:

$$\begin{aligned} \min_{\hat{u}_k^{[i,c]},\dots,\hat{u}_{k+N-1}^{[i,c]}} \quad & J_{k,i} = \sum_{l=k+1}^{k+N} \left[\hat{y}_l^\top Q^{[i]} \hat{y}_l + (\hat{u}_{l-1}^{[i,c]})^\top R^{[i]} \hat{u}_{l-1}^{[i,c]} \right] \\ \text{s.t.} \quad & \frac{\hat{x}_l - \hat{x}_{l-1}}{\Delta t} = f(\hat{x}_{l-1}) + g^{[i]}(\hat{x}_{l-1})\hat{u}_{l-1}^{[i,c]} + \sum_{j \in N(i)} g^{[j]}(\hat{x}_{l-1})\hat{u}_{l-1}^{[j,c-1]}, \\ & \hat{y}_l = h(\hat{x}_l), \;\; \hat{x}_k^{[i]} = x_k^{[i]}, \;\; \underline{u}^{[i]} \le \hat{u}_l^{[i,c]} \le \overline{u}^{[i]}, \\ & \phi^{[i]}(\hat{x}_l, \hat{u}_l^{[i,c]}) \le 0, \;\; \psi^{[i]}(\hat{x}_l, \hat{u}_l^{[i,c]}) = 0, \\ & l = k+1, \dots, k+N \end{aligned} \tag{5.43}$$

The sampling time and receding horizon, as well as diagonal weighting matrices for inputs and outputs, are set in the same values as described in Appendix A. We set the

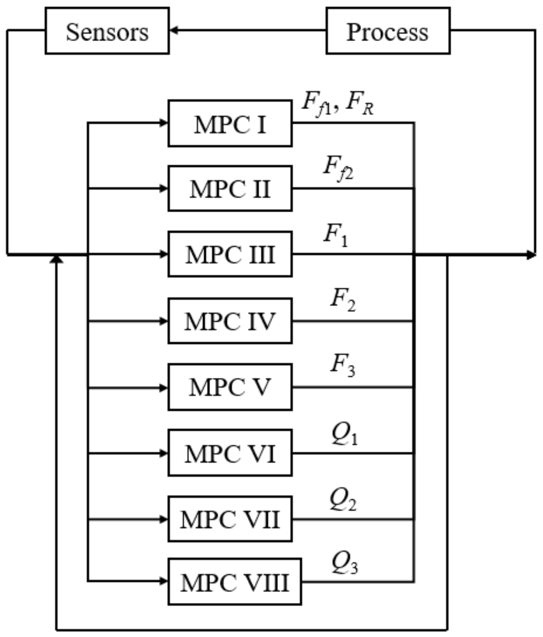


Figure 5.2: Iterative distributed MPC scheme.

initial state values as given in Table A.2. The simulation is performed in Matlab 2015b using a 3.4 GHz Intel® Core™ i7-6700 processor. Fig. 5.3 shows the evolution of the inputs and outputs within 2 hours. For brevity we only plot for the second decomposition here. It is seen that the state regulation is almost accomplished in 1.0–1.5 hours.

We note that our analysis is based on a single time scale parameter τ. However, the closed-loop system does not generally evolve exponentially. In other words, there is no uniform time constant for the closed-loop system. Nevertheless, we can assume that there is a distribution of time scale τ. In our simulation study, we can roughly estimate that the closed-loop system time constant is distributed between 0.1–1.0 hours. Within this distribution, the RTAGA gives the first two decompositions. This further corroborates that the first two decompositions are correctly chosen.

5.4.3 Comparative studies

To show the advantage of the proposed method for distributed control architecture design, we compare the different decompositions generated by this work and previous ones

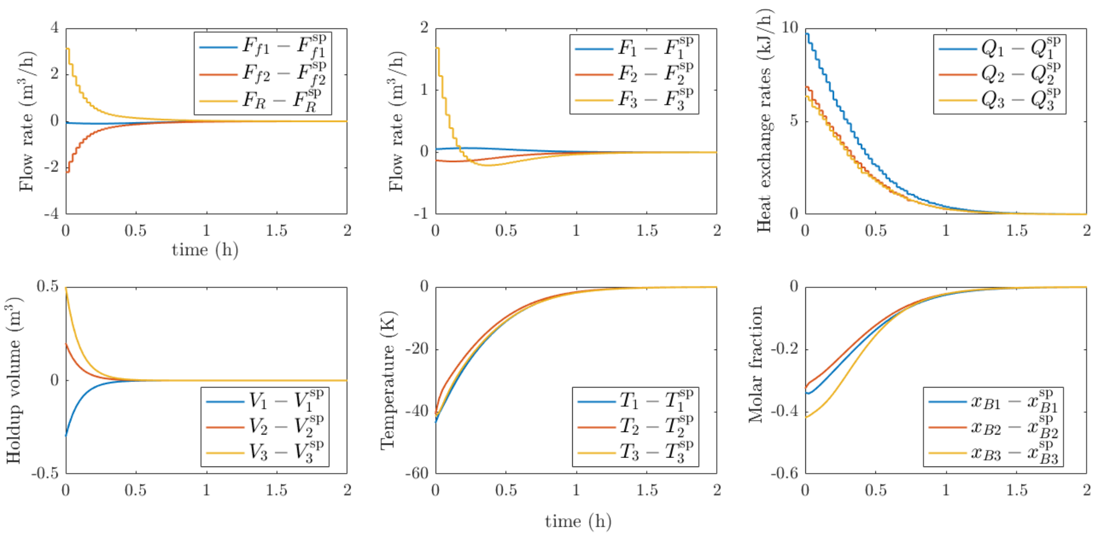


Figure 5.3: Simulation for the distributed model predictive control of the reactor-separator process (according to the second decomposition). Subscript "sp" stands for specified steady state values.

(see Table 5.2), together with the centralized control. Decomposition 1 and decomposition 2 are the previously mentioned ones obtained by the method in this work. Decomposition 3 is obtained by community detection in the input-output bipartite graph, in which the edge weight is defined based on a short-time response measure incorporating relative degrees and sensitivities (see [408]). Decompositions 4 and 5 are given by direct community detection in the system digraph which captures the interactions between inputs, states and outputs with the information of sensitivities neglected (see Jogwar and Daoutidis [180]). Specifically, decomposition 3 is the optimal decomposition suggested by the system digraph-based method, and decomposition 4 is a suboptimal one with a modularity value close to that of decomposition 5. The criteria that we use to compare the decompositions are the performance index (PI)

$$\mathrm{PI} = \sum_{k=0}^{\infty} \left(y_k^\top P y_k + u_k^\top Q u_k \right) \tag{5.44}$$

and the computational time required to repeatedly solve the control-relevant optimization problems within a simulation time of 120 minutes.

The following observations can be drawn from the Table 5.3:

Table 5.2: Decompositions obtained by different methods

Decomposition	Subsyst.	Inputs	Outputs
1 (this work)	I	F_{f1}, F_R	V_1, x_{B1}
	II	F_{f2}	x_{B2}
	III	F_1	V_2
	IV	F_2	x_{B3}
	V	F_3	V_3
	VI	Q_1	T_1
	VII	Q_2	T_2
	VIII	Q_3	T_3
2 (this work)	I	F_{f1}, F_R	V_1, x_{B1}
	II	F_{f2}, F_1	V_2, x_{B2}
	III	F_2	x_{B3}
	IV	F_3	V_3
	V	Q_1	T_1
	VI	Q_2	T_2
	VII	Q_3	T_3
3	I	F_{f1}, Q_1	T_1, x_{B1}
	II	F_{f2}, Q_2, Q_3	T_2, T_3, x_{B2}
	III	F_1	V_1, V_2
	IV	F_2, F_3, F_R	V_3, x_{B3}
4	I	F_{f1}, F_R, Q_1	V_1, T_1, x_{B1}
	II	F_{f2}, F_1, Q_2	V_2, T_2, x_{B2}
	III	F_2, F_3, Q_3	V_3, T_3, x_{B3}
5	I	F_{f1}, Q_1	V_1, T_1, x_{B1}
	II	F_{f2}, F_1, Q_2	V_2, T_2, x_{B2}
	III	F_2, F_3, F_R	V_3, x_{B3}
	IV	Q_3	T_3

1. The centralized architecture gives good control performance (if the optimization problems can be immediately solved), while on the other hand demands a calculation time much longer than all the distributed control architectures. Therefore, the centralized control is generally not preferred and can become impractical for larger systems.

2. The decompositions 4 and 5 greatly reduce the computational time to nearly 20% of the computational time required by centralized control by partitioning the entire system into several subsystems. The optimal decomposition (decomposition

Table 5.3: Dependence of control performance on network decompositions

Decomposition	PI (10^4)	Time (min)	Quality
1	1.171	1.79	6.35
2	1.014	1.99	6.60
3	0.657	3.85	5.26
4	0.611	4.67	4.67
5	1.182	5.23	2.15
Centralized	0.560	23.78	1

4), with maximized modularity in the system digraph, gives relatively better performance than the suboptimal one (decomposition 5) since the inter-subsystem interactions are minimized.

3. For decomposition 3, the computational time is further reduced, with a performance index very close to that of a centralized control. The reason for this good performance is that the decomposition is performed accounting for the intensity of input-output interactions, and hence the optimization subproblems involve inputs and outputs that are more relevant. As can be seen from the Table 5.2, the inputs and outputs within the same subsystems in decomposition 2 are physically associated with the governing law of mass balances and energy balances.

4. For decompositions 1 and 2, since most of the subsystems in this case has become single-input-single-output, resulting in optimization problems involving only one decision variable, the computational time is further reduced to approximately 8% of the centralized control, which are the shortest among all decomposition methods so far. On the other hand, as the partitioning becomes finer, although the difference between intra-subsystem and inter-subsystem interactions are large, the interactions between subsystems are more significant than coarser decompositions (e.g., decomposition 3), resulting in a larger PI.

In summary, the decomposition given by the method in our previous study [408, 180] (decompositions 3 and 4) gives good performance as well as a reduced computational time, and that the proposed method of this work (decompositions 1 and 2) further reduces the computational time with some performance loss. The preferred control architecture should be determined with a trade-off between these two aspects, which is dependent mostly on the computational capability and the algorithm. With faster computation, we prefer the coarser decompositions to minimizing the interactions for better performance index, while with more limited computational capability, finer decompositions should be suggested in order to avoid the cost resulted from the computational delay. For the present, we define a comprehensive quality measure of decompositions based on the product of PI and computation time,

$$\text{Quality} = \frac{(\text{PI} \times \text{Time})_{\text{centralized}}}{\text{PI} \times \text{Time}}. \tag{5.45}$$

We found that the comprehensive performances of the first two decompositions are similar (with the performance of decomposition 2 slightly better) and exceed the other decompositions.

5.5 Conclusion and Discussion

In this work, we use the relative time-averaged gain array (RTAGA) at time scale τ, $\Lambda(1/\tau)$, as an input-output interaction measure, which offers a middle ground between classical relative gain analysis ($\tau \to \infty$) and relative degree and sensitivity analysis ($\tau \to 0$). A systematic method of network decomposition minimizing the inter-subsystem interactions within time scale τ is proposed, which applies bipartite community detection in the modified $\Lambda(1/\tau)$. The time scale parameter is chosen following the guidelines including the connectivity of the subnetworks, robustness of decomposition, constraints on the manipulated inputs, tractability of associated optimization problems, and typical range of time scales of control problems. Through application to a reactor-separator benchmark process, we demonstrated the effectiveness of our proposed method of control architecture design, highlighting a fine decomposition with a significant reduction of computational time.

So far, the network decomposition approaches and distributed control strategies are tested only on benchmark reactor-separator process. On-going research is focusing on large-scale complex networks, for which the decomposition should involve state variables beyond input-output partitioning, so that each controller make use of only local models instead of the complete models. It is also of importance to take into consideration the external disturbances. The decomposition methods applicable to the systems containing disturbances need to be developed and the operability or the disturbance rejection performance of the closed-loop system need to be examined.

Chapter 6
Optimal Decomposition for Distributed Optimization in Nonlinear MPC by Community Detection

6.1 Introduction

Large-scale and complex systems are becoming the rule rather than the exception in process systems engineering problems [135, 18]. The design and operation of these systems involve large-size, non-convex, mixed-integer, multi-scale and uncertain optimization models [113, 270, 250, 136, 122] which are computationally expensive to solve. Decomposition is a major approach to solving these large-scale optimization problems. To deal with the complicating variables and constraints between different groups of unit operations, Benders and Lagrangian decompositions have been recognized as the most useful decomposition methods [66, 136]. Computational complexity is also a major issue in the implementation of many advanced closed-loop decision making schemes due to the requirement of real-time repeated solution of optimization problems. Examples include nonlinear model predictive control (NMPC) [344, 254], economic MPC (EMPC) [343, 98], closed-loop scheduling [142, 317] as well as their integration [19, 322, 86].

NMPC is an advanced control strategy where the manipulated inputs of a nonlinear process are determined by minimizing a cost associated with the predicted trajectory of the system based on real-time measurements. During the recent decade, highly efficient nonlinear programming algorithms, toolboxes (e.g., IPOPT [437, 37] and GPOPS [340]) and modelling platforms (e.g., CasADi [8], JuMP [93] and Pyomo [147]) have been developed to enable fast implementation of NMPC. Under interior-point optimization algorithms, a decomposition at the linear algebra level can be exploited for further acceleration by parallelizing the solution of the Karush-Kuhn-Tucker (KKT) conditions, if there exists an approximate block diagonal [67], bordered block diagonal [477, 187], or block tridiagonal [440] structure in the KKT matrix. Other techniques such as truncated Newtonian optimization [87, 474] and advanced-step optimization [476, 461] have also been proposed for the fast solution of optimal control problems.

©2018 Elsevier. Reprinted, with permission, from W. Tang, A. Allman, D. Babaei Pourkargar, P. Daoutidis. Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection. *Computers & Chemical Engineering* (Volume 111, March 2018). `DOI: 10.1016/j.compchemeng.2017.12.010`.

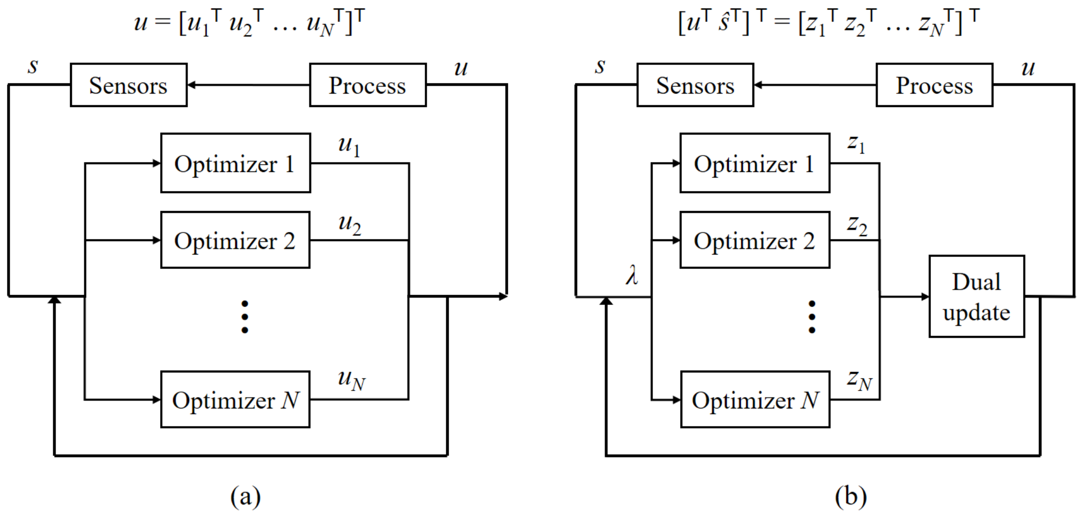


Figure 6.1: Two formulations and distributed optimization methods of the optimal control problem in NMPC. (a) BCD algorithms for the "input problem" formulation. (b) ADMM algorithms for the "input-state problem" formulation. u, s, $\hat{s}$, z, and λ stand for inputs, states, predicted states, primal variables and dual variables, respectively.

6.1.1 Distributed optimization for NMPC

Distributed optimization formulations of the associated optimal control problems, referred to as distributed model predictive control, have been a major focus recently (see [362, 63, 288] for reviews and references) due to fewer communication requirements, resilience to faults, and the potential of further improving the computational performance. So far, two types of distributed optimization algorithms – block coordinate descent (BCD) algorithms [452, 285] and multi-block alternative direction method of multipliers (ADMM) [44, 229, 162], have been applied in distributed NMPC.

In most of these works, the optimal control problem is formulated such that the manipulated inputs are the variables to be optimized, and the state and output variables are explicitly expressed in terms of inputs (see e.g., [234]). We refer to this as an "*input problem*" formulation. By BCD (see Fig. 6.1(a)), the input variables in the system are assigned to different controllers, and each controller solves an optimization problem for its own inputs. The controllers exchange their local solutions in a sequential, iterative or neighbor-to-neighbor scheme. We note that the BCD algorithms under an "input problem" formulation, in the presence of complex constraints, can cause the solution trajectory to escape from the feasible set if the optimization subproblems are solved in parallel, unless the feasible set is box-shaped, i.e., the constraints on the manipulated inputs include only individual bounds. Further, when the objective function is a complex function of the manipulated inputs, it becomes difficult to guarantee the optimality of the solution obtained.

In a different vein, the optimal control problem can also be formulated such that the inputs and the predicted states are both variables to be optimized, and these variables are (nonlinearly) constrained by the system dynamical equations (see e.g., [476]). In this way, the objective function is expressed in a simpler form (usually a quadratic form in terms of the inputs and the states) while the constraints become more complex. We refer to this as an "*input-state problem*" formulation. By ADMM (see Fig. 6.1(b)), an augmented Lagrangian is constructed to deal with the complex constraints, and the primal variables, namely the variables to be optimized, are updated through optimization in groups, with the dual variables updated after the optimization of primal variables [332, 166, 167]. The Lagrangian-based algorithms under an "input-state problem" formulation can potentially avoid the limitations of the "input problem" formulation. Through a dualization of the constraints, the solution trajectory is no longer confined inside the feasible set. With a simpler form of the objective function, the optimality of the solution obtained can be improved. On the other hand, the "input-state problem" involves a higher-dimensional optimization, and thus may generally require a longer computational time than the "input problem".

6.1.2 Choice of optimal decomposition

To implement any distributed control or optimization strategy, a decomposition, from which good control performance as well as reduced computational time can be expected, must be predetermined. The topic of system decomposition has been studied in different contexts for a long time. For dynamic systems and control systems, decompositions were proposed for decentralized control design [434, 321, 372]. In process system engineering, algorithms for tearing process flowsheets were proposed for process simulation and synthesis (see e.g., [24, 320, 426]). The decomposition of systems of equations was studied even earlier (see e.g., [92, 396, 156]). The key of these decomposition methods is to detect the underlying structures in the system – most importantly, blocks and hierarchies – with graph-theoretic representations and tools.

Network science has recently emerged as a promising framework for the analysis of complex systems based on macroscopic and statistical properties of graphs (networks)

[21]. In this setting, *community detection* has become a major field of study. Different from the traditional decomposition methods that seek a strict block diagonal or block triangular structure using paths, cycles or coverings, community detection aims to find subnetworks with significantly more links between the nodes inside than across them [127]. The decomposition can be efficiently generated for large and complex networks, and is optimal in the sense of the difference between inter- and intra-community interconnections, which is desirable for distributed control and optimization in networks. Details of this approach will be introduced later in Section 6.2.

Recently in our research, we have developed some methods of input-ouput partitioning for distributed control [180, 406, 408]. These decomposition methods, based on the analyses of the interactions among the process variables (inputs, states and outputs), generate compact and weakly interacting subsystems through community detection. It was shown that the choice of decomposition is indeed a major factor affecting the control performance and computational time of the resulting control scheme, and that the adoption of appropriate decompositions can relieve the computational burden while maintaining good control performance [326, 327]. However, these input-output partitioning methods were not specifically designed for NMPC applications, and were applicable only to "input problem" formulations. To find an optimal decomposition for the "input-state problem" formulation, we need to address the problem of *optimally decomposing the optimization problems.*

So far, decomposition for distributed optimization is typically done by intuitively finding such complicating variables and complicating constraints that the optimization problem becomes completely separated once these complicating variables or constraints are removed. For general problems without a special structure, there has been no systematic method for finding an optimal decomposition for distributed optimization.

In this work, we introduce network representations of a broad class of optimization problems, and propose an efficient framework of determining the optimal decomposition of these optimization problems based on community detection in these network representations. This framework is based on the structure of the optimization problem rather than on the specific parameters (namely the measurements at different sampling times). It can therefore be applied offline to generate a decomposition in advance of the online implementation of NMPC.

The remainder of this chapter is organized as follows. In Section 6.2, we first introduce the notion of variable-constraint bipartite network, variable unipartite network, and constraint unipartite network as structural representations of optimization problems with an objective function in separable form. Through the study of a large-scale convex optimization problem, we show the computational benefit of decomposing the network into community structures which are strongly interconnected inside but weakly interconnected between. Community detection algorithms as powerful tools in network science for finding modular structures are subsequently introduced. In Section 6.3, the network community detection framework is applied to generate the optimal decomposition of the optimal control problems in NMPC. In Section 6.4, we examine a benchmark reactor-separator process with comparisons of different decompositions under an ADMM algorithm slightly modified from that of [166] and the IPOPT algorithm. We conclude that the community detection in the variable unipartite network and the variable-constraint bipartite network result in the optimal decompositions, respectively.

6.2 Community Detection for the Decomposition of Optimization problems

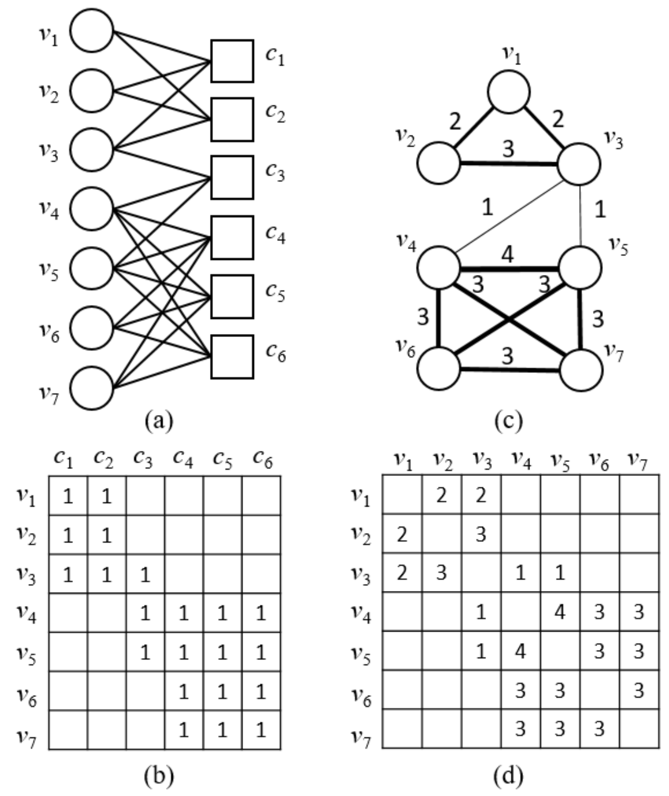
6.2.1 Network representations of optimization problems

We consider without loss of generality a constrained optimization problem in the following "separable" form:

$$\begin{aligned} \min \quad & f_1(v_1) + f_2(v_2) + \cdots + f_n(v_n) \\ \text{s.t.} \quad & c_j(v_1, v_2, \ldots, v_n) = 0, \;\; j = 1, 2, \ldots, m \\ & v_i \in \mathcal{V}_i, \;\; i = 1, 2, \ldots, n \end{aligned} \tag{6.1}$$

where $v_i, i = 1, 2, \ldots, n$ are scalar variables, and $\mathcal{V}_i, i = 1, 2, \ldots, n$ represents the set (interval) to which the variables belong. Note that in (6.1), the objective function is separable and the variables are coupled only by equality constraints (with every inequality constraint only acting upon a single variable). If a non-separable objective function or coupling inequality constraints are present, the original optimization problem can always be converted to the form of (6.1) by defining suitable new auxiliary variables and constraints. To decompose the optimization problem into a distributed architecture, we seek to exploit the inherent structure of the optimization formulation, i.e., the *interactions* between the variables $v_1, v_2, \ldots, v_n$ and the constraints $c_1, c_2, \ldots, c_m$.

The structural information of the variable-constraint interactions can be represented



	c_1	c_2	c_3	c_4	c_5	c_6
v_1	1	1				
v_2	1	1				
v_3	1	1	1			
v_4			1	1	1	1
v_5			1	1	1	1
v_6				1	1	1
v_7				1	1	1

	v_1	v_2	v_3	v_4	v_5	v_6	v_7
v_1		2	2				
v_2	2		3				
v_3	2	3		1	1		
v_4			1		4	3	3
v_5			1	4		3	3
v_6				3	3		3
v_7				3	3	3	

Figure 6.2: Network representations of optimization problems. (a) Variable-constraint bipartite network. (b) Bipartite adjacency matrix. (c) Variable unipartite network. (d) Edge weight matrix of the variable unipartite network.

by constructing a *variable-constraint bipartite network (graph)* (see Fig. 6.2(a)). In this bipartite network, the nodes are classified into two sets – the set of variables $\{v_1, v_2, \ldots, v_n\}$ and the set of constraints $\{c_1, c_2, \ldots, c_m\}$, and every edge connects a variable node and a constraint node: if the variable v_i is present in the expression of c_j (i.e., $\partial c_j / \partial v_i \not\equiv 0$, assuming the continuous differentiability of all the constraints), then there is an edge between v_i and c_j. The network information can be equivalently expressed by a bipartite adjacency matrix $\mathbf{B} \in \{0,1\}^{n \times m}$, where its (i,j)-th element $b_{ij} = 1$ if an edge exists between v_i and c_j, and $b_{ij} = 0$ otherwise (see Fig. 6.2(b)). The bipartite adjacency matrix $\mathbf{B}$ is simply the transposed Jacobian matrix $[\partial c_j / \partial v_i]$, with every nonzero element substituted by 1.

The information in the bipartite network can be simplified into a *variable unipartite network* which captures the interactions between the variables (see Fig. 6.2(c)). The nodes in the variable unipartite graph include only the variables $v_1, v_2, \dots, v_n$. The weight of the edge between v_i and $v_{i'}$, $u_{ii'}$, is defined as the number of the constraints to which both v_i and $v_{i'}$ are adjacent in the variable-constraint bipartite network, i.e.,

$$u_{ii'} = \sum_{j=1}^{m} b_{ij} b_{i'j} = \left(\mathbf{B}\mathbf{B}^\top\right)_{ii'} . \tag{6.2}$$

The variable unipartite network can also be represented by the edge weight matrix $\mathbf{U} = [u_{ii'}] \in \mathbb{N}^{n\times n}$ (see Fig. 6.2(d)). In graph theory, the variable unipartite network is said to be the one-mode projection of the variable-constraint network onto the variable set [487]. Compared to the bipartite network, for each pair of variables, the unipartite network only tells *how many* constraints they share, without providing information on *which* constraints they share.

We can also define the *constraint unipartite network* to capture the interactions between the constraints in the same way as for the unipartite unipartite network. The corresponding edge weight matrix $\mathbf{W} = [w_{jj'}] \in \mathbb{N}^{m\times m}$ is obtained by

$$w_{jj'} = \sum_{i=1}^{n} b_{ij} b_{ij'} = \left(\mathbf{B}^\top\mathbf{B}\right)_{jj'} . \tag{6.3}$$

To decompose the optimization problem, we propose to partition the variables (constraints) into groups such that the interactions in the same groups, i.e., the common involvement in the constraints (variables) between pairs of variables (constraints) in the same groups, are stronger than the interactions between a pair of variables (constraints) across different groups. In network theory terminology, such groups are called *community structures*, and the procedure to reveal the communities in networks is called *community detection* [127]. Community structures are widely present in biological, social and technological networks, and community detection has already become a major area of study in network science [114, 294, 115]. Illustrations of community structures in bipartite and unipartite networks are given in Fig. 6.3(a) and (b), respectively.

Community structures can not usually be determined intuitively, and therefore efficient community detection algorithms should be adopted. Before introducing such algorithms, we consider a typical large-scale convex ℓ_1-norm minimization problem with highly coupling equality constraints to show how the different decompositions of the

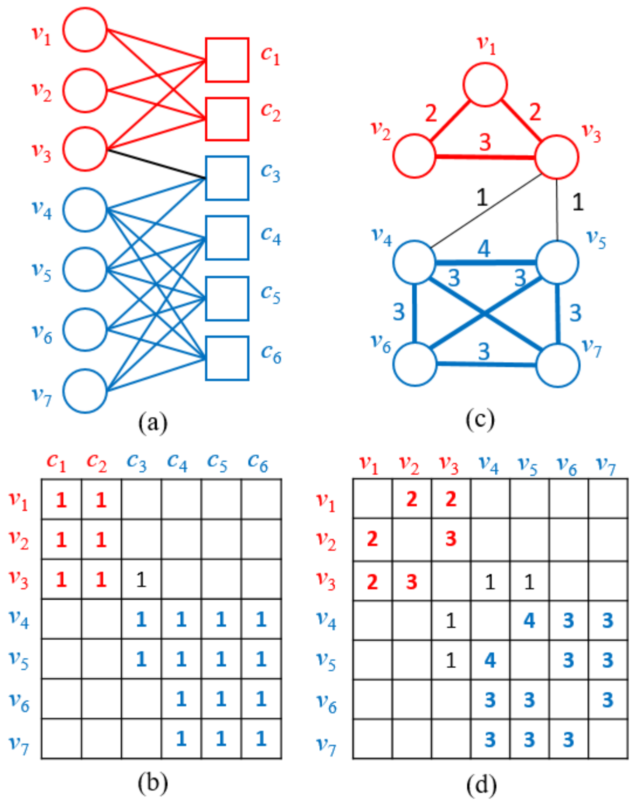


Figure 6.3: Communities in variable-constraint bipartite network (a,b) and variable unipartite network (c,d). The edge weights between communities formed by v_1, v_2, v_3 (red) and v_4, v_5, v_6, v_7 (blue) is smaller than the edge weights inside the communities. The communities are represented in both the network topology (a,c) and the adjacency matrix (b,d).

same optimization problem make a difference in the solution, in the presence of communities.

6.2.2 The role of community structures in the decomposition of optimization problems

We consider a quadratic minimization subject to linear constraints, also known as the "basis pursuit problem" [44]:

$$\min_{v} \quad \|v\|_1 = \sum_{i=1}^{n} |v_i| \quad \text{s.t.} \quad Av = b \tag{6.4}$$

where $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^{n\times 1}$. This model can be taken as an optimization problem associated with the unconstrained MPC of a linear system if the cost function assumes the form of an ℓ_1-norm. We solve the optimization problem in a distributed manner by dividing the variables into groups $v^1, v^2, \dots, v^N$, $v^i = [v^i_1, v^i_2, \dots, v^i_{n_i}]^\top$, $i = 1, 2, \dots, N$, and using the parallel multi-block ADMM method [44]. Specifically, an augmented Lagrangian is constructed, in which λ is the Lagrange multiplier, γ is a penalty coefficient enforcing the constraints, and A^i is composed of the columns of A corresponding to variable block v^i:

$$\begin{aligned} L_\gamma(v^1, v^2, \dots, v^N; \lambda) &= \|v\|_1 - \lambda^\top (Av - b) + \frac{\gamma}{2}\|Av - b\|_2^2 \\ &= \sum_{i=1}^N \|v^i\|_1 - \lambda^\top \left(\sum_{i=1}^N A^i v^i - b\right) + \frac{\gamma}{2}\left\|\sum_{i=1}^N A^i v^i - b\right\|_2^2 . \end{aligned} \tag{6.5}$$

The minimizations are performed on primal variables in parallel blocks, followed by a dual variable update in each iteration:

$$\begin{aligned} v^1 &:= \arg\min_{z^1} L_\gamma(z^1, v^2, \dots, v^N; \lambda) \\ v^2 &:= \arg\min_{z^2} L_\gamma(v^1, z^2, \dots, v^N; \lambda) \\ &\cdots \\ v^N &:= \arg\min_{z^N} L_\gamma(v^1, v^2, \dots, z^N; \lambda) \\ \lambda &:= \lambda - \gamma\left(\sum_{i=1}^N A^i v^i - b\right) \end{aligned} \tag{6.6}$$

To examine the effect of decompositions on the computational performance, we generate a matrix A of size 1000×600 with nonzero elements more densely located within 4 separate blocks, which is visualized in Fig. 6.4. These 4 blocks which appear almost yellow in Fig. 6.4 correspond to 4 communities of variables. The vector b is generated with the same procedure as in [229] by $b = Av^* + w$ where v^* has 50 nonzero components randomly generated in $[-1, 1]$, and w is a random noise vector with identically distributed Gaussian components, each with a zero mean and a standard deviation of 10^{-3}.

We consider five different decompositions of the variables: (i) Decomposition according to the community structures, i.e., $v^1 = v_{1:200}$, $v^2 = v_{201:400}$, $v^3 = v_{401:650}$,

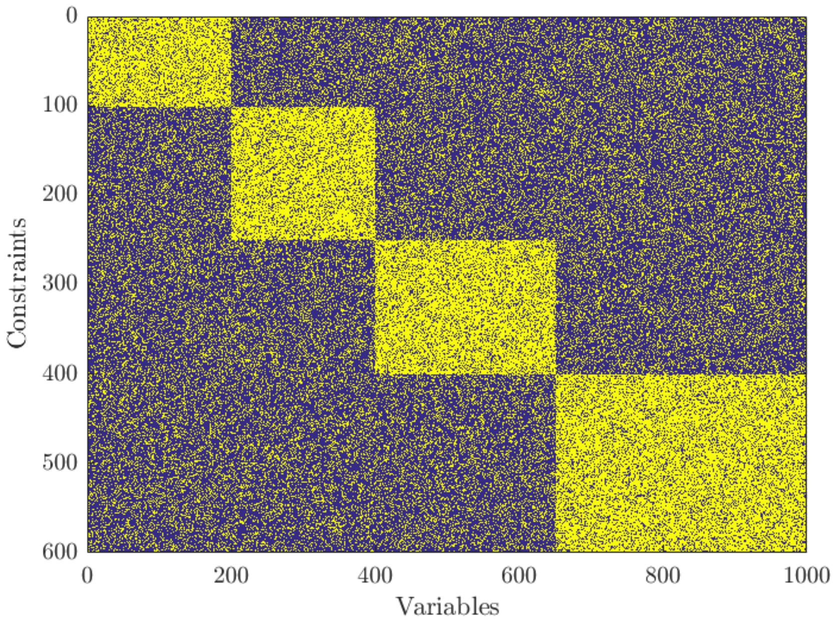


Figure 6.4: Visualization of the constraint matrix A in the ℓ_1-norm minimization problem. Each nonzero element is represented a yellow pixel and a zero element by a blue pixel.

$v^4 = v_{651:1000}$. (ii) Decomposition with equal number of variables $v^1 = v_{1:250}$, $v^2 = v_{251:500}$, $v^3 = v_{501:750}$, $v^4 = v_{751:1000}$. (iii) Decomposition in randomized four groups of indices, with the size of the groups equal to those of the communities. (iv) A decomposition coarser than (i): $v^1 = v_{1:400}$, $v^2 = v_{401:1000}$. (v) A decomposition finer than (i): $v^1 = v_{1:200}$, $v^2 = v_{201:400}$, $v^3 = v_{401:525}$, $v^4 = v_{526:650}$, $v^5 = v_{651:825}$, $v^6 = v_{826:1000}$. As the iterations proceed, we record the relative error defined by $\|v - v_{\text{opt}}\|_2/\|v_{\text{opt}}\|_2$, in which the v_{opt} is the true optimal solution obtained by directly solving the centralized optimization problem. The variations of the relative error with increasing iteration numbers are shown in Fig. 6.5 for four decompositions, except for the randomized decomposition which does not result in a solution trajectory convergent to the optimal solution v^* with increasing iterations.

Remark 6.1. *For the constrained ℓ_1-norm minimization problem, each block minimization step in* (6.6) *is performed in the following way. By defining an auxiliary variable $\epsilon \in \mathbb{R}^m$,*

$$A^i v^i + \sum_{j \neq i} A^j v^j - b + \epsilon = \frac{1}{\gamma}\lambda, \tag{6.7}$$

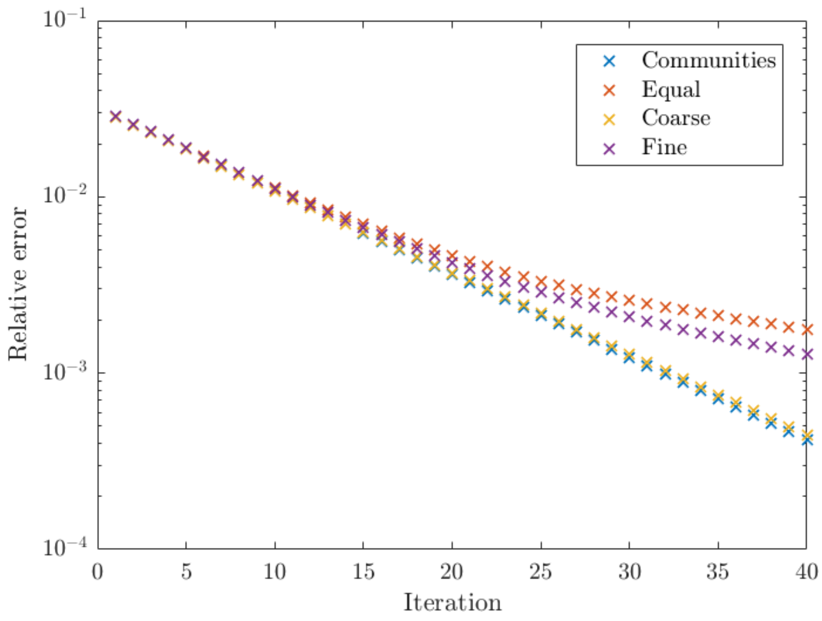


Figure 6.5: Convergence rate of the multi-block ADMM algorithm applied to the ℓ_1-norm minimization problem using different decompositions.

the subproblem of optimizing v^i can be transformed to the following form

$$\min_{v^i,\epsilon} \quad \|v^i\|_1 + \frac{\gamma}{2}\epsilon^\top \epsilon. \tag{6.8}$$

Denote by v^i_+ and v^i_- the positive and the negative part of v^i, respectively, i.e., $v^i_+ = (|v^i| + v^i)/2$ and $v^i_- = (|v^i| - v^i)/2$. Denote by $\mathbf{1}$ the column vector of 1. The v^i optimization step is hence transformed into a quadratic programming problem.

$$\begin{aligned} \min_{v^i_+, v^i_-, \epsilon} \quad & \mathbf{1}^\top v^i_+ + \mathbf{1}^\top v^i_- + \frac{\gamma}{2}\epsilon^\top \epsilon \\ \text{s.t.} \quad & A^i v^i_+ - A^i v^i_- + \epsilon = \frac{1}{\gamma}\lambda + b - \sum_{j \neq i} A^j v^j, \;\; v^i_+ \geq 0, \;\; v^i_- \geq 0. \end{aligned} \tag{6.9}$$

To obtain the true optimal value v^*, the centralized optimization is converted to the following linear programming problem, which can be exactly solved, by defining the positive and negative parts of v, $v_+ = (|v| + v)/2$, $v_- = (|v| - v)/2$.

$$\min_{x_+, x_-} \quad \mathbf{1}^\top v_+ + \mathbf{1}^\top v_- \quad \text{s.t. } Av_+ - Av_- = b, \;\; v_+ \geq 0, \;\; v_- \geq 0. \tag{6.10}$$

Remark 6.2. *We choose this particular example in the form of* (6.4) *for the following reasons: (i) The ℓ_1-norm minimization problem is convex, which has a unique optimum, thus facilitating the comparison. (ii) The problem is not strongly convex, thus allowing distributed ADMM to diverge under some bad decompositions. (iii) This problem is often used in the literature of distributed optimization to examine the performance of algorithms.*

From Fig. 6.5, we can make the following observations:

1. Among the five decompositions above, the randomized decomposition results in divergence, i.e., the distributed optimization can lose its basic performance when the community structure is totally not accounted for in the decomposition. The reason for this is that a random decomposition will generally result in strong interactions between variable groups. Therefore, as long as the problem is not strongly convex (such as the weakly convex ℓ_1-norm minimization), the decomposition for distributed optimization should not be randomly generated.

2. The decomposition of the variables into 4 groups of equal sizes, as a decomposition that possesses a certain degree of similarity to the community structures in the network, leads to convergence.

3. The convergence is accelerated by using a decomposition according to the communities, which renders the difference between cross-group interactions and within-group interactions significant. The coarser decomposition, where each group in the decomposition can contain more communities, has about the same performance as the decomposition exactly based on communities. Thus, *community structures act as the "core" playing a central role in maintaining good performance in the decomposition for distribution optimization.*

4. The finer decomposition, in which each group of variables only contains a portion of the corresponding community, deteriorates the convergence rate. This implies that the convergence performance is correlated to the integrity of the community structures.

Compared to the community-based decomposition, the coarser decompositions merge weakly interacting variable groups (communities) into larger groups. Except for very fast problems, larger subproblems will generally increase the computational burden of each iteration. Since the interactions are already concentrated in the communities, accounting for inter-community interactions does not significantly improve the optimality of the solutions.

The above observations and arguments make a strong case for using a community-based decomposition for distributed optimization. In the following subsection, we introduce the computationally efficient *Louvain algorithm (fast unfolding algorithm)* of community detection [39].

6.2.3 Modularity and Louvain algorithm for community detection

Modularity-based methods have been the mainstream among the community detection algorithms. Modularity-based methods consider the community detection problem as the maximization of a quality function, called *modularity*, defined over all the partitions of the network nodes. Newman-Girvan modularity [296, 292] has been a widely adopted classical form for unipartite networks. Given the adjacency matrix $\mathbf{U} = [u_{ii'}]$ in which $u_{ii'}$ is the weight of the edge between nodes v_i and $v_{i'}$, for any partition of nodes in a unipartite network, $\mathcal{P} = (C_1, C_2, \ldots, C_K)$, where C_i denotes the index of the community to which v_i belongs (e.g., $C_i = k$ means that the i-th node belongs to the k-th community), the corresponding Newman-Girvan modularity is defined as

$$Q(\mathcal{P}) = \sum_{i=1}^{n} \sum_{i'=1}^{n} \left(\frac{u_{ii'}}{u} - \frac{u_i u_{i'}}{u^2} \right) \delta(C_i, C_{i'}). \tag{6.11}$$

In the above expression of modularity, $\delta(C_i, C_{i'}) = 1$ if $C_i = C_{i'}$, i.e., if v_i and $v_{i'}$ belong to the same community and $\delta(C_i, C_{i'}) = 0$ otherwise. $u_i = \sum_{i'=1}^{n} u_{ii'}$ is the degree of node v_i, i.e., the total weight of edges adjacent to v_i. $u = \sum_{i=1}^{n} u_i = \sum_{i=1}^{n} \sum_{i'=1}^{n} u_{ii'}$ is the total edge weight in the network (with each edge counted twice). Hence the term in parentheses in (6.11) captures the difference between the existing fraction of edges between v_i and $v_{i'}$ and the expected fraction of edges between v_i and $v_{i'}$ when all the edge weights are randomly redistributed with the degree of all nodes fixed, thus measuring the level of interaction between v_i and $v_{i'}$ above a statistical average. The modularity $Q(\mathcal{P})$, as the sum of all such measures inside the communities, characterizes the statistical significance of the communities as given by partition $\mathcal{P}$.

The community detection in bipartite networks is therefore formulated as the maximization of modularity $Q(\mathcal{P})$ with respect to all feasible partitions, and the maximized modularity Q is called the modularity of the network, i.e., $Q = \max_{\mathcal{P}} Q(\mathcal{P})$. The global maximization of modularity over all the partitions is an NP-hard problem owing to its combinatorial nature [46]. Due to the computational complexity of the global maximization, many approximation algorithms have been proposed (see [114]). The Louvain algorithm or fast unfolding algorithm [39] has been recognized as the most efficient one with an observed linear scalability. The algorithm involves the following steps until the modularity Q reaches a maximal value.

1. Initialize by assigning every node into a separate community.
2. For each node and each community other than the community to which this node belongs, calculate the modularity increase ΔQ if the node is moved from its original community to this new community.
3. Choose the largest $\Delta Q > 0$ among all nodes and all communities and move the corresponding node into the corresponding community.
4. Repeat the steps 2 and 3 for all nodes until no $\Delta Q > 0$ can be found, i.e., the modularity has reached a local maximal value.
5. Agglomerate each community into a node. Define the edge weights between agglomerated nodes as the sum of the edge weights between the communities from which these two nodes are agglomerated. Define a self-edge for every node with a weight equal to the sum of the edge weights within the community from which it is agglomerated.
6. Repeat steps 2 to 5 until no move further increases the modularity value.

For bipartite networks, the modularity assumes a different form than that of unipartite networks [23] due to the absence of any expected edges between the nodes within the same category (in our case, the variables and the constraints). Given the bipartite adjacency matrix $\mathbf{B} = [b_{ij}]$, where b_{ij} is the weight of the edge between nodes v_i and c_j (for unweighted networks all the weights equal to 1), the modularity of any partition $\mathcal{P} = (C_1, C_2, \ldots, C_K)$ is expressed as

$$Q = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(\frac{b_{ij}}{b} - \frac{b'_i b''_j}{b^2} \right) \delta(C'_i, C''_j) \tag{6.12}$$

in which $b'_i = \sum_{j=1}^{n} b_{ij}$, $b''_j = \sum_{i=1}^{m} b_{ij}$ are the degrees of v_i and c_j, respectively. C'_i and C''_j are the community affiliations of nodes v_i and c_j, respectively. $b = \sum_{i=1}^{m} \sum_{j=1}^{n} b_{ij}$ is the total weight in the network. Similar to the unipartite modularity (6.11), the bipartite modularity (6.12) also characterizes the difference between intra-community interactions and its expected level, and hence the statistical significance of the existence of bipartite communities, each of which contains nodes from both categories. The maximization of bipartite modularity can follow the same steps in the Louvain algorithm as described above.

By applying the community detection algorithm to the bipartite and unipartite networks constructed in Subsection 6.2.1, groups of variables (or groups of constraints) are generated such that the variable (constraint) interactions are weak between different groups but strong within each group. In general, these decompositions need not be identical. Community detection in the bipartite network simultaneously partitions variables and constraints, so that the variable pairs in the same group roughly share a corresponding group of constraints. Community detection in the variable (constraint) unipartite network generates groups of variables (constraints) that share more constraints (variables) *in number*, but the constraints (variables) shared by variables (constraints) in the same group can be highly different and need not form groups.

Remark 6.3. *For distributed optimization, it is desirable to have balanced loads for distributed computing units, i.e., similar sizes for the subsystems generated by the decomposition method. However, load balancing can not be guaranteed by community detection. In the case where the subnetworks are highly heterogeneous in size, we can perform successive community detection (further partition) in the subnetworks that are computationally cumbersome, and agglomerate the subnetworks that are too small.*

6.3 Application to Distributed NMPC

We consider a time-invariant system governed by differential algebraic equations (DAEs):

$$\dot{x} = f_0(x, u), \quad 0 = g(x, z), \tag{6.13}$$

in which $x(t) \in \mathbb{R}^{n_x}$, $z(t) \in \mathbb{R}^{n_z}$ are the vectors of differential state variables and algebraic state variables respectively, and $u(t) \in \mathbb{R}^{n_u}$ denotes manipulated inputs. For simplicity, we only consider nominal models without plant-model mismatch, exogenous disturbances and measurement noise here. The MPC of the system (6.13) requires that at each sampling time the optimization problem (6.14) be solved:

$$
\begin{aligned}
\min\ & J = \sum_{t=0}^{N-1} \Phi(u_{1:n_u}^t, x_{1:n_x}^t) + \Psi(x_{1:n_x}^N) \\
\text{s.t. } & x_{i_x}^t = f_{i_x}(x_{1:n_x}^{t-1}, u_{1:n_u}^{t-1}),\ \ 0 = g_{i_z}(x_{1:n_x}^t, z_{1:n_z}^t), \\
& 0 = r_{i_r}(u_{1:n_u}^t, x_{1:n_x}^t, z_{1:n_z}^t), \\
& u_{i_u} \in [\underline{u}_{i_u}, \overline{u}_{i_u}], x_{i_x} \in [\underline{x}_{i_x}, \overline{x}_{i_x}], z_{i_z} \in [\underline{z}_{i_z}, \overline{z}_{i_z}], \\
& i_u = 1, \ldots, n_u; i_x = 1, \ldots, n_x; \\
& i_z = 1, \ldots, n_z; i_r = 1, \ldots, n_r; t = 1, \ldots, N.
\end{aligned} \tag{6.14}
$$

The variables include the manipulated inputs, differential and algebraic states, and outputs at each discretized time ($u_{i_u}^t$, $x_{i_x}^t$, $z_{i_z}^t$, $y_{i_y}^t$). The constraints include system equations – f_{i_x} (obtained by single step discretization), g_{i_z} (when algebraic equations exist), upper and lower bounds, and any additional coupling constraints (r_{i_r}). The additional constraints are needed for NMPC when, for example, the closed-loop system is required to be stabilized with a prescribed Lyapunov function [234], or when constraints on the manipulated inputs are needed to deal with open-loop instability [327]. Without loss of generality we assume all r_{i_r} are equalities (for inequalities we can appropriately define extra algebraic states with bounds). N represents the number of sampling times in a receding horizon, respectively. The initial states $x_{i_x}^0$ are given by the real-time measurements.

We assume that the stage cost function Φ and the terminal cost function Ψ are separable (e.g., in a classical quadratic form). Then the formulation (6.14) is expressed in the form of (6.1). An illustration of the variable-constraint bipartite network representing the structure of (6.14) is shown in Fig. 6.6 for a simple case. It shows a recurrent pattern throughout the time axis as long as the system is time-invariant, where the algebraic constraints connect variables with the same time index, and the differential constraints connect variables with the neighboring time indices.

In our previous papers [180, 406, 408], the distributed control architecture was generated based on interaction analysis in the system digraph, which is a directed network representation of the relations between process variables in the system. Hence the decomposition was based on process variables, e.g., if u_1 and u_2 are assigned into subsystems 1 and 2 respectively, then the optimization variables $u_1^{0:N-1}$ and $u_2^{0:N-1}$ have to be divided into two subproblems, although u_1^0 through u_1^{N-1} share no constraint and

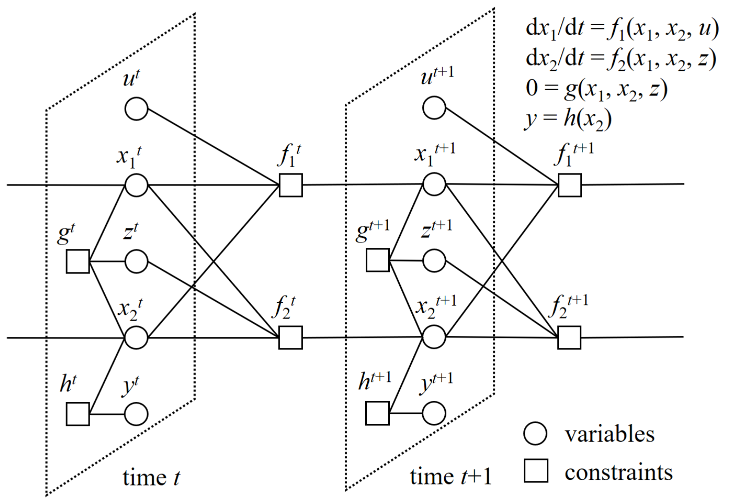


Figure 6.6: An illustration of variable-constraint bipartite network for optimal control problems in NMPC.

u_1^t and u_2^t may interact. Compared to the system digraph-based methods, the variable-constraint network-based method proposed here considers the optimal control problem in its optimization formulation, and detects the communities of optimization variables.

The variable-constraint bipartite network and variable and constraint unipartite networks also provide a more flexible framework than the system digraph, since any other constraints in addition to the system model can be incorporated into the network representation. Moreover, the variable-constraint network can be applied under circumstances where the system digraph can not be well established. For DAE systems which involve algebraic equations, any algebraic constraint can be included by creating a constraint node and variable-constraint edges. For time delay systems where the change of some states is dependent not only on the current states but also on the states of previous times, it suffices to create links between the corresponding constraint node and the state variables with different time superscripts.

6.4 Case Study: Reactor-Separator Process

6.4.1 System description and network decompositions

The reactor-separator process in Fig. 4.1 is a benchmark system for studying distributed model predictive control [234, 397, 421]. The system comprises of two reactors where

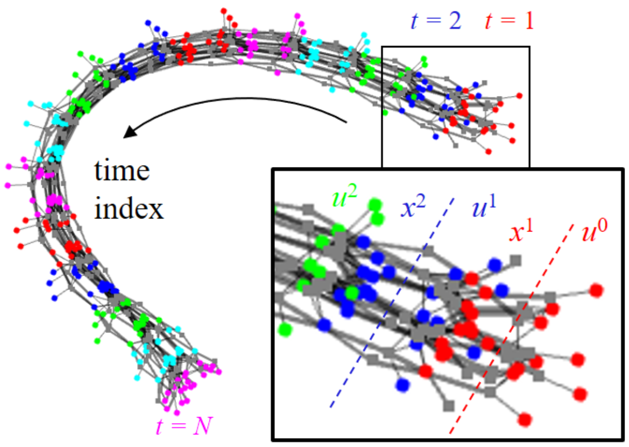


Figure 6.7: Variable-constraint bipartite network for the NMPC of reactor-separator process. The variables are represented by colored circles and the constraints are represented by squares in gray. Different colors represent time indices $t = 1, 2, \ldots, N$ for inputs u^{t-1} and states x^t.

reactions in series A→B→C take place, and a separator producing a product stream withdrawn and a recycle stream. 12 state variables – the concentrations of A and B, temperatures and holdup volumes in three units – describe the system. We use flow rates F_{f1}, F_{f2}, F_1, F_2, F_3, F_R and heat exchange rates Q_1, Q_2, Q_3 as 9 manipulated inputs. The details of the dynamical system model and its NMPC problem are given in Appendix A.

For the optimal control problem in the NMPC of the reactor-separator process, a variable-constraint bipartite network and subsequently a variable unipartite and a constraint unipartite network were constructed. Fig. 6.7 shows the bipartite network, in which the variable nodes are colored according to the time indices (for the inputs u^{t-1} and the states x^t, $t = 1, 2, \ldots, N$). The edges connecting the variable nodes with the constraint nodes in grey represent the variable-constraint interactions. The whip-like bipartite network exhibits a recursive pattern due to the time-invariance of the system, with the time index increasing as we traverse from one end to the other end on the whip. For each time index (i.e., each connected variable group in a specific color in Fig. 6.7), the input and state nodes can be seen clearly, as shown in an amplification of the head of the whip in Fig. 6.7.

To depict the network in a more simplified way, we agglomerate the manipulated inputs and system state variables according to their time indices, and count the number

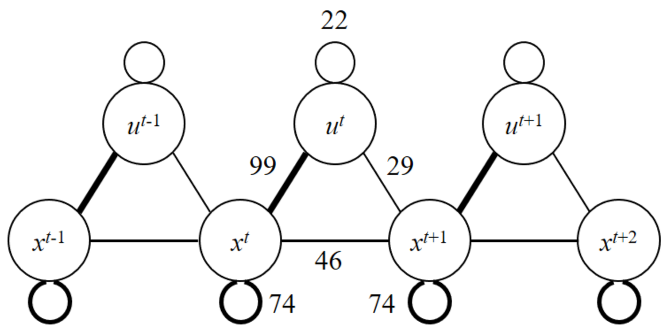


Figure 6.8: Simplified representation of the NMPC of reactor-separator process. Each number represents the total amount of pairwise interactions between two categories of variables.

of pairwise interactions between different types of variables and among themselves (by summing the corresponding edge weights in the unipartite adjacency matrix). This simplification generates a representation of the optimal control problem as a chain of triangles, as shown in Fig. 6.8. To decompose such a network into communities, two of the three edges for some triangles need to be removed. Since the u^t–x^{t+1} and the x^t–x^{t+1} variables have significantly fewer interactions than the u^t–x^t variables, we expect the community detection to generate decompositions by removing the u^t–x^{t+1} and x^t–x^{t+1} edges. In other words, there should be some time indices t such that x^{t+1} is not assigned to the same community as x^t and u^t.

The community detection in the bipartite network and the unipartite network are shown in Fig. 6.9(a) and (b). 8 and 7 communities are generated, respectively. In both cases, the communities are partitioned according to our previous analysis and the decompositions generated appear to be similar. The difference lies in the location where triangles are cut off – the u^t–x^{t+1} and x^t–x^{t+1} edges are removed for $t = 1, 3, 5, 7, 9, 11, 13$ in the bipartite network, and for $t = 2, 4, 6, 8, 10, 12$ in the unipartite network. This pattern, which is closely related to the physical structure of the optimization problem, appears also in the community detection of the constraint unipartite network, the result of which is omitted here for brevity.

For comparison of different decompositions, we apply the community detection in the system digraph [180], which generates 3 communities of process variables, according to the physical topology of the network, and directly extend it to the decomposition of the optimal control problem. That is, for the process variables in each system digraph

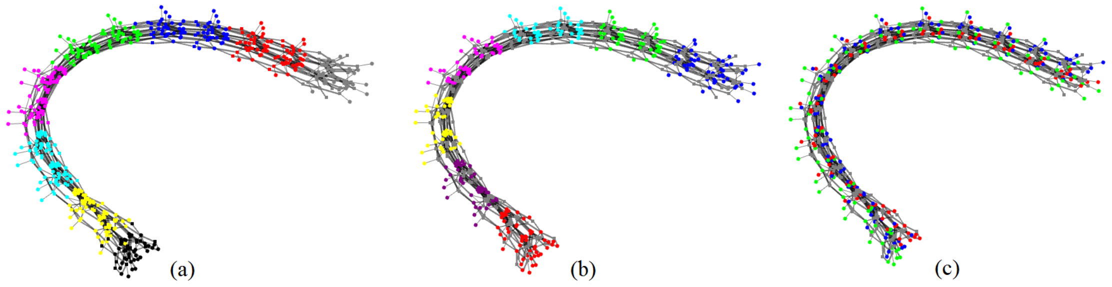


Figure 6.9: Different decompositions in the network representations of the optimal control problem of the reactor-separator process for comparison. (a) Communities in the variable-constraint bipartite network. (b) Communities in the variable unipartite network. (c) Communities in the system digraph. Different colors are used for different communities. Constraints are painted in gray expect for bipartite networks.

community, we assign these process variables with all the time indices $t = 1, 2, \dots, N$ to this community. The decomposition thus obtained in shown in Fig. 6.9(c), where each community (red, blue, green) extends from the head to the tail of the whip. We also consider the decomposition completely based on the time indices of the variables into $N = 15$ subsystems, i.e., each subsystem includes $\{u_{1:9}^{t-1}, x_{1:12}^t\}$ for each $t \in \{1, 2, \dots, 15\}$.

For any decomposition, we characterize its quality by (i) the total *computational time* during the simulation time, and (ii) the control *performance index* (PI), defined as the sum of the stage costs:

$$\mathrm{PI} = \sum_{t=0}^{\infty} \Phi(u^t, x^t)\Delta t. \tag{6.15}$$

Specifically, when the objective function assumes a quadratic form $\Phi(u^t, x^t) = (x^t)^\top Q x^t + (u^t)^\top R u^t$, then PI is the sum of an integrated square error (ISE) term and an integrated square control (ISC) term, namely

$$\mathrm{PI} = \underbrace{\sum_{t=0}^{\infty} (x^t)^\top Q x^t \Delta t}_{\text{ISE}} + \underbrace{\sum_{t=0}^{\infty} (u^t)^\top R u^t \Delta t}_{\text{ISC}}. \tag{6.16}$$

6.4.2 Comparison of decompositions under ADMM

Now we investigate the effect of different decompositions on the distributed NMPC under an ADMM algorithm. The optimal control problem (6.14) for the reactor-separator process can be expressed in the form of (6.1):

$$\min \quad J(v) \quad \text{s.t. } c(v) = 0, \;\; \underline{v} \leq v \leq \overline{v} \tag{6.17}$$

where v is the vector of variables to be optimized containing all the inputs and states within the receding horizon, J is the objective function, $c(v) = [c_1(v), c_2(v), \ldots, c_m(v)]^\top$ is the vector of constraints, and $\underline{v}$ and $\overline{v}$ are the element-wise lower bounds and upper bounds for v, respectively. Community detection is performed based on the structure of the constraints $c(v)$. The individual variable bounds $\overline{v}$ and $\underline{v}$ are not used for the decomposition. The distributed optimization of problem (6.17) in a parallel multi-block ADMM formulation with variable decomposition v^1, v^2, $\ldots$, v^N involves the following iterations:

$$\begin{aligned}
v^1 &:= \arg \min_{\underline{v}^1 \leq z^1 \leq \overline{x}^1} L_\gamma(z^1, v^2, \ldots, v^N; \lambda) \\
v^2 &:= \arg \min_{\underline{v}^2 \leq z^2 \leq \overline{v}^2} L_\gamma(v^1, z^2, \ldots, v^N; \lambda) \\
&\cdots \\
v^N &:= \arg \min_{\underline{v}^N \leq z^N \leq \overline{x}^N} L_\gamma(v^1, v^2, \ldots, z^N; \lambda) \\
\lambda &:= \lambda - \gamma c(v)
\end{aligned} \tag{6.18}$$

with the augmented Lagrangian

$$L_\gamma(v^1, v^2, \ldots, v^N; \lambda) = J(v) - \lambda^\top c(v) + \frac{\gamma}{2}\|c(v)\|_2^2. \tag{6.19}$$

To decrease the computational burden of the primal variable minimization steps, we approximate the $\arg\min_{z^i}$ with the result obtained by performing only one single step of line search in the gradient direction. The step size is chosen according to the Armijo-Goldstein condition [31], i.e.,

$$\begin{aligned}
& v^i := v^i + \alpha d \\
& d = -\nabla_{v^i} L(v^1, v^2, \ldots, v^N; \lambda) \\
& \alpha = \max_{k \in \mathbb{N}} \{\alpha_0 \delta^k | L_\gamma\left(v^1, \ldots, v^i + \alpha_0 \delta^k d, \ldots, v^N; \lambda\right) \leq \\
& \quad L_\gamma(v^1, \ldots, v^i, \ldots, v^N; \lambda) - \beta \alpha_0 \delta^k \|d\|_2^2\}.
\end{aligned} \tag{6.20}$$

More details of the controller are found in Appendix A.

Table 6.1: Comparison of control and computational performances for different decompositions

Decompositions	ISE	ISC	PI	Time	Iter.
Centralized	4339	818	5157	1496 s	1134
(a) Bipart.	3975	966	4941	779 s	3175
(b) Var. Unipart.	4005	975	4980	653 s	2365
(c) Syst. Digraph	3854	1150	5004	1221 s	2647
(d) Time Index	4078	1047	5125	1445 s	max

[1]The "max" stands for the maximum allowed iteration number.

The NMPC based on the decompositions mentioned above along with a centralized ADMM are compared using MATLAB R2017a on a Intel Core i7-2600 CPU with a 16GB RAM, and the results are shown in Table 6.1. Based on the results, we can reach the following conclusions:

1. Compared to centralized control, distributed control under some decompositions (a, b) is capable of decreasing the computational time. Although the distributed algorithms cannot always guarantee a full convergence to the KKT point of the centralized optimal control problem and may be stopped in stagnation, owing to the non-convexity of the problem, it turns out that the decompositions do not deteriorate the overall control performance (PI) compared to that of centralized control.

2. The quality of distributed MPC is not simplistically correlated to the number of subsystems that the original optimization problem is decomposed into. In our case, a fine decomposition into 15 subsystems (d) fails to convergence to an optimum (although the obtained suboptimal solution also gives a stabilizing control). A coarse decomposition into 3 subsystems (c) has longer computational times than the decompositions into 7 and 8 subsystems (a, b). The total computational time is contributed by two factors – the time for a single iteration, and the total iteration number until convergence. In order to find a "good" decomposition, an interaction analysis of the problem structure should be considered.

3. Community detection in the variable unipartite network (b) gives decompositions with better computational performance than that in the variable-constraint bipartite network (a). This is mainly due to the fact that the constraints are dualized in augmented Lagrangian-based optimization algorithms, and hence the interactions

Table 6.2: Comparison of control and computational performances of decomposed and centralized IPOPT

Decompositions	ISE	ISC	PI	Time	Iter.
Centralized	4121	1414	5535	131 s	–
(a) Bipart.	5682	611	6293	25 s	126
(b) Cons. Unipart.	5682	611	6293	35 s	126

between the variables are better captured by the number of terms in which they coexist in the augmented Lagrangian.

4. The optimal decomposition that we have found so far is the one by community detection in the variable unipartite network (b); it brings down the computational time to approximately 44%, and has a control performance slightly better than that of centralized control.

6.4.3 Comparison of decompositions under IPOPT

In the previous subsection, the proposed decomposition method was applied to distributed control under ADMM. However, we also note that ADMM has limitations. For example, its intrinsic slow convergence rate results in a long computation time even for this process system of moderate size.

We also considered a fast nonlinear programming algorithm, IPOPT, for distributed control in the following sense. Through community detection in the variable-constraint bipartite network and constraint unipartite network, we divide the constraints into several groups and find the relevant optimization variables for each group of constraints. In each iteration, we solve in parallel the optimization subproblems, each of which involving a group of constraints and their relevant variables, using IPOPT. If a variable is shared by more than one group of constraints and thus appears in more than one subproblems, each of which generates an optimized value in an iteration, then its value is updated by taking the average. The iterations are terminated according to the same criterion as in ADMM, which is described in Appendix A.

This algorithm is implemented in MATLAB 2017a with OPTI Toolbox v2.27 [70].[2] The results are shown in Table 6.2. It is seen that the decompositions result in a significant decrease in computation time down to about 19% and 27% of that of centralized

[2]Available at `https://www.inverseproblem.co.nz/OPTI/index.php`. See also `https://projects.coin-or.org/Ipopt/`.

control, respectively. A small control performance deterioration over the centralized MPC is also observed. The two decompositions give identical control performance due to the similarity in the community structures detected in the underlying network (as we have discussed in Subsection 6.4.1), while the community detection in the bipartite network results in fewer overlapped variables between different groups of constraints, thus reducing the total computational burden.

We note that distributed optimization algorithms can not in general guarantee the convergence to the optimum of the centralized problem, and therefore, we can not guarantee *a priori* the closed-loop stability even when the centralized controller is stabilizing. To this end, after obtaining a good decomposition through community detection, simulations are still needed to examine the control performance of distributed NMPC.

Remark 6.4. *[67] proposed a decomposition based on finding an approximate Newtonian direction based on a block diagonal approximation of the KKT matrix. In our case, since the interactions between subsystems are not all weak, it is difficult to do such an approximation. Alternatively, it is in principle possible to use the value of Jacobian elements as edge weights and perform community detection in the resulting Jacobian bipartite network. For this example, we observed that this results in a decomposition that does not lead to convergence of the system states towards the steady state.*

Remark 6.5. *Although the application of global optimization in optimal control problems has been proposed by [100], global algorithms have not been prevalent in NMPC applications, probably due to the very high computational effort required. However, since global algorithms (e.g., [10, 358]) guarantee the global optimality of the solution, thus offering a potential of further improving the process economics, their application and influence on the NMPC and the resulting computational performances should be explored. The decomposition of global algorithms for distributed optimization, for which Lagrangian-based global algorithms [38] may provide a useful tool, is also an open problem for future studies.*

6.5 Conclusions and Discussion

Decomposition techniques are widely used for large-scale and complex optimization problems for a distributed solution. Distributed optimization for NMPC has also been a recent focus of research. This article has proposed community detection as a systematic method of decomposing optimization problems applicable to NMPC.

Specifically, we defined a variable-constraint bipartite network and subsequently variable and constraint unipartite networks to capture the structural interactions in the optimization problem, and used community detection as a tool to partition the network into constitutive parts that are strongly interconnected inside but weakly interconnected between. In this way, the computational performance can be benefited by decomposing the problem according to its structure; this is corroborated by a comparison of different decompositions for a constrained ℓ_1-norm minimization. The proposed method was applied to the NMPC of a reactor-separator process. The optimal decompositions found by community detection in the variable unipartite network and the variable-constraint bipartite network significantly reduce the computational time while maintaining the control performance under ADMM and IPOPT algorithms, respectively.

Examples of other complex optimization problems such as in process synthesis, production scheduling and supply chain management can also be considered. Although decompositions have been considered to decrease the complexity of such problems, an optimal decomposition obtained with the proposed method may be advantageous in improving computational efficiency and optimality.

Part II
Optimality: Leveraging Mathematical Programming to Design Control Algorithms

From the east on the grand rock, I view the vast ocean.	東臨碣石，以觀滄海。
The water vibrantly rolls, with isles amidst towering.	水何澹澹，山島竦峙。
Trees have grown luxuriant, and grasses are lush.	樹木叢生，百草豐茂。
In the bleak wind of the autumn, waves go fiercely surging –	秋風蕭瑟，洪波湧起。
as if the Sun and the Moon's motion rises up from between,	日月之行，若出其中。
as if the Milky Way's brilliance rises up from inside.	星漢燦爛，若出其裏。
So blessed to see! I sing this poem of the aspiration of mine.	幸甚至哉！歌以詠志。

Cao Cao, "*Viewing the Ocean*", ca. 207.
曹操《觀滄海》

Chapter 7
Fast and Stable Nonconvex Constrained Distributed Optimization: The ELLADA Algorithm

7.1 Introduction

Distributed optimization [44, 286, 271] refers to methods of performing optimization using a distributed architecture – multiple networked agents are used for subsystems and necessary information among the agents is communicated to coordinate the distributed computation. An important desirable application of distributed optimization is in model predictive control (MPC), where control decisions are made through solving an optimal control problem minimizing the cost associated with the predicted trajectory in a future horizon subject to the system dynamics and operational constraints [344]. For large-scale systems, it is desirable to seek a decomposition (e.g., using community detection or network block structures [77, 404, 76]) and deploy distributed MPC strategies [362, 63, 288], which allows better performance than fully decentralized MPC by enabling coordination, while avoiding assembling and computing on a monolithic model in centralized MPC.

Despite efficient algorithms for solving monolithic nonlinear programming (NLP) problems in centralized MPC (e.g., [316, 249, 36]), extending them into distributed algorithms is nontrivial. A typical approach of distributed MPC is to iterate the control inputs among the subsystems (in sequence or in parallel) [397, 232, 59]. The input iteration routine is typically either semi-decentralized by implicitly assuming that the subsystems interact only through inputs and considering state coupling as disturbances, or semi-centralized by using moving-horizon predictions based on the entire system, which, however, contradicts the fact that the subsystem models should be usually packaged inside the local agents rather than shared over the entire system. Distributed MPC under truly localized model information is typically restricted to linear systems [431, 104, 128].

We note that in general, distributed nonlinear MPC with subsystem interactions should be considered as a *distributed optimization problem under nonconvex constraints*. To solve such problems using distributed agents with local model knowledge, Lagrangian decomposition using dual relaxation of complicating interactions [140, Section 9] and the alternating direction method of multipliers (ADMM) algorithm using augmented Lagrangian [105, 276] were proposed as general frameworks. These are primal-dual iterative algorithms. As illustrated in Fig. 7.1, in each iteration, the distributed agents

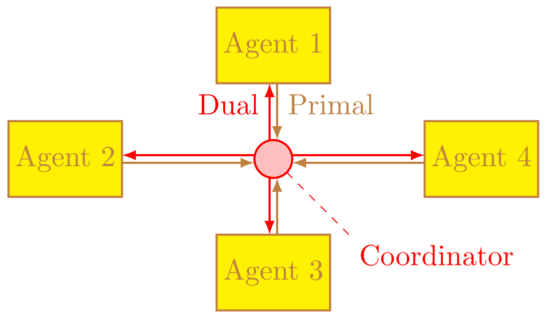


Figure 7.1: Primal-dual distributed optimization.

receive the dual information from the coordinator and execute subroutines to solve their own subproblems, and a coordinator collects information of their solutions to update the duals.

Convergence is the most basic requirement but also a major challenge in distributed optimization under nonconvex constraints. Although distributed optimization with nonconvex objective functions has been well discussed [244, 419, 55, 442], the nonconvex constraints appear much more difficult to handle. To guarantee convergence, [165] suggested dualizing and penalizing all nonconvex constraints, making them undifferentiated and tractable by ADMM; however, this alteration of the problem structure eliminates the option for distributed agents to use any subroutine other than the method of multipliers (MM). [167] used a quadratic programming problem to decide the dual variables in the augmented Lagrangian as well as an extrapolation of primal updates; this algorithm, however, involves a central agent that extracts Hessian and gradient information of the subsystem models from the distributed agents in every iteration, and is thus essentially semi-centralized. [369] adopted feasibility-preserving convex approximations to approach the solution, which is applicable to problems without nonconvex equality constraints. We note that several recent papers (e.g., [400, 177, 463]) proposed the idea of placing slack variables corresponding to the inter-subsystem constraints and forcing the decay to zero by tightening the penalty parameters of slack variables. This modification to the ADMM with slack variables and their penalties leads to a globally convergent *extra-layer* augmented Lagrangian-based algorithm with preserved agent-coordinator problem architecture.

Computational efficiency is also of critical importance for distributed optimization, especially in MPC. The slothfulness of primal-dual algorithms typically arises from two issues. First, the subgradient (first-order) update of dual variables restricts the number of iterations to be of linear complexity [229, 161, 245]. For convex problems, momentum methods [131, 310] can be adopted to obtain second-order dual updates. Such momentum acceleration can not be directly extended to nonconvex problems without a positive definite curvature, although our previous numerical study showed that a discounted momentum may allow limited improvement [413]. Another effort to accelerate dual updates in convex ADMM is based on Krylov subspace methods [486]. Under nonconvexity, it was only very recently realized that *Anderson acceleration*, a multi-secant technique for fixed-point problems, can be generally used to accelerate the dual variables [483, 118, 484].

The second cause for the high computational cost of distributed optimization is the instruction on the distributed agents to fully solve their subproblems to high precision in each iteration. Such exhaustive efforts may be unnecessary since the dual information to be received from the coordinator will keep changing. For convex problems, it is possible to linearize the augmented Lagrangian and replace the distributed subproblems with Arrow-Hurwicz-Uzawa gradient flows [84]. In the presence of nonconvexity of the objective functions, a dual perturbation technique to restore the convergence of the augmented Lagrangian was proposed in [144]. It is yet unknown how to accommodate such gradient flows to nonconvex constraints. A different approach is to allow inexact solution of the subproblems with adaptively tightening tolerances [95]. Such an *approximate ADMM algorithm* allows a better balance between the primal and dual updates, and avoids wasteful computational steps inside the subroutines.

The purpose of this work is to develop a convergent and computationally efficient algorithm for distributed optimization under nonconvex constraints. Although the algorithm is in principle not restricted to any specific problem, we consider the implementation of distributed nonlinear MPC as an important application. Based on the above discussion, we identify the following modifications to the classical ADMM algorithm as the key to mitigating the challenges in convergence and computational complexity: (i) additional slack variables are placed on the constraints relating the distributed agents and the coordinator, (ii) approximate optimization is performed in the distributed agents, and (iii) the Anderson acceleration technique is adopted by the coordinator. We therefore combine and extend as appropriate these techniques into a new algorithm with a two-layer augmented Lagrangian-based architecture, in which the outer layer handles the slack variables as well as inequality constraints by using a barrier technique, and the inner layer performs approximate ADMM under an acceleration scheme. With guaranteed stability and elevated speed, *to the best knowledge of the authors, the proposed algorithm is the first practical and generic algorithm of its kind for distributed nonlinear MPC with truly localized model information.* We name this algorithm as **ELLADA** (standing for **e**xtra-**l**ayer augmented **L**agrangian-based **a**ccelerated **d**istributed **a**pproximate optimization).

This chapter discusses the movitation, develops the ELLADA algorithm and establishes its theoretical properties. An application to a benchmark quadruple tank process is also presented. The remainder of this chapter is organized as follows. In Section 7.2, we first review the classical ADMM and its modified versions. Then we derive our ELLADA algorithm in Section 7.3 with a trilogy pattern. First, a basic two-layer augmented Lagrangian-based algorithm (ELL) is introduced and its convergence is discussed. Then the approximate solution of equality-constrained NLP problems and the Anderson acceleration scheme are incorporated to form the ELLA and ELLADA algorithms. The implementation of the ELLADA algorithm on the distributed optimization problem involved in distributed nonlinear MPC is shown in Section 7.4, and the case study is examined in Section 7.5. Conclusions and discussions are given in Section 7.6.

7.2 ADMM and Its Modifications

7.2.1 ADMM

The alterating direction method of multipliers is the most commonly used algorithm for distributed optimization under linear equality constraints [44]. Specifically, consider the following problem

$$\min_{x,\bar{x}} \ f(x) + g(\bar{x}) \quad \text{s.t.} \ Ax + B\bar{x} = 0 \tag{7.1}$$

with two blocks of variables x and $\bar{x}$, where f and g are usually assumed to be convex. (The symbols in (7.1) are not related to the ones in Section 7.4.) The augmented Lagrangian for such a constrained optimization problem is

$$L(x, \bar{x}; y) = f(x) + g(\bar{x}) + y^\top (Ax + B\bar{x}) + \frac{\rho}{2}\|Ax + B\bar{x}\|^2, \tag{7.2}$$

in which y stands for the vector of dual variables (Lagrangian multipliers) and $\rho > 0$ is called the penalty parameter. According to the duality theory, the optimal solution should be determined by a saddle point of the augmented Lagrangian:

$$\sup_{y} \min_{x,\bar{x}} \; L(x, \bar{x}; y). \tag{7.3}$$

The classical method of multipliers (MM) deals with this saddle point problem with an iterative procedure, where the primal variables are optimized first and then the dual variables are updated with a subgradient ascent [31, Chapter 6]:

$$\begin{aligned} (x^{k+1}, \bar{x}^{k+1}) &= \arg\min_{x,\bar{x}} L(x, \bar{x}; y^k), \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + B\bar{x}^{k+1}), \end{aligned} \tag{7.4}$$

in which the superscript stands for the count of iterations. In a distributed context, x and $\bar{x}$ usually can not be optimized simultaneously. ADMM is thus an approximation of MM that allows the optimization of x and $\bar{x}$ to be performed separately, i.e.,

$$\begin{aligned} x^{k+1} &= \arg\min_{x} L(x, \bar{x}^k; y^k), \\ \bar{x}^{k+1} &= \arg\min_{\bar{x}} L(x^k, \bar{x}; y^k), \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + B\bar{x}^{k+1}). \end{aligned} \tag{7.5}$$

Since the appearance of ADMM in 1970s [129, 119], there have been many works regarding its theoretical properties, extensions and applications. As we have mentioned in the Introduction, ADMM is known to have a linear convergence rate for convex problems. This does not change when the variables are constrained in convex sets. For example, if $x \in \mathcal{X}$, it suffices to modify the corresponding term $f(x)$ in the objective function by adding an indicator function $\mathbb{I}_{\mathcal{X}}(x)$ (equal to 0 if $x \in \mathcal{X}$ and $+\infty$ otherwise), which is still a convex function.

7.2.2 ADMM with approximate updates

Unless the objective terms $f(x)$ and $g(\bar{x})$ are of simple forms such as quadratic functions, the optimization of x and $\bar{x}$ in (7.5) does not have an exact solution. Usually, iterative algorithms for nonlinear programming need to be called for the first two lines of (7.5), and always searching for a highly accurate solution in each ADMM iteration will result in an excessive computational cost. It is thus desirable to solve the optimization subproblems in ADMM inexactly when the dual variables are yet far from the optimum, i.e., to allow x^{k+1} and $\bar{x}^{k+1}$ to be chosen such that

$$d_x^{k+1} \in \partial_x L(x^{k+1}, \bar{x}^k; y^k), \;\; d_{\bar{x}}^{k+1} \in \partial_{\bar{x}} L(x^{k+1}, \bar{x}^{k+1}; y^k), \tag{7.6}$$

where ∂_x and $\partial_{\bar{x}}$ represent the subgradients with respect to x and $\bar{x}$, respectively, and d_x and $d_{\bar{x}}$ are not exactly 0 but only converging to 0 asymptotically. For example, one can assign externally a shrinking and summable sequence of absolute errors [94]:

$$\|d_x^k\| \leq \epsilon_x^k, \;\; \|d_{\bar{x}}^k\| \leq \epsilon_{\bar{x}}^k, \;\; \sum_{k=1}^{\infty} \epsilon_x^k < \infty, \;\; \sum_{k=1}^{\infty} \epsilon_{\bar{x}}^k < \infty, \tag{7.7}$$

or a sequence of relative errors to the errors proportional to other variations in the algorithm [95, 455].

It was shown in [95] that a relative error criterion for terminating the iterations in subproblems, compared to other approximation criteria such as a summable absolute error sequence, better reduces the total number of subroutine iterations throughout the ADMM algorithm. Such a relative error criterion is a *constructive* one, rendered to guarantee the decrease of a quadratic distance between the intermediate solutions $(x^k, \bar{x}^k, y^k)$ and the optimum $(x^*, \bar{x}^*, y^*)$. In the context of distributed optimization problems under nonconvex constraints, since the convergence proof is established on a different basis from the quadratic distance, the construction of such a criterion must be reconsidered. We will address this issue in Subsection 7.3.2.

7.2.3 Anderson acceleration

Linear convergence of the classical ADMM is essentially the result of subgradient dual update, which uses the information of only the first-order derivatives with respect to the dual variables: $\partial_y L = Ax + B\bar{x}$. The idea of creating a *quadratically convergent* algorithm using only first-order derivatives originates back from Nesterov's approach of solving convex optimization problems, which performs iterations based on a linear extrapolation of the previous two iterations instead of the current solution alone [290]. Such a *momentum* method can be used to accelerate the ADMM algorithm, which can be seen as iterations over the second block of primal variables $\bar{x}$ and the dual variables y [131]. However, such a momentum is inappropriate for nonconvex problems, since the behavior of the extrapolated point can not be well controlled by a bound on the curvature of the objective function.

Therefore, we resort to a different type of technique – Anderson acceleration, proposed in [7] first and later "rediscovered" in the field of chemical physics [334]. Generally speaking, Anderson acceleration is used to solve the fixed-point iteration problem $w = h_0(w)$ for some vector w and non-expansive mapping h_0 (satisfying $\|h_0(w) - h_0(w')\| \leq \|w - w'\|$ for any w and w'). Different from the simple Krasnoselskii-Mann iteration $w^{k+1} = \kappa w^k + (1-\kappa)h_0(w^k)$ ($\kappa \in (0,1)$), Anderson acceleration takes a quasi-Newton approach, which aims at a nearly quadratic convergence rate [102]. Specifically[1], in each iteration k, the results from the previous m iterations are recalled from memory to form the matrix of secants in w and $h(w) = w - h_0(w)$:

$$\begin{aligned} \Delta_w^k &= \begin{bmatrix} \delta_w^{k-m} & \dots & \delta_w^{k-1} \end{bmatrix}, \quad \delta_w^{k'} = w^{k'+1} - w^{k'},\ k' = k-m, \dots, k-1; \\ \Delta_h^k &= \begin{bmatrix} \delta_h^{k-m} & \dots & \delta_h^{k-1} \end{bmatrix}, \quad \delta_h^{k'} = h(w^{k'+1}) - h(w^{k'}),\ k' = k-m, \dots, k-1. \end{aligned} \tag{7.8}$$

An estimated inverse Jacobian is given by

$$H_k = I + (\Delta_h^k - \Delta_w^k)(\Delta_w^{k\top}\Delta_w^k)^{-1}\Delta_w^{k\top} \quad \Rightarrow \quad H_k^{-1} = I + (\Delta_w^k - \Delta_h^k)(\Delta_w^{k\top}\Delta_h^k)^{-1}\Delta_w^{k\top}, \tag{7.9}$$

which minimizes the Frobenius norm of $B_k - I$ subject to $B_k \Delta_w^k = \Delta_h^k$. Then the quasi-Newton iteration $w^{k+1} = w^k - H_k^{-1}h^k$ leads to a weighted sum of the previous m function values:

$$w^{k+1} = \sum_{m'=0}^{m} \alpha_{m'}^k h_0(x^{k-m+m'}) \tag{7.10}$$

where the weights $\{\alpha_{m'}^k\}_{m'=0}^m$ are specified by

$$\alpha_{m'}^k = \begin{cases} s_0^k, & m' = 0 \\ s_{m'}^k - s_{m'-1}^k, & m' = 1, \dots, m-1 \\ 1 - s_{m-1}^k, & m' = m \end{cases}, \tag{7.11}$$

with $s_{m'}^k$ being the m'-th component s^k:

$$s^k = (\Delta_w^{k\top}\Delta_h^k)^{-1}\Delta_w^{k\top}h^k. \tag{7.12}$$

[1]There are two different types of Anderson acceleration. Here we focus on Type I, which was found to have better performance [102] and was improved in [483].

Anderson acceleration (7.10) may not always be convergent, although local convergence was studied in some special cases [423]. Recently, a globally convergent modification of Anderson acceleration was proposed in [483], where regularization, restarting, and safeguarding measures are taken to ensure the well-conditioning of the Δ_w^k matrix, boundedness of the inverse Jacobian estimate (7.9), and acceleration only in a safety region, respectively.

The relevance of Anderson acceleration to ADMM lies in that the ADMM algorithm (7.5) can be seen as fixed-point iterations $(\bar{x}^k, y^k) \to (\bar{x}^{k+1}, y^{k+1})$, $k = 0, 1, 2, \dots$ [484], which is the same idea underlying the ADMM with Nesterov acceleration. For problems with nonconvex constraints, the iteration mapping h is not necessarily non-expansive, and hence one can not directly establish the convergence of Anderson acceleration with the original techniques used in [483]. We will address this issue in Subsection 7.3.3.

7.2.4 ADMM under nonconvex constraints

The presence of nonconvexity largely increases the difficulty of distributed optimization. Most of the work in nonconvex ADMM considers problems with nonconvex objective function with bounded Hessian eigenvalues or the Kurdyka-Łojasiewicz property assumptions, under which a convergence rate of $\mathcal{O}(1/\sqrt{k})$ (slower than that of convex ADMM, $\mathcal{O}(1/k)$) was established [221, 162, 442]. However, for many distributed optimization problems, e.g., the distributed MPC of nonlinear processes, there exist nonconvex constraints on the variables, which is intrinsically inequivalent to the problems with nonconvex objectives. For our problem of interest, the relevant works are scarce.

Here we introduce the algorithm of [400] for (7.1) under nonconvex constraints $x \in \mathcal{X}$ and $\bar{x} \in \bar{\mathcal{X}}$, reformulated with slack variables z:

$$\min_{x,\bar{x},z} \; f(x) + g(\bar{x}) \quad \text{s.t. } Ax + B\bar{x} + z = 0, \;\; z = 0, \;\; x \in \mathcal{X}, \;\; \bar{x} \in \bar{\mathcal{X}}. \tag{7.13}$$

The augmented Lagrangian is now written as

$$\begin{aligned} L(x, \bar{x}, z; y, \lambda, \rho, \beta) &= f(x) + g(\bar{x}) + \mathbb{I}_{\mathcal{X}}(x) + \mathbb{I}_{\bar{\mathcal{X}}}(\bar{x}) \\ &+ y^\top (Ax + B\bar{x} + z) + \frac{\rho}{2}\|Ax + B\bar{x} + z\|^2 + \lambda^\top z + \frac{\beta}{2}\|z\|^2. \end{aligned} \tag{7.14}$$

The algorithm is a two-layer one, where each outer iteration (indexed by k) contains a series of inner iterations (indexed by r). In the inner iterations, the classical ADMM algorithm is used to update x, $\bar{x}$, z and y in sequence, while keeping λ and β unchanged:

$$\begin{aligned}
x^{k,r+1} &= \arg\min_{x} L(x, \bar{x}^{k,r}, z^{k,r}; y^{k,r}, \lambda^k, \rho^k, \beta^k) \\
&= \arg\min_{x\in\mathcal{X}} f(x) + \frac{\rho^k}{2}\left\|Ax + B\bar{x}^{k,r} + z^{k,r} + \frac{y^{k,r}}{\rho^k}\right\|^2 \\
\bar{x}^{k,r+1} &= \arg\min_{\bar{x}} L(x^{k,r+1}, \bar{x}, z^{k,r}; y^{k,r}, \lambda^k, \rho^k, \beta^k) \\
&= \arg\min_{\bar{x}\in\bar{\mathcal{X}}} g(\bar{x}) + \frac{\rho^k}{2}\left\|Ax^{k,r+1} + B\bar{x} + z^{k,r} + \frac{y^{k,r}}{\rho^k}\right\|^2 \\
z^{k,r+1} &= \arg\min_{z} L(x^{k,r+1}, \bar{x}^{k,r+1}, z; y^{k,r}, \lambda^k, \rho^k, \beta^k) \\
&= -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k \\
y^{k,r+1} &= y^{k,r} + \rho^k(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1})
\end{aligned} \tag{7.15}$$

Under mild assumptions, in the presence of slack variables z, it was proved [400] that if one chooses $\rho^k = 2\beta^k$, then the inner iterations converge to the set of stationary points $(x^k, \bar{x}^k, z^k, y^k)$ of the relaxed problem

$$\begin{aligned}
\min_{x,\bar{x},z} \;& f(x) + g(\bar{x}) + \lambda^{k\top} z + \frac{\beta^k}{2}\|z\|^2 \\
\text{s.t.} \;& Ax + B\bar{x} + z = 0, \;\; x \in \mathcal{X}, \;\; \bar{x} \in \bar{\mathcal{X}}.
\end{aligned} \tag{7.16}$$

Then in the outer iterations, the dual variables λ^k are updated. To enforce the convergence of the slack variables to zero, the corresponding penalty β^k is amplified by a ratio $\gamma > 1$ if the returned z^k from the inner iterations does not decay enough from the previous outer iteration z^{k-1} ($\|z^k\| > \omega\|z^{k-1}\|$, $\omega \in (0,1)$). The outer iteration is written as

$$\lambda^{k+1} = \Pi_{[\underline{\lambda},\overline{\lambda}]}(\lambda^k + \beta^k z^k), \quad \beta^{k+1} = \begin{cases} \gamma\beta^k, & \|z^k\| > \omega\|z^{k-1}\| \\ \beta^k, & \|z^k\| \le \omega\|z^{k-1}\| \end{cases} \tag{7.17}$$

in which the projection Π onto a predefined compact hypercube $[\underline{\lambda}, \overline{\lambda}]$ is used to guarantee the boundedness of the dual variables and hence the augmented Lagrangian L. If the augmented Lagrangian L remains bounded despite the increase of the penalty parameters ρ^k and β^k, the algorithm converges to a stationary point of the original problem (7.1). The iterative complexity of such an algorithm to reach an ϵ-approximate stationary point is $\mathcal{O}(\epsilon^{-4}\ln(\epsilon^{-1}))$.

In the next section, building on the algorithm of [400] that guarantees the convergence of distributed optimization under nonconvex constraints, we propose a new algorithm that integrates into it the ideas of approximate ADMM and Anderson acceleration, aiming at improving the computational efficiency.

7.3 Proposed Algorithm

7.3.1 Basic algorithm and its convergence

Consider an optimization problem in the following form:

$$\begin{aligned} \min_{x,\bar{x}} \ & f(x) + g(\bar{x}) \\ \text{s.t.} \ & Ax + B\bar{x} = 0, \ \ x \in \mathcal{X} = \{x | \phi(x) \le 0, \psi(x) = 0\}, \ \ \bar{x} \in \bar{\mathcal{X}} \end{aligned} \tag{7.18}$$

or equivalently with slack variables

$$\begin{aligned} \min_{x,\bar{x},z} \ & f(x) + g(\bar{x}) \\ \text{s.t.} \ & Ax + B\bar{x} + z = 0, \ \ z = 0, \ \ x \in \mathcal{X} = \{x | \phi(x) \le 0, \psi(x) = 0\}, \ \ \bar{x} \in \bar{\mathcal{X}}. \end{aligned} \tag{7.19}$$

We make the following assumptions.

Assumption 7.1. *Assume that f is lower bounded, i.e., there exists $\underline{f}$ such that $f(x) \ge \underline{f}$ for any $x \in \mathcal{X}$.*

Assumption 7.2. *Function g is convex and is lower bounded.*

Our basic algorithm (Algorithm 7.1) for (7.19) is slightly modified from the procedure of [400], which considered the case where $g(x) = 0$ and $\bar{\mathcal{X}}$ is a hypercube. The algorithm uses an inner loop of ADMM iterations and an outer loop of MM with possibly amplifying penalty parameters. The inner iterations are terminated when the following criterion is met

$$\begin{aligned} \epsilon_1^k \ge \epsilon_1^{k,r} &:= \|\rho^k A^\top (B\bar{x}^{k,r+1} + z^{k,r+1} - B\bar{x}^{k,r} - z^{k,r})\|, \\ \epsilon_2^k \ge \epsilon_2^{k,r} &:= \|\rho^k B^\top (z^{k,r+1} - z^{k,r})\|, \\ \epsilon_3^k \ge \epsilon_3^{k,r} &:= \|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\|. \end{aligned} \tag{7.20}$$

```
Set: Bound of dual variables $[\underline{\lambda}, \overline{\lambda}]$, shrinking ratio of slack variables $\omega \in [0,1)$, amplifying ratio of penalty parameter $\gamma > 1$, diminishing outer iteration tolerances $\{\epsilon_1^k, \epsilon_2^k, \epsilon_3^k\}_{k=1}^{\infty} \downarrow 0$, terminating tolerances $\epsilon_1$, $\epsilon_2$, $\epsilon_3 > 0$;
Initialization: Starting points $x^0$, $\bar{x}^0$, $z^0$, dual variable and bounds $\lambda^1 \in [\underline{\lambda}, \overline{\lambda}]$, penalty parameter $\beta^1 > 0$;
outer iteration count $k \leftarrow 0$;
while stationarity criterion (7.24) is not met do
    $\rho^k = 2\beta^k$;
    inner iteration count $r \leftarrow 0$;
    Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ satisfying $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;
    while stopping criterion (7.20) is not met do
        $x^{k,r+1} = \arg\min_{x\in\mathcal{X}} f(x) + \frac{\rho^k}{2}\left\|Ax + B\bar{x}^{k,r} + z^{k,r} + \frac{y^{k,r}}{\rho^k}\right\|^2$;
        $\bar{x}^{k,r+1} = \arg\min_{\bar{x}\in\bar{\mathcal{X}}} g(\bar{x}) + \frac{\rho^k}{2}\left\|Ax^{k,r+1} + B\bar{x} + z^{k,r} + \frac{y^{k,r}}{\rho^k}\right\|^2$;
        $z^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;
        $y^{k,r+1} = y^{k,r} + \rho^k(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1})$;
        $r \leftarrow r+1$;
    end
    $(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}) \leftarrow (x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r})$;
    $\lambda^{k+1} = \Pi_{[\underline{\lambda},\overline{\lambda}]}(\lambda^k + \beta^k z^k)$;
    if $\|z^{k+1}\| > \omega\|z^k\|$ then
        $\beta^{k+1} \leftarrow \gamma\beta^k$;
    else
        $\beta^{k+1} \leftarrow \beta^k$;
    end
    $k \leftarrow k+1$;
end
```

Algorithm 7.1: Basic algorithm (ELL).

The proof uses the augmented Lagrangian (7.14) as a decreasing Lyapunov function [162, 161], which gives the convergence of the inner iterations.

Lemma 7.1 (Descent of the augmented Lagrangian)**.** *Suppose that Assumptions 7.1 and 7.2 hold. When $\rho^k = 2\beta^k$, it holds that*

$$\begin{aligned} L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) \leq & L(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}) \\ & - \beta^k\|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2 - \frac{\beta^k}{2}\|z^{k,r+1} - z^{k,r}\|^2 \end{aligned} \tag{7.21}$$

for $r = 0, 1, 2, \ldots$ [2]*, and hence the augmented Lagrangian nonincreasingly converges to*

[2]For simplicity we did not write the last three entries λ^k, ρ^k, β^k that do not change during inner

a limit $\underline{L}_k$.

Corollary 7.1 (Convergence of inner iterations)**.** *Suppose that Assumptions 7.1 and 7.2 hold. As $r \to \infty$, $B\bar{x}^{k,r+1} - B\bar{x}^{k,r} \to 0$, $z^{k,r+1} - z^{k,r} \to 0$, and $Ax^{k,r} + B\bar{x}^{k,r} + z^{k,r} \to 0$. Hence the inner iterations are terminated at a finite r when* (7.20) *is met and the point $(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1})$ the following conditions*

$$\begin{aligned} d_1^k &\in \partial f(x^{k,r+1}) + \mathcal{N}_{\mathcal{X}}(x^{k,r+1}) + A^\top y^{k,r+1}, \\ d_2^k &\in \partial g(\bar{x}^{k,r+1}) + \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k,r+1}) + B^\top y^{k,r+1} \\ 0 &= \lambda^k + \beta^k z^{k,r+1} + y^{k,r+1}, \\ d_3^k &= Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1} \end{aligned} \tag{7.22}$$

for some d_1^k, d_2^k and d_3^k satisfying $\|d_1^k\| \le \epsilon_1^k$, $\|d_2^k\| \le \epsilon_2^k$ and $\|d_3^k\| \le \epsilon_3^k$, respectively. $\mathcal{N}_{\mathcal{X}}(x)$ ($\mathcal{N}_{\bar{\mathcal{X}}}(\bar{x})$) refers to the normal cone to the set $\mathcal{X}$ ($\bar{\mathcal{X}}$) at point x ($\bar{x}$):

$$\mathcal{N}_{\mathcal{X}}(x) = \{v | v^\top (x' - x) \le 0, \ \forall x' \in \mathcal{X}\}. \tag{7.23}$$

The proofs of the above lemma and corollary are given in Appendix B.1 and Appendix B.2, respectively. It is apparent that if $\epsilon_1^k, \epsilon_2^k, \epsilon_3^k$ are all equal to 0, (7.22) is the Karush-Kuhn-Tucker optimality condition of the relaxed problem (7.16) [349].

We note that although the augmented Lagrangian decreases throughout the inner iterations, the increase in the penalty parameters may cause an increase in the augmented Lagrangian across outer iterations, thus losing the guarantee of overall convergence. To establish the convergence of outer iterations, we need to make the following assumption to restrict the upper level of the augmented Lagrangian.

Assumption 7.3. *The augmented Lagrangians are uniformly upper bounded at initialization of all inner iterations, i.e., there exists $\overline{L} \ge L(x^{k,0}, \bar{x}^{k,0}, z^{k,0}, y^{k,0}, \lambda^k, \rho^k, \beta^k)$ for all k.*

The above assumption is actually a "warm start" requirement. Suppose that we have a feasible solution $(x^0, \bar{x}^0)$ to the original problem (7.18), then we can always choose $x^{k,0} = x^0$, $\bar{x}^{k,0} = \bar{x}^0$, $z^{k,0} = 0$, $y^{k,0} = -\lambda^k$ to guarantee an $\overline{L} = f(x^0) + g(\bar{x}^0)$.

Lemma 7.2 (Convergence of outer iterations)**.** *Suppose that Assumptions 7.1, 7.2 and 7.3 hold. Then for any ϵ_1, ϵ_2, and $\epsilon_3 > 0$, within a finite number of outer iterations k,*

iterations in the augmented Lagrangian.

Algorithm 7.1 finds an approximate stationary point $(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1})$ *of* (7.18), *satisfying*

$$\begin{aligned} d_1 &\in \partial f(x^{k+1}) + \mathcal{N}_{\mathcal{X}}(x^{k+1}) + A^\top y^{k+1}, \\ d_2 &\in \partial g(\bar{x}^{k+1}) + \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k+1}) + B^\top y^{k+1}, \\ d_3 &= Ax^{k+1} + B\bar{x}^{k+1} \end{aligned} \tag{7.24}$$

for some d_1, d_2, d_3 *satisfying* $\|d_j\| \le \epsilon_j$, $j = 1, 2, 3$.

See Appendix B.3 for a proof. In addition to the convergence, we can also establish a theoretical complexity. Previously in [400], it was shown that to reach an ϵ-approximate stationary point satisfying (7.24) with $\epsilon_1, \epsilon_2, \epsilon_3 = \epsilon > 0$, the total number of inner iterations needed is of the order $\mathcal{O}(\epsilon^{-4}\ln(1/\epsilon))$. Here, we show that by appropriately choosing the way that the tolerances $(\epsilon_1^k, \epsilon_2^k, \epsilon_3^k)$ shrink, the iteration complexity can be provably reduced anywhere in $(\mathcal{O}(\epsilon^{-2}), \mathcal{O}(\epsilon^{-4})]$, for which a proof is given in Appendix B.4.

Lemma 7.3 (Complexity of the basic algorithm)**.** *Suppose that Assumptions 7.1, 7.2 and 7.3 hold. For some constant* $\vartheta \in (0, \omega]$, *choose* $\epsilon_1^k \sim \mathcal{O}(\vartheta^k)$, $\epsilon_2^k \sim \mathcal{O}(\vartheta^k)$, *and* $\epsilon_3^k \sim \mathcal{O}((\vartheta/\beta)^k)$. *Then each outer iteration* k *requires* $R^k \sim \mathcal{O}((\vartheta\omega)^{-2k})$ *inner iterations. Hence, for the Algorithm 7.1 to reach an* ϵ*-approximate stationary point, the total iterations needed is* $R \sim \mathcal{O}(\epsilon^{-2(1+\varsigma)})$, *where* $\varsigma = \log_\vartheta \omega \in (0, 1]$.

7.3.2 Approximate algorithm

We note that the basic algorithm requires complete minimization of x and $\bar{x}$ in each inner iteration (Lines 9–10, Algorithm 7.1). However, this is neither desirable due to the computational cost, nor practical since any nonlinear programming (NLP) solver finds only a point that approximately satisfies the KKT optimality conditions, except for very simple cases. For simplicity, we assume that such a minimization oracle[3], namely an explicit mapping G depending on matrix B, $Ax^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k$, and ρ^k, exists for $\bar{x}$. For example, if $g(\bar{x}) = 0$ and $B^\top B = aI$ for some $a > 0$, then $G(B, v, \rho) = -\frac{1}{2a}B^\top v$. For the x-minimization, however, such an oracle usually does not exist. In this subsection, we will modify Algorithm 7.1 so as to allow approximate x-optimization on Line 9.

[3]We use the word "oracle" with its typical meaning in mathematics and computer science. An oracle refers to an ad hoc numerical or computational procedure, regarded as a black box mechanism, to generate the needed results as its outputs based on some input information.

Assumption 7.4. *The minimization of the augmented Lagrangian with respect to $\bar{x}$ (Line 10, Algorithm 7.1) admits a unique explicit solution*

$$\bar{x}^{k,r+1} = G(B, Ax^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k). \tag{7.25}$$

Let us also assume that the problem has a smoothness property as follows.

Assumption 7.5. *Functions f, ϕ and ψ are continuously differentiable, and $\mathcal{X}$ has a nonempty interior.*

Under this smoothness assumption, the KKT condition for x-minimization is written as the following equalities with $\mu \geq 0$ and ν representing the Lagrangian dual variables corresponding to the inequalities $\phi(x) \leq 0$ and $\psi(x) = 0$, respectively

$$\begin{aligned}
0 &= \nabla f(x^{k,r+1}) + \rho^k A^\top (Ax^{k,r+1} + B\bar{x}^{k,r} + z^{k,r} + y^{k,r}/\rho^k) \\
&\quad + \sum_{c=1}^{C_\phi} \mu_c \nabla \phi_c(x^{k,r+1}) + \sum_{c=1}^{C_\psi} \nu_c \nabla \psi_c(x^{k,r+1}), \\
0 &= \mu_c \phi_c(x^{k,r+1}),\ \ c = 1, \dots, C_\phi, \quad 0 = \psi_c(x^{k,r+1}),\ \ c = 1, \dots, C_\psi.
\end{aligned} \tag{7.26}$$

Line 9 of Algorithm 7.1 is thus to solve the above equations for $x^{k,r+1}$. This can be achieved through an interior point algorithm, which employs double-layer iterations to find the solution. In the outer iteration, a barrier technique is used to convert the inequality constraints into an additional term in the objective; the optima (or stationary points) of the resulting barrier problems converge to true optima (stationary points) as the barrier parameter converges to 0. In the inner iteration, a proper search method is used to obtain the optimum of the barrier problem. *Since both the interior point algorithm and the basic ADMM algorithm 7.1 have a double-layer structure, we consider matching these two layers.*

Specifically, in the k-th outer iteration, the function $f(x)$ is appended with a barrier term $-b^k \sum_{c=1}^{C_\phi} \ln(-\phi_c(x))$ (b^k is the barrier parameter, converging to 0 as $k \to \infty$). Hence a "barrier augmented Lagrangian" can be specified as

$$L_b = L - b \sum_{c=1}^{C_\phi} \ln(-\phi_c(x)). \tag{7.27}$$

Based on the arguments in the previous subsection, if the x-optimization step returns a

$x^{k,r+1}$ minimizing L_b with respect to x, then the inner iterations result in the descent of L_{b^k}, which implies the satisfaction of conditions (7.22), with f modified by the barrier function. Obviously, if Assumption 7.3 holds for L, then it also holds for L_{b^k} when $\mathcal{X}$ has a nonempty interior. It follows that the outer iterations can find an approximate stationary point of the original problem with the decay of barrier parameters b^k.

However, precisely finding the $x^{k,r+1}$ that minimizes L_b with respect to x, which is an equality-constrained NLP problem, still requires an iterative search procedure [436]. By matching the inner iterations of the interior point algorithm and the inner iterations of the ADMM, we propose to perform only *a proper amount of searching steps* instead of the entire equality-constrained NLP in each inner iteration, so that the solution to the equality-constrained NLP problem can be approached throughout the inner iterations. For this purpose, we assume that we have at hand a solver that can find any approximate solution of equality-constrained NLP.

Assumption 7.6. *Assume that for any equality-constrained smooth NLP problem*

$$\min_x \ \chi(x) \quad \text{s.t. } \psi(x) = 0 \tag{7.28}$$

a solver that guarantees the convergence to any approximate stationary point of the above problem with a lower objective function is available. That is, starting from any initial point x^0, for any tolerances $\epsilon_4, \epsilon_5 > 0$, within a finite number of searches the solver finds a point (x, ν) satisfying

$$d_4 = \nabla\chi(x) + \sum_{c=1}^{C_\psi} \nu_c \nabla\psi_c(x), \quad d_{5c} = \psi_c(x), \ \ c = 1, \dots, C_\psi. \tag{7.29}$$

for some $\|d_4\| \leq \epsilon_4$, $\|d_2\| \leq \epsilon_5$, and $f(x) \leq f(x^0)$. Such an approximate solution is denoted as $F(x^0; \chi, \psi, \epsilon_4, \epsilon_5)$.

The above approximate NLP solution oracle is realizable by NLP solvers where the tolerances of the KKT conditions are allowed to be specified by the user, e.g., the IPOPT solver [437]. Under Assumption 7.6, the x-update step on Line 9 of Algorithm 7.1 is replaced by an approximate NLP solution

$$x^{k,r+1} = F(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r}), \tag{7.30}$$

where the objective function in the current iteration is the part of barrier augmented

```
Set: Bound of dual variables $[\underline{\lambda}, \overline{\lambda}]$, shrinking ratio of slack variables $\omega \in [0, 1)$,
  amplifying ratio of penalty parameter $\gamma > 1$, diminishing outer iteration tolerances
  $\{\epsilon_1^k, \epsilon_2^k, \epsilon_3^k, \epsilon_4^k, \epsilon_5^k, \epsilon_6^k\}_{k=1}^{\infty} \downarrow 0$, diminishing barrier parameters $\{b^k\}_{k=1}^{\infty} \downarrow 0$, terminating
  tolerances $\epsilon_1$, $\epsilon_2$, $\epsilon_3$, $\epsilon_4$, $\epsilon_5$, $\epsilon_6 > 0$;
Initialization: Starting points $x^0$, $\bar{x}^0$, $z^0$, dual variable and bounds $\lambda^1 \in [\underline{\lambda}, \overline{\lambda}]$, penalty
  parameter $\beta^1 > 0$;
outer iteration count $k \leftarrow 0$;
while $\epsilon_4^k \geq \epsilon_4$ or $\epsilon_5^k \geq \epsilon_5$ or $b^k \geq \epsilon_6$ or stationarity criterion (7.24) is not met do
    Set: Diminishing tolerances $\{\epsilon_4^{k,r}, \epsilon_5^{k,r}\}_{r=1}^{\infty} \downarrow 0$;
    let $\rho^k = 2\beta^k$;
    inner iteration count $r \leftarrow 0$;
    Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ satisfying $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;
    while $\epsilon_4^{k,r} \geq \epsilon_4^k$ or $\epsilon_5^{k,r} \geq \epsilon_5^k$ or stopping criterion (7.20) is not met do
        $x^{k,r+1} = F(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r})$, where $\chi^{k,r}$ is given by (7.31);
        $\bar{x}^{r+1} = G(B, Ax^{r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k)$, where $G$ is given by (7.25);
        $z^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;
        $y^{k,r+1} = y^{k,r} + \rho^k(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1})$;
        $r \leftarrow r + 1$;
    end
    $(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}) \leftarrow (x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r})$;
    $\lambda^{k+1} = \Pi_{[\underline{\lambda}, \overline{\lambda}]}(\lambda^k + \beta^k z^k)$;
    if $\|z^{k+1}\| > \omega \|z^k\|$ then
        $\beta^{k+1} \leftarrow \gamma\beta^k$;
    else
        $\beta^{k+1} \leftarrow \beta^k$;
    end
    $k \leftarrow k + 1$;
end
```

Algorithm 7.2: Approximate algorithm (ELLA).

Lagrangian L_{b^k} that is related to x with the indicator function $\mathbb{I}_{\mathcal{X}}(x)$ excluded:

$$\chi^{k,r}(x) = f(x) - b_k \sum_{c=1}^{C_\phi} \ln(-\phi_c(x)) + \frac{\rho^k}{2} \left\| Ax + B\bar{x}^{k,r} + z^{k,r} + y^{k,r}/\rho^k \right\|^2 \qquad (7.31)$$

This approximate algorithm with inexact x-minimization is summarized as Algorithm 7.2. The inner iterations are performed until $\epsilon_4^{k,r}$ and $\epsilon_5^{k,r}$ are lower than ϵ_4^k and ϵ_5^k, respectively, and (7.20) holds. The outer iterations are terminated when $\epsilon_4^k \leq \epsilon_4$, $\epsilon_5^k \leq \epsilon_5$, the barrier parameter is sufficiently small $b^k \leq \epsilon_6$, and (7.24) holds.

Lemma 7.4 (Convergence of the approximate algorithm)**.** *Suppose that Assumptions*

7.1–7.6 hold. For any outer iteration k, given any positive tolerances $\{\epsilon_1^k, \dots, \epsilon_5^k\}$, within a finite number of inner iterations r, the obtained solution satisfies

$$\begin{aligned} d_1^k + d_4^k &= \nabla f(x^{k,r+1}) + \sum_{c=1}^{C_\phi} \mu_c^{k,r+1} \nabla \phi_c(x^{k,r+1}) + \sum_{c=1}^{C_\psi} \nu_c^{k,r+1} \nabla \psi_c(x^{k,r+1}) + A^\top y^{k,r+1}, \\ d_2^k &\in \partial g(\bar{x}^{k,r+1}) + \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k,r+1}) + B^\top y^{k,r+1}, \\ 0 &= \lambda^k + \beta^k z^{k,r+1} + y^{k,r+1}, \quad d_3^k = Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}, \\ d_5^k &= \psi(x^{k,r+1}), \quad -b^k = \mu_c^{k,r+1} \phi_c(x^{k,r+1}), \;\; c = 1, \dots, C_\phi. \end{aligned} \tag{7.32}$$

for some $d_1^k, \dots, d_5^k$ with $\|d_1^k\| \le \epsilon_1^k$, $\dots$, $\|d_5^k\| \le \epsilon_5^k$. Then, suppose that the outer iteration tolerances $\{\epsilon_1^k, \dots, \epsilon_5^k\}$ and barrier parameters b^k are diminishing with increasing k, given any terminating tolerances $\epsilon_1, \dots, \epsilon_6 > 0$, within a finite number of outer iterations, Algorithm 7.2 finds a point $(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}, \mu^{k+1}, \nu^{k+1})$ satisfying

$$\begin{aligned} d_1 + d_4 &= \nabla f(x^{k+1}) + \sum_{c=1}^{C_\phi} \mu_c^{k+1} \nabla \phi_c(x^{k+1}) + \sum_{c=1}^{C_\psi} \nu_c^{k+1} \nabla \psi_c(x^{k+1}) + A^\top y^{k+1} \\ d_2 &\in \partial g(\bar{x}^{k+1}) + \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k+1}) + B^\top y^{k+1}, \\ 0 &= \lambda^k + \beta^k z^{k+1} + y^{k+1}, \quad d_3 = Ax^{k+1} + B\bar{x}^{k+1}, \\ d_5 &= \psi(x^{k+1}), \quad -d_6 = \mu_c^{k+1} \phi_c(x^{k+1}), \;\; c = 1, \dots, C_\phi. \end{aligned} \tag{7.33}$$

for some $d_1, \dots, d_6$ with $\|d_j\| \le \epsilon_j$, $j = 1, \dots, 5$, $d_6 \in (0, \epsilon_6]$.

The proof is self-evident following the techniques in the Proofs of Lemma 7.1, Corollary 7.1 and Lemma 7.2 given in Appendix B.1 to Appendix B.3. The conditions (7.32) indicate an $(\epsilon_1^k, \dots, \epsilon_5^k)$-approximate stationary point to the relaxed barrier problem

$$\begin{aligned} \min_{x,\bar{x},z} \;& f(x) + g(\bar{x}) - b^k \sum_{c=1}^{C_\phi} \ln(-\phi_c(x)) \\ \text{s.t. } \;& Ax + B\bar{x} + z = 0, \;\; \psi(x) = 0, \;\; \bar{x} \in \bar{\mathcal{X}} \end{aligned} \tag{7.34}$$

and the condition (7.33) gives an $(\epsilon_1, \dots, \epsilon_6)$-approximate stationary point to the original problem (7.18).

7.3.3 Accelerated algorithm

The key factor restricting the rate of convergence is the y-update, which is not a full or approximate maximization but only one step of subgradient ascent. As was proved in Lemma 7.3, such a subgradient ascent approach for nonconvex problems leads to a number of inner iterations proportional to the inverse squared error. Here, by modifying the Anderson acceleration scheme in [483], we propose an accelerated algorithm. Let us make the following assumption regarding our choice of tolerances $\epsilon_4^{k,r}$ and $\epsilon_5^{k,r}$.

Assumption 7.7. *Suppose that we choose a continuous and strictly monotonically increasing function $\pi : [0, \infty) \to [0, \infty)$ with $\pi(0) = 0$ such that $\epsilon_5^{k,r} = \pi(\epsilon_4^{k,r})$, and choose $\epsilon_4^{k,r+1}$ proportional to $\|\rho^k A^\top (B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r})\|$ when such a value is strictly smaller than the previous tolerance $\epsilon_4^{k,r}$ but not smaller the ultimate one ϵ_4^k.*

The choice of function π to relate the stationarity tolerance and equality tolerance in NLP subroutine is aimed at balancing the effort to reduce both errors. The choice of $\epsilon_4^{k,r+1}$ is based on the following rationale. After the r-th inner iteration, the obtained solution $x^{k,r+1}$ satisfies the approximate stationarity condition

$$\begin{aligned} d_4^{k,r+1} = \nabla f(x^{k,r+1}) + \sum_{c=1}^{C_\phi} \mu_c^{k,r+1} \nabla \phi_c(x^{k,r+1}) + \sum_{c=1}^{C_\psi} \nu_c^{k,r+1} \cdot \\ \nabla \psi_c(x^{k,r+1}) + \rho^k A^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r} + z^{k,r} + y^{k,r}/\rho^k \right) \end{aligned} \tag{7.35}$$

for some $d_4^{k,r+1}$ with a modulus not exceeding $\epsilon_4^{k,r}$, $\mu^{k,r+1}$ satisfying $\mu_c^{k,r+1} \phi_c(x^{k,r+1}) = -b^k$, $c = 1, \ldots, C_\phi$. Using the formula for y-update (Line 13, Algorithm 7.2), we rearrange the above equation to obtain

$$\begin{aligned} d_4^{k,r+1} + \rho^k A^\top (B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r}) = \nabla f(x^{k,r+1}) \\ + \sum_{c=1}^{C_\phi} \mu_c^{k,r+1} \nabla \phi_c(x^{k,r+1}) + \sum_{c=1}^{C_\psi} \nu_c^{k,r+1} \nabla \psi_c(x^{k,r+1}) + A^\top y^{k,r+1} \end{aligned} \tag{7.36}$$

Hence after the update of $\bar{x}$, z and y variables, the violation of the stationarity condition is bounded by $\epsilon_4^{k,r} + \|\rho^k A^\top (B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r})\|$. Therefore, $\epsilon_4^{k,r}$ should be balanced with the second term, which, however, is realizable only after the $\bar{x}$- and z-updates after the x-update and hence assigned to $\epsilon_4^{k,r+1}$.

We note from Algorithm 7.2 that under Assumption 7.7, each inner iteration r is a mapping from $(x^{k,r}, \bar{x}^{k,r}, z^{k,r+1}, y^{k,r+1}, \epsilon_4^{k,r})$ to $(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}, \epsilon_4^{k,r+1})$. In fact, despite the dependence of the latter variables on $x^{k,r}$ and $\epsilon_4^{k,r}$, such dependence can be ignored in the sense that the descent of the barrier augmented Lagrangian L_{b^k} will always guide the sequence of intermediate solutions towards the set of ϵ_4^k-approximate stationary points of the relaxed barrier problem (7.34). Using Lemma 7.1 with the augmented Lagrangian substituted by the barrier augmented Lagrangian, it immediately follows that under the approximate algorithm, the sequence $\{(\bar{x}^{k,r}, z^{k,r})\}_{r=1}^{\infty}$ will converge to a fixed point, and the convergence of $\{y^{k,r}\}$ accompanies the convergence of $\{z^{k,r}\}$ due to (B.11). It is thus clear that we may resort to Anderson acceleration introduced in Subsection 7.2.3 by denoting $w = (\bar{x}, z)$, the iteration as a mapping h_0, and $h(w) = w - h_0(w)$, and collecting at the r-th inner iteration the following multi-secant information about the previous m inner iterations:

$$\Delta_w^{k,r} = [\delta_w^{k,r-m} \ \ \ldots \ \ \delta_w^{k,r-1}], \ \ \Delta_h^{k,r} = [\delta_h^{k,r-m} \ \ \ldots \ \ \delta_h^{k,r-1}], \tag{7.37}$$

where $\delta_h^{r-m'} = w^{k,r-m'+1} - w^{k,r-m'}$ and $\delta_h^{r-m'} = h(w^{k,r-m'+1}) - h(w^{k,r-m'})$, $m' = m-1, \ldots, 0$.

However, the possibility that Δ_w^k may not be of full rank and H_k may be singular requires certain modifications to the original accelration scheme. The following technique was used in [483]. First, it can be shown that the matrix H defined in (7.9) can be constructed in an inductive way, starting from $H_{k,r}^0 = I$, by rank-one updates

$$H_{k,r}^{m'+1} = H_{k,r}^{m'} + \frac{(\delta_h^{k,r-m+m'} - H_{k,r}^{m'}\delta_w^{k,r-m+m'})(\hat{\delta}_w^{k,r-m+m'})^\top}{(\hat{\delta}_w^{k,r-m+m'})^\top \delta_w^{k,r-m+m'}} \tag{7.38}$$

for $m' = 0, \ldots, m-1$ with $H_{k,r}^m = H_{k,r}$, where $\hat{\delta}_w^{k,r-m}, \ldots, \hat{\delta}_w^{k,r-1}$ are obtained from $\delta_w^{k,r-m}, \ldots, \delta_w^{k,r-1}$ through Gram-Schmidt orthogonalization. To ensure the invertibility of $H_{k,r}$, the δ_h vector in (7.38) is perturbed to

$$\tilde{\delta}_h^{k,r-m+m'} = (1 - \theta_{k,r}^{m'})\delta_h^{k,r-m+m'} + \theta_{k,r}^{m'}\delta_w^{k,r-m+m'}, \tag{7.39}$$

where the perturbation magnitude $\theta_{k,r}^{m'}$ is determined by

$$\theta_{k,r}^{m'} = \varphi\left(\frac{(\hat{\delta}_w^{k,r-m+m'})^\top (H_{k,r}^{m'})^{-1}\delta_h^{k,r-m+m'}}{\|\hat{\delta}_w^{k,r-m+m'}\|^2}; \eta_\theta\right). \tag{7.40}$$

with regularization hyperparameter $\eta_\theta \in (0,1)$. The function $\varphi(\theta;\eta)$ is defined by

$$\varphi(\theta;\eta) = \begin{cases} (\eta \text{sign}\theta - \theta)/(1-\theta) & ,|\theta| \leq \eta \\ 0 & ,|\theta| > \eta \end{cases}. \tag{7.41}$$

Using the Sherman-Morrison formula for inverting the rank-one update, $H_{k,r}^{-1}$ can be induced from $(H_{k,r}^0)^{-1} = I$ according to

$$(H_{k,r}^{m'+1})^{-1} = (H_{k,r}^{m'})^{-1} + \frac{\left(\delta_w^{k,r-m+m'} - (H_{k,r}^{m'})^{-1}\tilde{\delta}_h^{k,r-m+m'}\right)(\hat{\delta}_w^{k,r-m+m'})^\top (H_{k,r}^{m'})^{-1}}{(\hat{\delta}_w^{k,r-m+m'})^\top (H_{k,r}^{m'})^{-1}\tilde{\delta}_h^{k,r-m+m'}}. \tag{7.42}$$

To avoid the rank deficiency Δ_w, a restart checking strategy is used, where the memory is cleared when the Gram-Schmidt orthogonalization becomes ill conditioned ($\|\hat{\delta}_w^{k,r}\| < \eta_w \|\delta_w^{k,r}\|$ for some $\eta_w \in (0,1)$) or the memory exceeds a maximum M; otherwise the memory is allowed to grow. Hence the Anderson acceleration is well-conditioned.

Lemma 7.5 (Well-conditioning of Anderson acceleration, [483]). *Using the regularization and restart checking techniques, it is guaranteed that*

$$\|H_{k,r}^{-1}\|_2 \leq \theta^{-M}\left[3(1+\theta+\eta_w)^M \eta_w^{-N} - 2\right]^{N-1} < +\infty \tag{7.43}$$

where M is the maximum number of steps in the memory and N is the dimension of w.

A well-conditioned Anderson acceleration is not yet sufficient to guarantee the convergence. Hence we employ a safeguarding technique modified from [483] which aims at suppressing a too large increase in the barrier augmented Lagrangian by rejecting such acceleration steps. When Anderson acceleration suggests an update from $w = (\bar{x}, z)$ to $\tilde{w} = (\tilde{\bar{x}}, \tilde{z})$ under the current value of Ax, the resulting Lagrangian increase is

$$\begin{aligned}\tilde{L}^k(w,\tilde{w};Ax) &= g(\tilde{\bar{x}}) - g(\bar{x}) + \lambda^{k\top}(\tilde{z} - z) + \frac{\beta^k}{2}(\|\tilde{z}\|^2 - \|z\|^2) + \tilde{y}^\top(Ax + B\tilde{\bar{x}} + \tilde{z}) \\ &- y^\top(Ax + B\bar{x} + z) + \frac{\rho^k}{2}\left(\|Ax + B\tilde{\bar{x}} + \tilde{z}\|^2 - \|Ax + B\bar{x} + \tilde{z}\|^2\right)\end{aligned} \tag{7.44}$$

where y and $\tilde{y}$ are calculated by

$$y = -\lambda^k - \beta^k z, \;\; \tilde{y} = -\lambda^k - \beta^k \tilde{z}, \tag{7.45}$$

which results from Line 12–13 of Algorithm 7.2. We require that such a change, if positive, must not exceed an upper bound:

$$\tilde{L}^k(w, \tilde{w}; Ax) = \tilde{L}_0 \eta_L (R_+ + 1)^{-(1+\sigma)} \tag{7.46}$$

where $\tilde{L}_0$ is the expected Lagrangian decrease after the first non-accelerated iteration after initialization according to Lemma 7.1, used as a scale for the change in the barrier augmented Lagrangian:

$$\tilde{L}_0 = \beta^k \|B\bar{x}^{k,1} - B\bar{x}^{k,0}\|^2 + \frac{\beta^k}{2}\|z^{k,1} - z^{k,0}\|^2, \tag{7.47}$$

where $\eta_L, \sigma > 0$ are hyperparameters, and R_+ is the number of already accepted acceleration steps. With safeguarding, it can be guaranteed that the barrier augmented Lagrangian always stays bounded, since $\sum_{R_+=0}^{\infty}(R_+ + 1)^{-(1+\sigma)} < +\infty$. We also require that the acceleration should not lead to a drastic change in w:

$$\|\tilde{w} - w\|^2 \leq \frac{\tilde{L}_0}{\beta^k} \frac{\eta_{\tilde{w}}}{\sqrt{1 + R_+}}, \tag{7.48}$$

where $\eta_{\tilde{w}} > 0$ is a hyperparameter. $1/\sqrt{1 + R_+}$ reflects an expected change according to the plain ADMM iteration, which is used to suppress disproportionate large deviations due to Anderson acceleration.

Finally, the accelerated algorithm using the Anderson acceleration technique for fixed-point iteration of $(\bar{x}, z)$ is summarized as Algorithm 7.3. This is our final EL-LADA algorithm, whose distributed implementation will be briefly discussed in the next subsection. With well-conditioned $H_{k,r}^{-1}$ matrix and a bounded barrier augmented Lagrangian, its convergence can now be guaranteed by the following lemma, the proof of which is given in Appendix B.5.

Lemma 7.6 (Convergence under Anderson acceleration)**.** *Suppose that Assumptions 7.1–7.7 hold. Under regulated and safe-guarded Anderson acceleration, Algorithm 7.3 finds within a finite number of inner iterations r a point satisfying* (7.32)*. The convergence of outer iterations to an approximate stationary point satisfying* (7.33) *hence follows.*

Summarizing the conclusions of all the previous lemmas in this section, we have arrived at the following theorem.

```
1  Set: Dual bounds $[\underline{\lambda}, \overline{\lambda}]$, outer iteration parameters $\omega \in [0,1)$, $\gamma > 1$, $\{\epsilon_1^k, \epsilon_2^k, \epsilon_3^k, \epsilon_4^k, \epsilon_6^k\}_{k=1}^{\infty} \downarrow 0$,
     $\{b^k\}_{k=1}^{\infty} \downarrow 0$, final tolerances $\epsilon_1$, $\epsilon_2$, $\epsilon_3$, $\epsilon_4$, $\epsilon_6 > 0$, function $\pi$, acceleration parameters
     $\theta \in (0,1)$, $\sigma > 0$, $\eta_\epsilon > 0$, $\eta_w \in (0,1)$, $\eta_L > 0$, $\eta_{\tilde{w}} > 0$, $M \in \mathbb{N}$. Let $\epsilon_5 = \pi(\epsilon_4)$;
2  Initialization: Starting points $x^0$, $\bar{x}^0$, $z^0$, $\lambda^1 \in [\underline{\lambda}, \overline{\lambda}]$, penalty parameter $\beta^1 > 0$, $\epsilon_5^0 = \pi(\epsilon_4^0)$;
3  Outer iteration count $k \leftarrow 0$;
4  while $\epsilon_4^k \geq \epsilon_4$ or $\epsilon_5^k \geq \epsilon_5$ or $b^k \geq \epsilon_6$ or stationarity criterion (7.24) is not met do
5      Set: Initial tolerances $\epsilon_4^{k,0}$, $\epsilon_5^{k,0} = \pi(\epsilon_4^{k,0})$, penalty $\rho^k = 2\beta^k$, Jacobian estimate $H_{k,0}^{-1} = I$;
6      Inner iteration count $r \leftarrow 0$, count of accelerated steps $R_+^k = 0$, memory length $m \leftarrow 0$;
7      Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ satisfying $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;
8      while $\epsilon_4^{k,r} \geq \epsilon_4^k$ or $\epsilon_5^{k,r} \geq \epsilon_5^k$ or stopping criterion (7.20) is not met do
9          $x^{k,r+1} = F(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r})$, where $\chi^{k,r}$ is given by (7.31);
10         $\bar{x}^{k,r+1} = G(B, Ax^{r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k)$, where $G$ is given by (7.25);
11         $z^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;
12         $y^{k,r+1} = y^{k,r} + \rho^k(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1})$;
13         $\tilde{y}^{k,r} = -\lambda^k - \beta^k \tilde{z}^{k,r}$;
14         $\tilde{x}^{k,r+1} = F(x^{k,r}; \tilde{\chi}^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r})$, with $\tilde{\chi}$ in (7.31) with $\bar{x}, z, y$ replaced by $\tilde{\bar{x}}, \tilde{z}, \tilde{y}$;
15         $\tilde{\bar{x}}^{k,r+1} = G(B, A\tilde{x}^{r+1} + \tilde{z}^{k,r} + \tilde{y}^{k,r}/\rho^k, \rho^k)$;
16         $\tilde{z}^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(A\tilde{x}^{k,r+1} + B\tilde{\bar{x}}^{k,r+1} + \frac{\tilde{y}^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;
17         if $r = 0$ then
18             $\tilde{w}^{k,1} \leftarrow (\bar{x}^{k,1}, z^{k,1})$, and calculate $\tilde{L}_0$ by (7.47);
19         else
20             $\delta_w^{k,r-1} = \tilde{w}^{k,r} - w^{k,r-1}$, $\delta_h^{k,r} = \tilde{w}^{k,r} - \tilde{w}^{k,r+1} - w^{k,r-1} + w^{k,r}$, $m \leftarrow m+1$;
21             $\hat{\delta}_w^{k,r-1} = \delta_w^{k,r-1} - \sum_{m'=2}^{m} \frac{(\hat{\delta}_w^{k,r-m'})^\top \delta_w^{k,r-1}}{\|\hat{\delta}_w^{k,r-m'}\|^2} \hat{\delta}_w^{k,r-m'}$;
22             if $m = M+1$ or $\frac{\|\hat{\delta}_w^{k,r-1}\|}{\|\delta_w^{k,r-1}\|} < \eta_w$ then $m \leftarrow 0$, $\hat{\delta}_w^{k,r-1} \leftarrow \delta_w^{k,r-1}$, and $H_{k,r-1}^{-1} \leftarrow I$;
23             Compute $\tilde{\delta}_h^{k,r-1}$ by (7.39) with $m' = m-1$ and
                 $\theta_{k,r-1} = \varphi\left(\frac{(\hat{\delta}_w^{k,r-1})^\top H_{k,r-1}^{-1} \delta_h^{k,r-1}}{\|\hat{\delta}_w^{k,r-1}\|^2}; \theta\right)$;
24             Update $H_{k,r}^{-1} = H_{k,r-1}^{-1} + \frac{(\delta_w^{k,r-1} - H_{k,r-1}^{-1}\tilde{\delta}_h^{k,r-1})(\hat{\delta}_w^{k,r-1})^\top H_{k,r-1}^{-1}}{(\hat{\delta}_w^{k,r-1})^\top H_{k,r-1}^{-1} \tilde{\delta}_w^{k,r-1}}$, and suggest
                 $\tilde{w}^{k,r+1} = w^{k,r} - H_{k,r}^{-1}(w^{k,r} - w^{k,r+1})$;
25             if $\frac{\tilde{L}^k(w^{k,r}, \tilde{w}^{k,r+1}; Ax^{k,r})}{\tilde{L}_0 \eta_L (R_+ + 1)^{-(1+\sigma)}} \leq 1$ and $\|\tilde{w}^{k,r+1} - w^{k,r}\|^2 \leq \frac{\tilde{L}_0 \eta_{\tilde{w}}}{\beta^k \sqrt{R_+ + 1}}$ then accept the
                 acceleration $w^{k,r+1} \leftarrow \tilde{w}^{k,r+1}$, and let $y^{k,r+1} \leftarrow -\lambda^k - \beta^k \tilde{z}^{k,r+1}$;
26         end
27         $\epsilon_4^{k,r+1} = \|\rho^k A^\top (B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r})\|$, $\epsilon_5^{k,r+1} = \pi(\epsilon_4^{k,r+1})$;
28         $r \leftarrow r+1$;
29     end
30     $(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}) \leftarrow (x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r})$;
31     Update $\lambda^{k+1} = \Pi_{[\underline{\lambda}, \overline{\lambda}]}(\lambda^k + \beta^k z^k)$ and $\beta^{k+1}$ according to (7.17);
32     $k \leftarrow k+1$;
33 end
```

Algorithm 7.3: Accelerated algorithm (ELLADA).

Theorem 7.1. *Suppose that the following assumptions hold:*

1. *Function f is lower bounded on $\mathcal{X}$;*
2. *Function g is convex and lower bounded on $\bar{\mathcal{X}}$;*
3. *Initialization of outer iterations allows a uniform upper bound of the augmented Lagrangian, e.g., a feasible solution is known a priori;*
4. *Minimization of $g(\bar{x}) + \frac{\rho}{2}\|B\bar{x} + v\|^2$ with respect to $\bar{x}$ allows an oracle $G(B, v, \rho)$ returning a unique solution for any v of appropriate dimension and $\rho > 0$;*
5. *Functions f, ϕ, and ψ are continuously differentiable, and the constraints (ϕ, ψ) are strictly feasible;*
6. *There exists a solver for equality-constrained NLP to any specified tolerances of KKT conditions.*

Then given any tolerances $\epsilon_1, \dots, \epsilon_6 > 0$, the ELLADA algorithm (Algorithm 7.3) gives an $(\epsilon_1, \dots, \epsilon_6)$-approximate KKT point satisfying the conditions (7.33).

If the problem itself has intrinsically better properties to guarantee that each KKT point is a local minimum, e.g., the second-order sufficient condition [31, §4.3.2], then the algorithm converges to a local minimum. Of course, it is well known that certifying a local minimum is itself a difficult problem.

7.4 Implementation on Distributed Nonlinear MPC

Consider a nonlinear discrete-time dynamical system

$$x(t+1) = f(x(t), u(t)) \tag{7.49}$$

where $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ are the vectors of states and inputs, respectively, for $t = 0, 1, 2, \dots$, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. Suppose that at time t we have the current states $x = x(t)$, then in MPC, the control inputs are determined by the following optimal control problem:

$$\begin{aligned} \min\ & J = \sum_{\tau=t}^{t+T-1} \ell(\hat{x}(\tau), \hat{u}(\tau)) + \ell^{\mathrm{f}}(\hat{x}(t+T)) \\ \text{s.t. } & \hat{x}(\tau+1) = f(\hat{x}(\tau), \hat{u}(\tau)),\ \ \tau = t, \dots, t+T-1 \\ & p(\hat{x}(\tau), \hat{u}(\tau), \tau) \le 0,\ \ q(\hat{x}(\tau), \hat{u}(\tau), \tau) = 0,\ \ \tau = t, \dots, t+T-1 \\ & \hat{x}(t) = x. \end{aligned} \tag{7.50}$$

In the above formulation, the optimization variables $\hat{x}(\tau)$ and $\hat{u}(\tau)$ represent the predicted states and inputs in a future horizon $\{t, t+1, \dots, t+T\}$ with length $T \in \mathbb{N}$. The predicted trajectory is constrained by the dynamics (7.49) as well as some additional path constraints p, q such as the bounds on the inputs and states or Lyapunov descent to enforce stability. Functions ℓ and ℓ^{f} are called the stage cost and terminal cost, respectively. By solving (7.50), one executes $u(t) = \hat{u}(t)$. For simplicity it is assumed here that the states are observable; otherwise, the states can be estimated using an optimization formulation such as moving horizon estimation (MHE). For continuous-time systems, collocation techniques can be used to discretize the resulting optimal control problem into a finite-dimensional one.

Now suppose that the system (7.49) is large-scale with its states and outputs decomposed into n subsystems: $x = [x_1^\top, x_2^\top, \dots, x_n^\top]^\top$, $u = [u_1^\top, u_2^\top, \dots, u_n^\top]^\top$, and that the optimal control problem should be solved by the corresponding n agents, each containing the model of its own subsystem:

$$x_i(\tau+1) = f_i(x_i(\tau), u_i(\tau), \{x_{ji}(\tau), u_{ji}(\tau)\}_{j\in\mathcal{P}(i)}). \tag{7.51}$$

where $\{x_{ji}, u_{ji}\}$ stands for the states and inputs in subsystem j (i.e., components of x_j and u_j) that appear in the arguments of f_i, which comprise of the components of f corresponding to the i-th subsystem. $\mathcal{P}_i$ is the collection of subsystems j that has some inputs and outputs influencing subsystem i. We assume that the cost functions and the path constraints are separable:

$$\begin{aligned}
&\ell(\hat{x}, \hat{u}) = \sum_{i=1}^{n} \ell_i(\hat{x}_i, \hat{u}_i), \;\; \ell^{\mathrm{f}}(\hat{x}) = \sum_{i=1}^{n} \ell_i^{\mathrm{f}}(\hat{x}_i), \\
&p(\hat{x}, \hat{u}, \tau) = [p_1(\hat{x}_1, \hat{u}_1, \tau)^\top, \dots, p_n(\hat{x}_n, \hat{u}_n, \tau)^\top]^\top, \\
&q(\hat{x}, \hat{u}, \tau) = [q_1(\hat{x}_1, \hat{u}_1, \tau)^\top, \dots, q_n(\hat{x}_n, \hat{u}_n, \tau)^\top]^\top.
\end{aligned} \tag{7.52}$$

7.4.1 Formulation on directed and bipartite graphs

To better visualize the problem structure and systematically reformulate the optimal control problem (7.50) into the distributed optimization problem in the form of (7.19) for the implementation of the ELLADA algorithm, we introduce some graph-theoretic descriptions of optimization problems [76]. For problem (7.50), we first define a directed graph (digraph), which is a straightforward characterization of the relation of mutual impact among the subsystem models.

Definition 7.1 (Digraph)**.** *The digraph of system* (7.49) *under the decomposition* $x = [x_1^\top, x_2^\top, \dots, x_n^\top]^\top$ *and* $u = [u_1^\top, u_2^\top, \dots, u_n^\top]^\top$ *is* $\mathcal{G}_1 = \{\mathcal{V}_1, \mathcal{E}_1\}$ *with nodes* $\mathcal{V}_1 = \{1, 2, \dots, n\}$ *and edges* $\mathcal{E}_1 = \{(j,i) | j \in \mathcal{P}(i)\}$*. If* $(i,j) \in \mathcal{E}_1$*, i.e.,* $j \in \mathcal{P}(i)$*, we say that* j *is a parent of* i *and* i *is a child of* j *(denoted as* $i \in \mathcal{C}(j)$*).*

Then under the decomposition, (7.50) can be written as

$$\begin{aligned} \min \ & \sum_{i\in\mathcal{V}_1} J_i = \sum_{i\in\mathcal{V}} \sum_{\tau=t}^{t+T-1} \ell_i(\hat{x}_i(\tau), \hat{u}_i(\tau)) + \ell_i^{\mathrm{f}}(\hat{x}_i(t+T)) \\ \text{s.t. } & \hat{x}_i(\tau+1) = f_i(\hat{x}_i(\tau), \hat{u}_i(\tau), \{\hat{x}_{ji}(\tau), \hat{u}_{ji}(\tau)\}_{j\in\mathcal{P}(i)}), \\ & p_i(\hat{x}_i(\tau), \hat{u}_i(\tau), \tau) \le 0, \ \ \tau = t, \dots, t+T-1, \ \ i \in \mathcal{V}_1 \\ & q_i(\hat{x}_i(\tau), \hat{u}_i(\tau), \tau) = 0, \ \ \tau = t, \dots, t+T-1, \ \ i \in \mathcal{V}_1 \\ & \hat{x}_i(t) = x_i, \ \ i \in \mathcal{V}_1, \end{aligned} \tag{7.53}$$

We denote the variables of the i-th agent as

$$\begin{aligned} \xi_i = [& \hat{x}_i(t)^\top, \hat{u}_i(t)^\top, \dots, \hat{x}_i(t+T-1)^\top, \hat{u}_i(t+T-1)^\top, \hat{x}_i(t+T)^\top, \\ & \{\hat{x}_{ji}(t)^\top, \hat{u}_{ji}(t)^\top\}_{j\in\mathcal{P}(i)}, \dots, \{\hat{x}_{ji}(t+T-1)^\top, \hat{u}_{ji}(t+T-1)^\top\}_{j\in\mathcal{P}(i)}]^\top, \end{aligned} \tag{7.54}$$

in which the variables related to the j-th subsystem are denoted as ξ_{ji}. Since ξ_{ji} is a part of the predicted states and inputs from subsystem j, i.e., some components of ξ_j, the interactions between the parent j and the child i be captured by a matrix $\overrightarrow{D}_{ji}$ with exactly one unit entry ("1") on every row: $\xi_{ji} = \overrightarrow{D}_{ji}\xi_j$, where the right arrow represents the impact of the parent subsystem j on the child subsystem i. By denoting the model and path constraints in agent i as $\xi_i \in \Xi_i$, the optimal control problem (7.49) is expressed in a compact way as follows:

$$\begin{aligned} \min \ & \sum_{i\in\mathcal{V}_1} J_i(\xi_i) \\ \text{s.t. } & \xi_i \in \Xi_i, \ \ i \in \mathcal{V}_1, \ \ \xi_{ji} = \overrightarrow{D}_{ji}\xi_j, \ \ (j,i) \in \mathcal{E}_1. \end{aligned} \tag{7.55}$$

This is an optimization problem defined on a *directed graph.* An illustration for a simple case when $\mathcal{E}_1 = \{(1,2), (2,3), (3,1)\}$ is shown in Fig. 7.2(a).

Although it is natural to represent the interactions among the subsystems in a digraph, performing distributed optimization on digraphs where the agents communicate

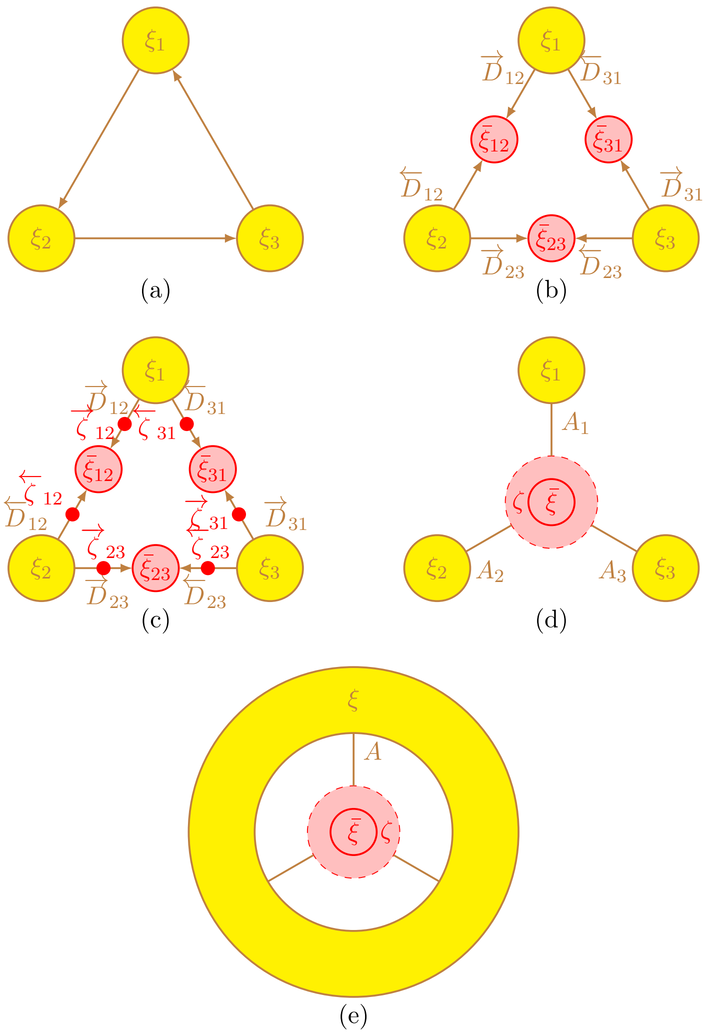


Figure 7.2: Graphical illustrations of the problem structure of distributed MPC.

among themselves without a coordinator can be challenging. For example, it is known that the ADMM algorithm, which behaves well for distributed optimization with 2 blocks of variables, can become divergent when directly extended to multi-block problems [56]. Hence we construct such a 2-block architecture by using a *bipartite graph*.

Definition 7.2 (Bipartite graph)**.** *The bipartite graph of system* (7.49) $\mathcal{G}_2$ *is constructed from the digraph* $\mathcal{G}_1$ *by taking both the nodes and edges as the new nodes, and adding an edge between* $i \in \mathcal{V}_1$ *and* $e \in \mathcal{E}_1$ *if* i *is the head or tail of* e *in the digraph, i.e.,* $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2)$ *with* $\mathcal{V}_2 = \mathcal{V}_1 \cup \mathcal{E}_1$*,* $\mathcal{E}_2 = \{(i,e) | i \in \mathcal{V}_1, e \in \mathcal{E}_1, e = (i,j), j \in \mathcal{C}(i)$ *or* $e =$

$(j,i), j \in \mathcal{P}(i)\}$.

Such a graph is bipartite since any edge is between a node of $\mathcal{V}_1$ and a node of $\mathcal{E}_1$.

We note that the last line of (7.55) corresponds to the digraph edges $\mathcal{E}_1$. In the bipartite graph, these edges should become nodes and hence new groups of variables should be associated with them. For this purpose, we simply need to pull out ξ_{ji} as overlapping variables $\bar{\xi}_{ji}$, and add the constraint that $\bar{\xi}_{ji}$ are some selected components of ξ_i: $\xi_{ji} = \overleftarrow{D}_{ji}\xi_i$:

$$\begin{aligned} \min \ & \sum_{i\in\mathcal{V}_1} J_i(\xi_i) \\ \text{s.t.} \ & \xi_i \in \Xi_i, \ \ i \in \mathcal{V}_1, \ \ \bar{\xi}_{ji} = \overrightarrow{D}_{ji}\xi_j = \overleftarrow{D}_{ji}\xi_j, \ \ (j,i) \in \mathcal{E}_1 \end{aligned} \tag{7.56}$$

In (7.56), variables ξ_i ($i \in \mathcal{V}_1$) and $\bar{\xi}_{ji}$ ($(j,i) \in \mathcal{E}_1$) are defined on the nodes of the bipartite graph, and the constraints captured by the matrices $\overrightarrow{D}_{ji}$ and $\overleftarrow{D}_{ji}$ correspond to the bipartite edges $(j,(j,i))$ and $(i,(j,i))$, respectively. We may also write the last line of (7.56) as

$$\bar{\xi}_e = D_{ie}\xi_i, \ \ (i,e) \in \mathcal{E}_2. \tag{7.57}$$

Therefore (7.56) is an optimization problem on the bipartite graph. An illustration is given in Fig. 7.2(b). Under this reformulation, the problem structure becomes a 2-block one – distributed agents $i = 1, \ldots, N$ manage the decision variables ξ_i, $\mathcal{V}_1$ in parallel without interference, and the coordinator regulates the agents by using overlapping variables $\bar{\xi}_e$, $e \in \mathcal{E}_1$.

7.4.2 Reformulation with slack variables

It is known that a key condition for distributed optimization in the context of the ADMM algorithm to converge is that one block of variables can always be made feasible given the other block [442]. Unfortunately this condition is not always met by the problem (7.56). For example, given ξ_1 and ξ_2, there may not be a $\bar{\xi}_{12}$ satisfying both $\bar{\xi}_{12} = \overrightarrow{D}_{12}\xi_1$ and $\bar{\xi}_{12} = \overleftarrow{D}_{12}\xi_2$. To deal with this issue, it was proposed to associate with each linear constraint in (7.56), namely each edge in the bipartite graph, a slack variable ζ_{ie}:

$$\begin{aligned} \min \ & \sum_{i\in\mathcal{V}_1} J_i(\xi_i) \\ \text{s.t.} \ & \xi_i \in \Xi_i, \ \ i \in \mathcal{V}_1, \quad D_{ie}\xi_i - \bar{\xi}_e + \zeta_{ie} = 0, \ \ (i,e) \in \mathcal{E}_2, \quad \zeta_{ie} = 0, \ \ (i,e) \in \mathcal{E}_2. \end{aligned} \tag{7.58}$$

Similar to the notation for D, we write ζ_{ie} as $\overrightarrow{\zeta}_{ij}$ if $e = (i, j)$ and $\overleftarrow{\zeta}_{ij}$ if $e = (j, i)$. Such a problem structure is graphically illustrated in Fig. 7.2(c).

Finally, we stack all the subscripted variables into ξ, $\bar{\xi}$, ζ in a proper ordering of $i \in \mathcal{V}_1$, $e \in \mathcal{E}_1$, and $(i, e) \in \mathcal{E}_2$. The matrices D_{ie} are stacked in a block diagonal pattern in the same ordering of $(i, e) \in \mathcal{E}_2$ into A. The appearance of $\bar{\xi}_e$ in the equality constraints is represented by a matrix B (satisfying $B^\top B = 2I$). We write the objective function as $J(\xi)$, and the set constraints Ξ_i are lumped into a Cartesian product $\Xi = \times_{i\in\mathcal{V}_1}\Xi_i$. Finally, we reach a compact formulation for (7.58):

$$\begin{aligned} &\min\ J(\xi) \\ &\text{ s.t. }\ \xi \in \Xi,\ \ A\xi + B\bar{\xi} + \zeta = 0,\ \ \zeta = 0 \end{aligned} \tag{7.59}$$

Such an architecture is shown in Figs. 7.2(d) and 7.2(e). The variables $\bar{\xi}$ and ζ belong to the coordinator (marked in red), and ξ is in the distributed agents.

7.4.3 Implementation of ELLADA

Clearly, the optimal control problem formulated as (7.58) is a special form of (7.19) with ξ, $\bar{\xi}$ and ζ rewritten as x, $\bar{x}$ and z, respectively, and $g(\bar{x}) = 0$, $\bar{\mathcal{X}}$ equal to the entire Euclidean space. As long as the cost function J is lower bounded (e.g., a quadratic cost), Algorithm 7.3 is applicable to (7.58), where the operations on $\bar{x}$, z, y are performed by the coordinator, and the operations on x is handled by the distributed agents. Specifically,

- The update steps of $\bar{x}, z, y$ (Lines 10–13, 15, 16) and the entire Anderson acceleration (Lines 17–26) belong to the coordinator. The updates of penalty parameters and outer-layer dual variables λ (Lines 31) should also be performed by the coordinator. The conditions for $\epsilon_1^k, \epsilon_2^k, \epsilon_3^k$ and $\epsilon_1, \epsilon_2, \epsilon_3$ are checked by the coordinator.
- The distributed agents are responsible for carrying out a trial x-update step for the Anderson acceleration (Line 9) as well as the plain x-update (Line 14). The conditions and updates for $\epsilon_4^{k,r}, \epsilon_5^{k,r}, \epsilon_4^k, \epsilon_5^k$, and $\epsilon_4, \epsilon_5, \epsilon_6$ are checked by the agents.

When executing the updates, the agents need the values of $B\bar{x} + z + y/\rho$ to add to Ax, and the coordinator needs the value of Ax from the agents. When the variables x are distributed into agents $x_1, \dots, x_n$, and the equality constraints between the agents and the coordinator is expressed on a bipartite graph:

$$D_{ie}x_i - \bar{x}_e + z_{ie} = 0, \;\; (i,e) \in \mathcal{E}_2, \tag{7.60}$$

the communication of Ax and $B\bar{x} + z + y/\rho$ takes place in a distributed and parallel way, i.e., the i-th agent obtains the information of $-\bar{x}_e + z_{ie} + y_{ie}/\rho$ for all e such that $(i,e) \in \mathcal{E}_2$ from the coordinator. The coordinator, based on inter-subsystem edges e in the digraph, obtains the information of $D_{ie}x_i$ for all related agents i. When the objective function and $\mathcal{X}$ are separable $f(x) = \sum_{i=1}^{n} f_i(x_i)$, $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, based on such distributed and parallel communication, the optimization problem

$$\begin{aligned} &\min \; f(x) - b\sum_{c=1}^{C_\phi} \ln(-\phi_c(x)) + \frac{\rho}{2}\left\| Ax + B\bar{x} + z + \frac{y}{\rho} \right\|^2 \\ &\text{s.t. } \; \psi(x) = 0 \end{aligned} \tag{7.61}$$

in an x-update step can be solved in a distributed and parallel manner:

$$\left.\begin{aligned} &\min_{x_i} \; f_i(x_i) - b\sum_{c=1}^{C_{\phi,i}} \ln(-\phi_{c,i}(x_i)) \\ &\qquad + \frac{\rho}{2} \sum_{\{e|(i,e)\in\mathcal{E}_2\}} \left\| D_{ie}x_i - \bar{x}_e + z_{ie} + \frac{y_{ie}}{\rho} \right\|^2 \\ &\text{s.t. } \; \psi_i(x_i) = 0 \end{aligned}\right\} \; /\!/ \text{ for } i. \tag{7.62}$$

Similarly, the $\bar{x}$-update with the G-mapping is in parallel for its components e, if $\bar{\mathcal{X}}$ is separable, i.e., if $\bar{\mathcal{X}}$ is a closed hypercube (whether bounded or unbounded), and if g is also separable. That is, $\bar{x}$-update can be expressed as

$$\left.\begin{aligned} &\min_{\bar{x}_i} \; g_i(\bar{x}_i) + \frac{\rho}{2} \sum_{\{i|(i,e)\in\mathcal{E}_2\}} \left\| D_{ie}x_i - \bar{x}_e + z_{ie} + \frac{y_{ie}}{\rho} \right\|^2 \\ &\text{s.t. } \; \bar{x}_i \in \bar{\mathcal{X}}_i \end{aligned}\right\} \; /\!/ \text{ for } e. \tag{7.63}$$

The z and y updates are in parallel for the edges (i,e) on the bipartite graph.

In Algorithm 7.3, the procedures are written such that in each iteration, the update steps are carried out in sequence. This requires a synchronization of all the agents i and the coordinating elements e and (i,e). For example, for the x-update, every distributed agent needs to create a “finish” signal after solving x_i in (7.62) and send it to the coordinator. Only after the coordinator receives the "finish" signals from all the distributed agents can the $\bar{x}$-update be carried out. Due to the possible computational imbalance among the agents and the coordinator, such synchronization implies that faster updates must idle for some time to wait for slower ones. In fact, the convergence properties of the ELLADA algorithm do not rely on the synchronization. Even when the inner iterations are asynchronous, the update steps still contribute to the convergence of the barrier augmented Lagrangian and hence result in convergence to KKT conditions. The only exception is that under Anderson acceleration, the steps for generating the candidate of accelerated updates are allocated to another coordinator and another set of distributed agents, and they should communicate to make the decision on executing the accelerations.

7.5 Application to a Quadruple Tank Process

The quadruple tank process is a simple benchmark process for distributed model predictive control [181] with 4 states (water heights in the 4 tanks) and 2 inputs (flow rates from the reservoir). The dynamic model is written as follows:

$$\begin{aligned}
\dot{h}_1 &= -\frac{a_1}{A_1}\sqrt{h_1} + \frac{a_3}{A_1}\sqrt{h_3} + \frac{\gamma_1 k_1}{A_1} v_1 \\
\dot{h}_2 &= -\frac{a_2}{A_2}\sqrt{h_2} + \frac{a_4}{A_2}\sqrt{h_4} + \frac{\gamma_2 k_2}{A_2} v_2 \\
\dot{h}_3 &= -\frac{a_3}{A_3}\sqrt{h_3} + \frac{(1-\gamma_2) k_2}{A_3} v_2 \\
\dot{h}_4 &= -\frac{a_4}{A_4}\sqrt{h_4} + \frac{(1-\gamma_1) k_1}{A_4} v_1.
\end{aligned} \tag{7.64}$$

Other parameter values and the nominal steady state are given in Table 7.1. The process is considered to have 2 subsystems, one containing tanks 1 and 4 and the other containing tanks 2 and 3. Each subsystem has 2 states, 1 input and 1 upstream state. We first design a centralized MPC with quadratic objective function for each tank, and bounds on the inputs $2.5 \leq v_1, v_2 \leq 3.5$. We first decide through the simulation of centralized MPC that a receding horizon of $T = 400$ with sampling time $\delta t = 10$ is appropriate. (The computations are performed using the Python module `pyomo.dae` with an IPOPT solver [297].)

Table 7.1: Parameters and nominal steady state

Parameter	Value	Parameter	Value
A_1, A_3	28	a_1, a_3	3.145
A_2, A_4	32	a_2, a_4	2.525
γ_1	0.43	k_1	3.14
γ_2	0.34	k_2	3.29
Input	Value	Input	Value
v_1	3.15	v_2	3.15
State	Value	State	Value
h_1	12.44	h_2	13.17
h_3	4.73	h_4	4.99

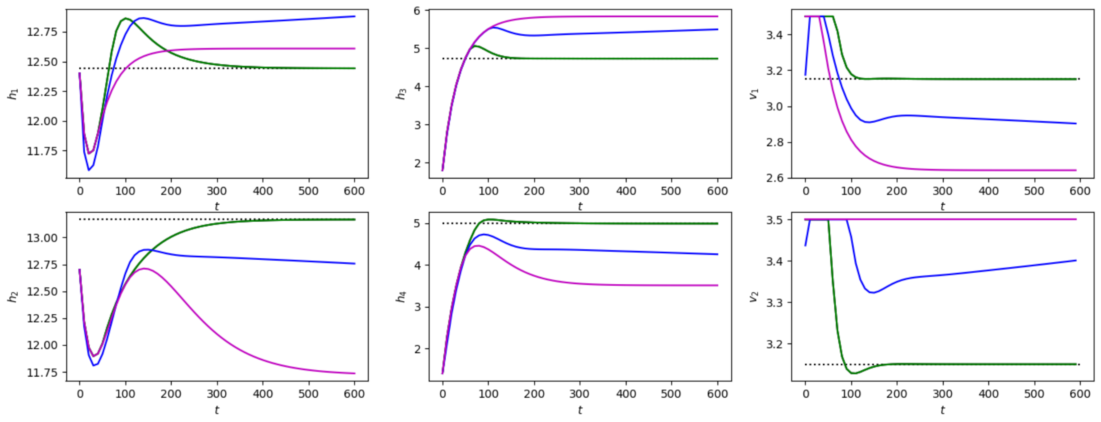


Figure 7.3: Closed-loop trajectories under traditional MPC controllers.

The closed-loop trajectories under the traditional MPC controllers, including a centralized MPC (black), a semi-centralized MPC where the inputs are iteratively updated based on predictions over the entire process (green), a decentralized MPC (blue), and a distributed MPC with only state feedforwarding among the agents (purple), are shown in Fig. 7.3. It was observed that a semi-centralized MPC based on system-wide prediction maintains the control performance, yielding trajectories overlapping with those of the centralized MPC. However, the state-feedforward distributed MPC without sufficient coordination accounting for the state interactions results in unsatisfactory control performance, whose ultimate deviation from the steady state is even larger than the decentralized MPC without any communication between the controllers.

Next we use the proposed ELLADA algorithm for distributed nonlinear MPC of the process. We first examine the basic ELL algorithm (Algorithm 7.1) by solving the

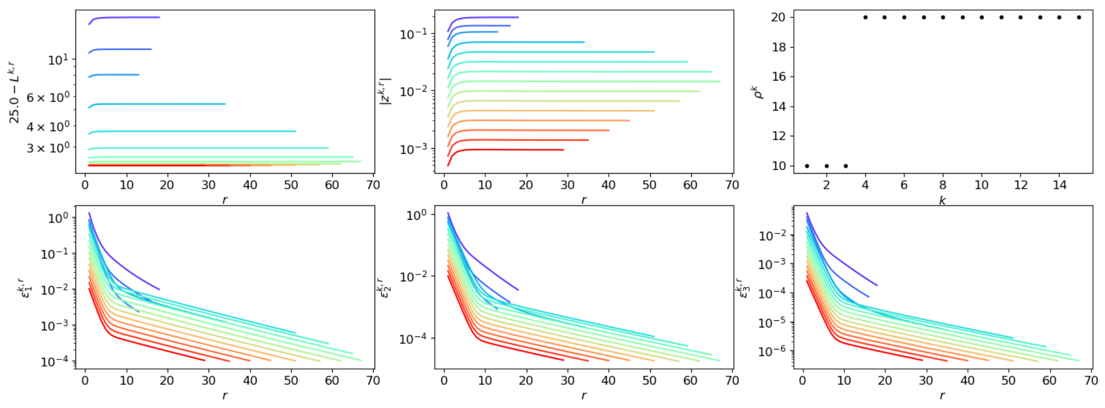


Figure 7.4: Solution results of the ELL algorithm.

corresponding distributed MPC problem at a state with $h_1 = 12.6$, $h_2 = 12.4$, $h_3 = 5.0$, $h_4 = 4.5$, where we set $\omega = 0.75$, $\gamma = 2$, $\epsilon_1^k = \epsilon_2^k = 10^{-2}/2^{k-1}$, $\epsilon_3^k = 10^{-1}/2^{k-1}$, $\epsilon_1 = \epsilon_2 = 10^{-4}$, $\epsilon_3 = 10^{-3}$ and $\overline{\lambda} = -\underline{\lambda} = 10$ (in an element-wise sense) through empirical tuning. The solution results in terms of the variation of the augmented Lagrangian $L^{k,r}$, the violations to the KKT conditions $\epsilon_{1,2,3}^{k,r}$, and penalty parameters ρ^k throughout the inner and outer iterations are presented in Fig. 7.4, where the rainbow colormap from blue to red colors stand for increasing outer iteration number. In accordance to the conclusion of Lemma 7.1, the augmented Lagrangian is monotonically decreasing in each outer iterations and remains upper bounded, which guarantees the convergence of the algorithm. Using the ELL algorithm for the afore-mentioned closed-loop MPC simulation, the resulting trajectories are found identical to those of the centralized control, which corroborates the theoretical property of the algorithm of converging to the set of stationary solutions.

With the preserved control performance of the ELL algorithm, we seek to improve its computational efficiency with the ELLA and ELLADA algorithms (Algorithms 7.2 and 7.3). In ELLA, the tolerances for approximate NLP solution are set as $\epsilon_1 = \epsilon_2 = \epsilon_4 = 10^3\epsilon_3 = 1$, $\epsilon_1^k = \epsilon_2^k = 10^3\epsilon_3^k = \epsilon_4^k = 100/2^{k-1}$, $\epsilon_4^{k,r} = 10^3\epsilon_5^{k,r} = \max(\epsilon_4^k, 40(\epsilon_1^{k,r})^2)$. The barrier constants are updated throughout outer iterations according to $\|z\|$ according to $b^{k+1} = \min(10^{-1}, \max(10^{-4}, 25(\epsilon_3^k)^2))$. Compared to ELL, the accumulated number of iterations and computational time of ELLA are reduced by over an order of magnitude. To seek for better computational performance, we apply the ELLADA algorithm, where we set $M = 10$, $\sigma = 1$, $\eta_L = \eta_{\tilde{w}} = 0.01$, $\eta_\theta = 0.5$, $\eta_w = 0.05$. This further reduces the number of iterations and computational time. These results are shown in Fig. 7.5.

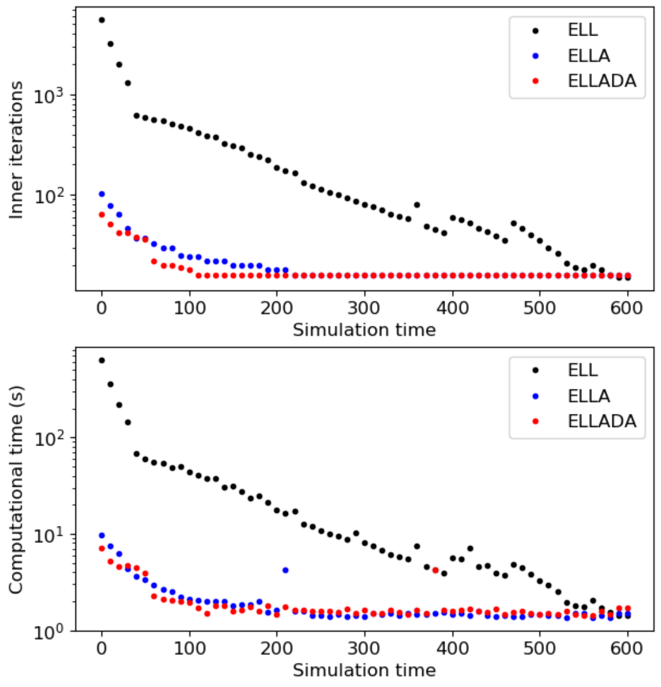


Figure 7.5: Iteration and computational time under ELL, ELLA and ELLADA algorithms.

Compared to the basic ELL algorithm, ELLADA achieves acceleration by approximately 18 times in terms of iterations and 19 times in computational time for the entire simulation time span. These improvements are more significant when the states are far from the target steady state (43 and 45 times, respectively, for the first 1/6 of the simulation). We note that the improvement from ELLA to ELLADA by using the Anderson scheme is not an order-of-magnitude one mainly because each outer iteration needs only a few number of inner iterations, leaving little space for further acceleration (e.g., for the first sampling time, 12 outer iterations including only 102 inner iterations are needed in ELLA, and in ELLADA, 61 inner iterations are needed). Under the accelerations, ELLADA returns the identical solution to the centralized optimization, thus preserving the control performance of the centralized MPC.

7.6 Conclusions and Discussions

We have proposed a new algorithm for distributed optimization allowing nonconvex constraints, which simultaneously guarantees convergence under mild assumptions and achieves fast computation. Specifically, convergence is established by adopting a two-layer architecture. In the outer layer, the slack variables are tightened using the method of multipliers, and the inequalities are handled using a barrier technique. In the inner layer, ADMM iterations are performed in a distributed and coordinated manner. Approximate NLP solution and Anderson acceleration techniques are integrated into inner iterations for computational acceleration.

Such an algorithm is generically suitable for distributed nonlinear MPC. The advantages include:

- Arbitrary input and state couplings among subsystems are allowed. No specific pattern is required a priori.
- The convergence property of the algorithm towards a stationary point is theoretically guaranteed, and its performance can be monitored throughout iterations.
- Equality-constrained NLP solvers can be used only as a subroutine. No internal modification of solvers is needed, and the choice of any appropriate solver is flexible.
- Asynchronous updates are allowed without affecting the convergence properties.
- Although motivated with a nominal optimal control problem, the algorithm could be suitable for more intricate MPC formulations such as stochastic/robust MPC or sensitivity-based advance-step MPC.

The application of the ELLADA algorithm on the distributed nonlinear MPC of a quadruple tank process has already shown its improved computational performance compared to the basic convergent Algorithms 7.1 and 7.2, and improved control performance compared to the decentralized MPC and distributed MPC without accounting for state interactions. Of course, due to the small size of the specific benchmark process, the control can be realized easily with a centralized MPC. A truly large-scale control problem is more suitable to demonstrate the effectiveness of our algorithm, and this shall be presented in the future research.

Chapter 8
A Bilevel Programming Approach to the Convergence Analysis of Control-Lyapunov Functions

8.1 Introduction

Control-Lyapunov functions are of central importance to the stability of nonlinear control systems (see, e.g., [193]). Control-Lyapunov functions can be exploited either passively, i.e., given a control policy, one finds a decaying Lyapunov function to prove closed-loop stability, or proactively using Lyapunov-based control, i.e., one designs a Lyapunov function and a control policy accordingly in order to enforce some convergence properties (see, e.g., [198]).

For example, control-Lyapunov functions have been used in model predictive control (MPC) [254] and economic model predictive control (EMPC) [98]. Typical approaches to enforce stability, such as terminal constraints or penalty terms, rely on constructing a decreasing Lyapunov function (see, e.g., [368, 225]). Different from these indirect approaches, Lyapunov-based MPC and EMPC [263, 59] explicitly guarantee the asymptotic convergence of the closed-loop system with a predetermined control-Lyapunov function.

The convergence analysis of control-Lyapunov functions is the basis of Lyapunov-based control. The following problems emerge to this end: (i) Given any Lyapunov function and any domain, what is the fastest convergence rate that this Lyapunov function can achieve in this domain? (ii) Given any Lyapunov function and any convergence rate, what is the domain on which this Lyapunov function can achieve this convergence rate? The answers to these two questions help examine the feasiblity of Lyapunov-based control and tune a control-Lyapunov function. However, this systematic analysis has been addressed only for input-affine nonlinear systems under norm-type input constraints [96, 263]. In a different approach, the domain of attraction can be estimated using sum-of-squares (SOS) programming for polynomial systems under unconstrained polynomial feedback control policies [175, 402, 61].

In this work, we provide a novel optimization framework for the estimation of convergence rate and domain of attraction of Lyapunov functions, which is applicable to

©2019 IEEE. Reprinted, with permission, from W. Tang, P. Daoutidis. A bilevel programming approach to the convergence analysis of control-Lyapunov functions. *IEEE Transactions on Automatic Control* (Volume: 64, Issue: 10, Oct. 2019). DOI:10.1109/TAC.2019.2892386.

systems with more general forms of nonlinear dynamics in the presence of operational constraints on the control inputs, as long as a convexity assumption is satisfied. Specifically, these two estimation problems are formulated as bilevel programs [65], which can be converted to and solved as mixed-integer nonlinear programs (MINLPs).

Remark 8.1. *Our work is motivated by the flexibility analysis of process systems design in chemical engineering [138, 88] (also see [137] for a recent review), which considers the following problems:*

1. *Flexibility test problem – Given any design and any variation range of uncertain parameters, does the design stay within the feasible region, and if not, how far is it away from the feasibility boundary?*
2. *Flexibility index problem – Given any design and the feasibility boundary, what is the magnitude of parameter variation under which the design remains feasible?*

The convergence rate estimation and domain of attraction estimation problems are the counterparts of the feasibility test problem and the feasibility index problem, respectively, with a conceptual correspondence between the feasibility of process design and the decay rate of the control-Lyapunov function, and between the parameter uncertainty set and the domain of attraction. The optimization techniques to be used in this paper – the transformation of bilevel programs into MINLPs using KKT conditions and active set method – are well established in the context of flexibility analysis.

8.2 Preliminaries

Consider a continuous, time-invariant control system:

$$\dot{x} = f(x, u) \tag{8.1}$$

in which $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are the state and control variables, respectively. We assume that admissible control inputs u obey some operational constraints with the system states x, written in the form of

$$\pi(x, u) \leq 0, \;\; \rho(x, u) = 0, \tag{8.2}$$

where π and ρ are vector-valued functions. f is Lipschitz continuous and $0 = f(0, 0)$, i.e., the origin is an equilibrium point towards which we aim to drive. For system (8.1), any admissible control policy $u = u(x)$ gives a closed-loop system

$$\dot{x} = f(x, u(x)). \tag{8.3}$$

A Lyapunov function is a notion of central importance for stability analysis of the closed-loop system. The relation between closed-loop stability and the decay of a positive definite Lyapunov function is established by the following fact (see, e.g., Theorems 4.8–4.10 in §4.5 of [193]).

Fact 8.1. *Let $D \subseteq \mathbb{R}^n$ be a domain containing the origin, and $x \mapsto u(x)$ be a given control policy on D. If there exists a continuously differentiable function (called Lyapunov function) $V : D \to \mathbb{R}$ satisfying*

$$w_1(x) \leq V(x) \leq w_2(x), \quad \frac{dV(x)}{dt} = \nabla V(x)^\top f(x, u(x)) \leq 0 \tag{8.4}$$

for all $x \in D$, in which w_1 and w_2 are continuous positive definite functions on D, then $x = 0$ is stable. If the latter inequality is enhanced by another positive definite function $w_3(x)$ on D:

$$\frac{dV(x)}{dt} \leq -w_3(x), \tag{8.5}$$

then $x = 0$ is asymptotically stable. Furthermore, if power functions of the same order can be found on D for $w_1(x), w_2(x), w_3(x)$, i.e. $w_i(x) = k_i\|x\|^a, k_i > 0, i = 1, 2, 3$ and $a > 0$, then $x = 0$ is exponentially stable, with a decay rate of the Lyapunov function

$$\frac{d \ln V(x)}{dt} = \frac{1}{V(x)} \frac{dV(x)}{dt} \leq -\frac{k_3}{k_2} < 0. \tag{8.6}$$

Asymptotic stability is a basic requirement of control[1]. However, it can not be automatically guaranteed for any admissible control. For example, for nonlinear MPC and EMPC, asymptotic stability holds only if the system satisfies a certain dissipativity condition [281]. To guarantee stability, a control-Lyapunov function can be artificially appointed, and a constraint on its decay

$$\frac{dV(x)}{dt} = \nabla V(x)^\top f(x, u) \leq -\sigma V(x) \tag{8.7}$$

[1]Although asymptotic stability is not always necessary, since a periodical or cyclical operation can be economically more favorable than a steady-state one. At minimum, the system has to be confined within a certain operational region for the sake of safety.

imposed for some $\sigma > 0$. In this way, for a control policy $u = u(x)$, as long as the constraints, including the operational contraints (8.2) and the Lyapunov constraints (8.7), are feasible for all times starting from any point on a given domain D, the system is stabilizable from D.

To guarantee the feasibility of Lyapunov-based control, one needs to estimate the achievable decay rate σ on a given domain D, or the size of the domain D on which a given decay rate σ is achievable, in advance of imposing the Lyapunov constraint (8.7). For nonlinear input-affine systems under operational constraints on some type of norm of the inputs, this problem has been well addressed [263] and a specific explicit control policy can be constructed[2]. In the remainder of this work, we will consider a broader class of nonlinear control systems, and propose a bilevel programming approach to the convergence analysis of control-Lyapunov functions.

8.3 Estimation of Convergence Rate

We consider the estimation of the convergence rate of any given control-Lyapunov function V on any given domain. We first introduce the notion of logarithmic decay rate:

Definition 8.1. *The (instantaneous) logarithmic decay rate of the Lyapunov function V when the state is at point x and the control value is u is*

$$\phi(V, x, u) = -\frac{\nabla V(x)^\top f(x, u)}{V(x)} = -\frac{d \ln V(x)}{dt}. \tag{8.8}$$

For any given x, by taking the maximum over all values of u satisfying the operational constraints, we obtain the fastest attainable decay rate at x. Since the Lyapunov function value decays at different rates at different points x in the domain D, by taking the minimum of this rate over all $x \in D$, we obtain a decay rate of V that can be achieved for any $x \in D$ by choosing a suitable u. We thus define the concept of uniformly achievable logarithmic decay rate $\sigma(V, D)$:

[2]Given any $\sigma > 0$, if (8.7) is satisfiable, it is always satisfied by the control policy constructed by [230], $u_{\text{L-S}}(x)$. This policy can be implemented directly as a stabilizing control policy, or as the "baseline" policy to guarantee a decay rate not slower than that under $u_{\text{L-S}}(x)$:

$$\frac{dV(x)}{dt} \leq \nabla V(x)^\top f(x, u_{\text{L-S}}(x)) \leq -\sigma V(x),$$

which is a typical approach in Lyapunov-based MPC and EMPC.

Definition 8.2. *The uniformly achievable logarithmic decay rate of V in D is*

$$\sigma(V, D) = \min_{x \in D} \max_{u} \ \phi(V, x, u) \quad \text{s.t. } \pi(x, u) \leq 0, \ \rho(x, u) = 0. \tag{8.9}$$

Remark 8.2. *When D contains the origin, the origin (with $V(0) = 0$) should be excluded from D to have $\phi(V, x, u)$ well defined. This will be implicit in the remaining text. However, for the practical application of the method (i.e., the solution of the corresponding optimization problems), we will need to remove a finite, sufficiently small open neighborhood of the origin so as to have well-posed optimization formulations.*

(8.9) considers the worst case among all $x \in D$ of the achievable logarithmic decay rate by choosing the optimal input u. With this definition, we specify our first problem as the determination of the uniformly achievable logarithmic decay rate of any Lyapunov function V in any domain D:

Problem 8.1. *Given V and D, find $\sigma(V, D)$, i.e., solve the bilevel program* (8.9).

The uniformly achievable decay rate $\sigma(V, D)$ specifies an upper bound for the feasible exponential decay rate of Lyapunov-based control. We consider the typical case when D is a sublevel set of the Lyapunov function V, i.e., $D = \{x | V(x) \leq \delta\}$ for some $\delta > 0$.

- If $\sigma(V, D) > 0$, then starting from any point in D, the control-Lyapunov function can reach a positive logarithmic decay rate. The trajectory is thus confined in D (i.e., D is a positive invariant set of the system), and the origin is exponentially stabilizable at a rate of $O(\mathrm{e}^{-\sigma(V,D)t})$. A Lyapunov-based control can be implemented with the corresponding constraint written as in (8.7), where σ is the prescribed logarithmic decay rate in $(0, \sigma(V, D)]$. If σ is chosen to be close to $\sigma(V, D)$, then the convergence rate is expected to be "radical", with a possible compromise in the control performance since larger control effort may be needed. By choosing a smaller σ, a "milder" convergence rate is allowed, which can be favorable in the context of optimal control which accounts for both the state trajectory and the control effort.

- If $\sigma(V, D) \leq 0$, then there exists some point $x \in D$ where any admissible control will cause the Lyapunov function value to stagnate or increase. It does not necessarily follow that the trajectory under any control will diverge. However, $\sigma(V, D) \leq 0$ implies that any Lyapunov-based control using V will have an infeasible region.

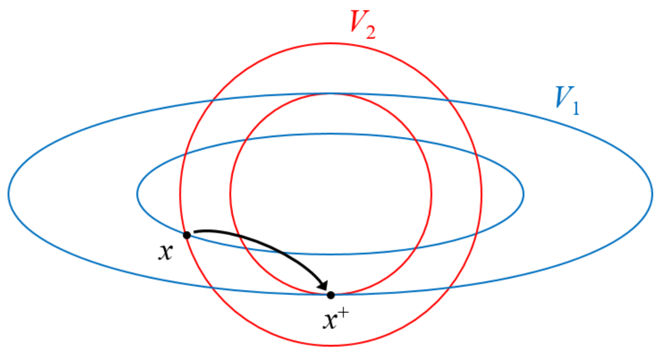


Figure 8.1: Different decay rates for different Lyapunov functions.

We note that different choices of V affect the value of $\sigma(V, D)$. Fig. 8.1 illustrates the change of two different Lyapunov functions V_1, V_2, whose contours are concentric ellipsoids (in blue and red, respectively), when the state moves from point x to point x^+. It is seen that Lyapunov function V_1 with the blue contours increases, while V_2 decreases. If at point x, feasible control can only allow trajectories towards x^+, a Lyapunov-based control strategy based on V_1 becomes infeasible while a V_2-based control is still feasible. From the uniformly achievable decay rates of different Lyapunov functions, one may tune the parameters in the Lyapunov function to find suitable contours with a reasonable shape (ratio between the vertical and horizontal intercepts).

8.4 Estimation of Domain of Attraction

We proceed to consider the second problem of estimating the domain on which a given Lyapunov function $V(x)$ achieves a given convergence rate of σ. We define the σ-domain of attraction as the largest domain on which the system states can be attracted to the origin with a Lyapunov decay rate of σ under an admissible control policy. In other words, the σ-domain of attraction is the largest region in which there exists a control $u(t)$ satisfying (8.7) for any time $t \geq 0$ as long as the initial point is in this domain.

Definition 8.3. *The σ-domain of attraction of Lyapunov function V, $D(V, \sigma)$ is the largest one among (or equivalently, the union of all) the domains D such that there exists a control policy $u(x)$ satisfying $\pi(x, u(x)) \leq 0$, $\rho(x, u(x)) = 0$ and $\phi(V, x, u(x)) \geq \sigma$ for all $x \in D$, and making D positive invariant, i.e., any trajectory under $u(x)$ starting in D remains in D. Specifically, $D(V, 0)$ is the largest positive invariant set in which $V(x)$*

is non-increasing. $D(V,0)$ *is a subset (underestimate) of the domain of attraction in the common sense.*

Hence if we have any domain D satisfying $\sigma(V,D) \geq \sigma > 0$ and being positive invariant under some admissible control policy which makes V to decay at a logarithmic rate not slower than σ, then $D \subseteq D(V,\sigma)$. In this sense, the problem of determining $D(V,\sigma)$ is the inverse problem of Problem 8.1. However, finding $D(V,\sigma)$ is computationally difficult, with only limited approaches (e.g., grid-meshing and expansion) available for low-dimensional systems (see, e.g., [160]). For this purpose, we consider a one-dimensional parameterization of the sublevel sets of Lyapunov functions $D_\delta = \{x|V(x) \leq \delta\}$, $\delta \geq 0$, satisfying $D_\delta \subset D_{\delta'}$ if $\delta < \delta'$. We note that D_δ is necessarily positive invariant as long as the Lyapunov function decays. If $\sigma(V,D_\delta) \geq \sigma$, then $D_\delta \subseteq D(V,\sigma)$.

Since the convergence rate $\sigma(V,D_\delta)$ is obtained through a minimization over $x \in D_\delta$ in (8.9), we have $\sigma(V,D_\delta) \geq \sigma(V,D_{\delta'})$ if $\delta < \delta'$, i.e., $\sigma(V,D_\delta)$ is a non-increasing function of δ. Therefore, the *best estimation of the σ-domain of attraction under the parameterization* $\{D_\delta|\delta \geq 0\}$ is determined by the largest δ satisfying $\sigma(V,D_\delta) \geq \sigma$, or equivalently, the smallest value of δ such that $\sigma(V,D_\delta) \leq \sigma$:

$$\delta(V,\sigma) = \max\{\delta|\sigma(V,D_\delta) \geq \sigma\} = \min\{\delta|\sigma(V,D_\delta) \leq \sigma\}. \tag{8.10}$$

According to the definition of the decay rate in (8.8), the solution of $\delta(V,\sigma)$ is

$$\begin{aligned} \delta(V,\sigma) = \min\ & \delta \\ \text{s.t. } & \min_{x\in D_\delta} \max_u \ \phi(V,x,u) \leq \sigma \\ & \qquad\qquad \text{s.t. } \pi(x,u) \leq 0,\ \ \rho(x,u) = 0. \end{aligned} \tag{8.11}$$

For any feasible solution δ to (8.11), there exists an $x \in D_\delta$ such that $\max_u \phi(V,x,u) \leq \sigma$. For any δ' smaller than the minimum in (8.11), any $x \in D_{\delta'} \subset D_\delta$ satisfies $\max_u \phi(V,x,u) > \sigma$. Hence there must exist a $x \in D_\delta$ with $\max_u \phi(V,x,u) = \sigma$. Therefore (8.11) is equivalent to

$$\begin{aligned} \delta(V,\sigma) = \min_{x\in D_\delta}\ \delta \quad \text{s.t. } \sigma = & \max_u \ \phi(V,x,u) \\ & \text{s.t. } \pi(x,u) \leq 0,\ \ \rho(x,u) = 0. \end{aligned} \tag{8.12}$$

The equality $\max_u \phi(V,x,u) = \sigma$ defines a surface of states parameterized by σ, as shown in the blue curves in Fig. 8.2, whose interior is the domain where a convergence rate of σ can be achieved. $\delta(V, \sigma)$ is the smallest δ such that the boundary of D_δ intersects with such a surface. Hence $D_{\delta(V,\sigma)}$ is the best approximation to the σ-domain of attraction among the regions of the prescribed “shape” (specified by the parametrization $\{D_\delta | \delta \geq 0\}$). Therefore we specify the estimation of domain of attraction as the following problem:

Problem 8.2. *Given V and $\sigma > 0$, find $\delta(V, \sigma)$, i.e., solve the bilevel program* (8.12).

If D_δ are chosen as Lyapunov sublevel sets, then $D_{\delta(V,\sigma)}$ is the largest region in which the Lyapunov-based control (8.7) is guaranteed to be recursively feasible. By choosing an increasing sequence $\sigma_1 > \sigma_2 > \cdots > 0$, a nested collection of domains of attraction $D_{\delta_1} \subset D_{\delta_2} \subset \cdots \subset \mathbb{R}^n$ can be determined. A multi-phase Lyapunov-based control can be implemented instead of (8.7):

$$(\nabla V(x))^\top f(x, u) \leq -\sigma_i' V(x), \text{ if } x \in D_{\delta_i} \setminus D_{\delta_{i-1}} \ (i = 1, 2, \dots) \tag{8.13}$$

in which $\sigma_i' \in (0, \sigma_i]$ for any i, and D_{δ_0} is the origin or a safety region inside which Lyapunov-based control is not needed.

Remark 8.3. *Here we defined $\phi(V, x, u)$ based on an exponential decay of the Lyapunov function. It is possible, depending on the specific form of the system, to examine the convergence in other forms if the Lyapunov function can not decay exponentially. By modifying the definition in* (8.8) *as*

$$\phi(V, x, u) = -\nabla V(x)^\top f(x, u)/V(x)^\alpha, \tag{8.14}$$

characterizations of exponential, polynomial, linear and sublinear convergence rates can be obtained for $\alpha = 1$, $1 < \alpha < 2$, $\alpha = 2$, and $\alpha > 2$, respectively. Consider Problem 8.1 when D contains the origin. If the optimal value of $\sigma(V, D)$ is a very small positive number obtained at a point as close as possible to the origin, then we infer that V can not decay at the rate corresponding to the current α, and we need to increase α to examine if V decreases at a lower rate.

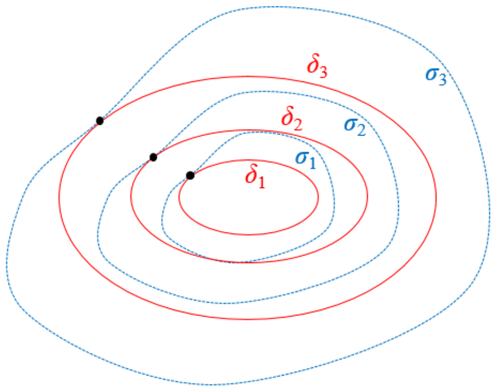


Figure 8.2: Different domains on which different decay rates can be achieved.

8.5 Solution of Bilevel Programs

The most common treatment for solving bilevel programs is to remove the inner level by deriving its Karush-Kuhn-Tucker (KKT) conditions [65]. For this purpose, we assume that the system satisfies (or is rendered to satisfy through a coordinate transformation) the following input-convexity conditions:

Assumption 8.1. *f is continuously differentiable and convex in u. $\pi(x, u)$ (if exists) is continuously differentiable and convex in u. $\rho(x, u)$ (if exists) is affine in u.*

The assumption guarantees that the maximization of $\phi(x, u)$ with respect to u, i.e., the inner problem of (8.9) and (8.12), is a convex optimization problem for which the KKT conditions are sufficient for a global optimum (see, e.g., §5.5.3 of [45]). This can be stated as the following proposition.

Proposition 8.1. *Under Assumption 8.1, the bilevel programs* (8.9) *and* (8.12)*, respectively, are equivalent to*

$$\begin{aligned}\sigma(V, D) = \min_{x \in D} \ & \sigma = \phi(V, x, u) \\ \text{s.t. } & 0 = -\frac{\partial \phi}{\partial u}(V, x, u) + \lambda^\top \frac{\partial \pi}{\partial u}(x, u) + \mu^\top \frac{\partial \rho}{\partial u}(x, u) \\ & 0 = \lambda \circ \pi(x, u), \ \ 0 \le \lambda, \ \ \pi(x, u), \ \ 0 = \rho(x, u)\end{aligned} \tag{8.15}$$

and

$$\begin{aligned}\delta(V,\sigma) = \min\ \delta \quad \text{s.t. }\ & \sigma = \phi(V,x,u),\ \ x \in D_\delta \\ & 0 = -\frac{\partial \phi}{\partial u}(V,x,u) + \lambda^\top \frac{\partial \pi}{\partial u}(x,u) + \mu^\top \frac{\partial \rho}{\partial u}(x,u) \\ & 0 = \lambda \circ \pi(x,u),\ \ 0 \le \lambda,\ \ 0 \ge \pi(x,u),\ \ 0 = \rho(x,u).\end{aligned} \tag{8.16}$$

In the above problems, λ and μ are the vectors of dual variables associated with the constraints $\pi(x,u)$ and $\rho(x,u)$, respectively, and $\circ$ stands for element-wise multiplication.

Since the $\circ$ operation involves a discrete choice of active constraints, the active set method [138, 88] can be applied to convert such complementary slackness constraints into constraints with integer variables. Let $y_j \in \{0,1\}$ indicate whether π_j is an active constraint, and s_j be the slack variable corresponding to π_j, $j = 1,\ldots,J$. Then we have

$$\begin{aligned}\sigma(V,D) = \min_{x\in D}\ & \sigma = \phi(V,x,u) \\ \text{s.t. }\ & 0 = -\frac{\partial \phi}{\partial u}(V,x,u) + \lambda^\top \frac{\partial \pi}{\partial u}(x,u) + \mu^\top \frac{\partial \rho}{\partial u}(x,u) \\ & 0 \ge \pi(x,u) + s,\ \ 0 = \rho(x,u) \\ & 0 \ge \lambda - My,\ \ 0 \ge s - M(1-y) \\ & 0 \le \lambda,\ \ 0 \le s,\ \ y \in \{0,1\}^J\end{aligned} \tag{8.17}$$

where M is an effective upper bound for the dual variables and slack variables. It can be observed that if $y_j = 1$, i.e., the j-th constraint is active ($\pi_j(x,u) = 0$), s_j has to be 0 and λ_j can be any nonnegative number smaller than M. If $y_j = 0$, i.e., the j-th constraint is inactive ($\pi_j(x,u) < 0$), λ_j has to be 0 and $0 \le s_j \le M$. The single level formulations become mixed integer nonlinear programming (MINLP) problems, which can be tackled using global optimization solvers. Similarly, the estimation of σ-domain of attraction (8.16) can be written as

$$\begin{aligned}\delta(V,\sigma) = \min\ \delta \quad \text{s.t. }\ & \sigma = \phi(V,x,u),\ \ x \in D_\delta \\ & 0 = -\frac{\partial \phi}{\partial u}(V,x,u) + \lambda^\top \frac{\partial \pi}{\partial u}(x,u) + \mu^\top \frac{\partial \rho}{\partial u}(x,u) \\ & 0 \ge \pi(x,u) + s,\ \ 0 = \rho(x,u) \\ & 0 \ge \lambda - My,\ \ 0 \ge s - M(1-y) \\ & 0 \le \lambda,\ \ 0 \le s,\ \ y \in \{0,1\}^J\end{aligned} \tag{8.18}$$

Now Problems 8.1 and 8.2 are restated as follows, respectively, given that Assumption 8.1 holds:

Problem 8.3. *Given V and D, obtain $\sigma(V, D)$ by solving the MINLP problem* (8.17).

Problem 8.4. *Given V and σ, obtain $\delta(V, \sigma)$ by solving the MINLP problem* (8.18).

Remark 8.4. *The transformation of bilevel programs using KKT conditions for the inner problem may lead to suboptimal solutions if the inner problem is non-convex, since in this case the KKT conditions may not be sufficient for global optimality [45]. In this case, [141] used convex relaxations of the inner problem. [266] developed a heuristic approach of bound tightening for bilevel programs with both inner and outer problems being non-convex. There also exist other algorithms under respective assumptions. In general, global algorithms for nonlinear bilevel programming, with the potential to solve our problems without Assumption 8.1, remain to be explored.*

Remark 8.5. *The transformed bilevel programs* (8.17) *and* (8.18) *are generalized disjunctive programs (GDPs) [139] with logical constraints:*

$$\begin{bmatrix} y_j = 1 \\ s_j = 0 \end{bmatrix} \underline{\bigvee} \begin{bmatrix} y_j = 0 \\ \lambda_j = 0 \end{bmatrix}, \quad j = 1, \dots, J. \tag{8.19}$$

where $\veebar$ is an "exclusive or" operation, and brackets with two propositions represent an "and" operation. In our discussion we have used a big-M formulation (with the help of an upper bound parameter M) to convert the GDPs into MINLPs for their solution. Another conversion method, called hull reformulation [17], is also often applied, especially for large-scale problems, since it tends to reduce the computational effort of branch-and-bound procedures during global optimization while needing more variables and constraints (see, e.g., [465]).

Remark 8.6. *A special case satisfying Assumption 8.1 is input-affine systems $f(x, u) = f_0(x) + g(x)u$ with bounds on the input norm $\|u\|_\alpha \leq M(x)$ for some $\alpha \in [1, \infty]$. The maximization of $\phi(V, x, u)$ with respect to u can be obtained through Hölder's inequality without resorting to KKT conditions. Specifically,*

$$\max_{\|u\|_\alpha \leq M(x)} \phi(V, x, u) = \frac{-\nabla V(x)^\top f_0(x) + M(x) \|\nabla V(x)^\top g(x)\|_\beta}{V(x)}, \tag{8.20}$$

where $\beta \in [1, \infty]$ satisfies $1/\alpha + 1/\beta = 1$. Hence the estimation of convergence rate and domain of attraction can be expressed as single layer optimization problems:

$$\sigma(V, D) = \min_{x \in D} \frac{-\nabla V(x)^\top f_0(x) + M(x)\|\nabla V(x)^\top g(x)\|_\beta}{V(x)} \tag{8.21}$$

and

$$\delta(V, \sigma) = \min \ \ \delta \quad \text{s.t. } \sigma = \frac{-\nabla V(x)^\top f_0(x) + M(x)\|\nabla V(x)^\top g(x)\|_\beta}{V(x)}, \ \ x \in D_\delta. \tag{8.22}$$

These formulations have been shown in the literature (see, e.g., [263]) for $M(x) = 1$, $\alpha = 2$. This suggests that our bilevel programming formulation is more general.

8.6 Example: An Input-Nonaffine System

Consider the following system adapted from [26]:

$$\begin{aligned}
\dot{x}_1 &= \left(1 + \frac{x_1^2}{\|x\|^2}\right)(x_2 - x_1) - \frac{1}{1 + x_3^2} u_1 \cos u_2 \\
\dot{x}_2 &= \left(1 + \frac{x_1^2}{\|x\|^2}\right)(x_1 + x_3 - 2x_2) - (x_2 + \tanh x_2) \\
\dot{x}_3 &= \left(1 + \frac{x_1^2}{\|x\|^2}\right)(x_2 - x_3) - \frac{1}{1 + x_3^2} u_1 \sin u_2
\end{aligned} \tag{8.23}$$

For the quadratic Lyapunov function $V(x) = \frac{1}{2}\|x\|^2$, we have

$$\begin{aligned}
\phi(V, x, u) = & \frac{2}{\|x\|^2} \left\{ \frac{u_1(x_1 \cos u_2 + x_3 \sin u_2)}{1 + x_3^2} + x_2 \tanh x_2 \right. \\
& \left. + \left(1 + \frac{x_1^2}{\|x\|^2}\right)\left[(x_1 - x_2)^2 + (x_3 - x_2)^2\right] \right\}.
\end{aligned} \tag{8.24}$$

For the case where $u_1 \in [0, 1]$, $u_2 \in [0, 2\pi]$ and $D_R = \{x | V(x) \leq \frac{1}{2}R^2\} = \{x | \|x\|^2 \leq R^2\}$ (where R is a parameterization for difference choices of D in Problem 8.3), the maximization with respect to u can be obtained analytically without resorting to the

KKT transformation. Since $u_1(x_1 \cos u_2 + x_3 \sin u_2) \leq \sqrt{x_1^2 + x_3^2}$, we have

$$\max_u \phi(V, x, u) = \frac{2}{\|x\|^2} \left\{ \frac{\sqrt{x_1^2 + x_3^2}}{1 + x_3^2} + x_2 \tanh x_2 + \left(1 + \frac{x_1^2}{\|x\|^2}\right) \left[(x_1 - x_2)^2 + (x_3 - x_2)^2\right] \right\}. \tag{8.25}$$

The convergence rate $\sigma(V, D_R)$ is estimated by minimizing the above expression subject to $x \in D_R$.

However, if there exist additional constraints on the input components acting on the states: $-L \leq u_1 \cos u_2 \leq L$ and $-L \leq u_1 \sin u_2 \leq L$, then the maximized $\phi(V, x, u)$ no more has the above analytical form. Instead, we will need the KKT conditions to resolve the inner problem. We first use a coordinate transformation $v_1 = u_1 \cos u_2$, $v_2 = u_1 \sin u_2$ which converts the system into an input-affine form with input $v = (v_1, v_2)$ constrained by the intersection of the unit disk and a square of side length $2L$: $v \in \{(v_1, v_2) | -L \leq v_1, v_2 \leq L, v_1^2 + v_2^2 \leq 1\}$, and thus transforms the inner problem into a convex one to satisfy Assumption 8.1:

$$\begin{aligned} v = \arg\min_v \ & (-x_1 v_1 - x_3 v_2) \\ \text{s.t.} \ & v_1 - L \leq 0, \ \ -v_1 - L \leq 0 \\ & v_2 - L \leq 0, \ \ -v_2 - L \leq 0, \ \ v_1^2 + v_2^2 - 1 \leq 0. \end{aligned} \tag{8.26}$$

This leads to the KKT conditions of the form

$$\begin{aligned} & -x_1 + \lambda_1 - \lambda_2 + 2\lambda_5 v_1 = 0, \ \ -x_3 + \lambda_3 - \lambda_4 + 2\lambda_5 v_2 = 0 \\ & v_1 - L + s_1 = 0, \ \ -v_1 - L + s_2 = 0 \\ & v_2 - L + s_3 = 0, \ \ -v_2 - L + s_4 = 0, \ \ v_1^2 + v_2^2 - 1 + s_5 = 0 \\ & \lambda_i - M y_i \leq 0, \ \ s_i - M(1 - y_i) \leq 0, \ \ i = 1, \ldots, 5 \\ & \lambda_i, s_i \geq 0, \ \ y_i \in \{0, 1\}, \ \ i = 1, \ldots, 5. \end{aligned} \tag{8.27}$$

According to (8.17), $\sigma(V, D_R)$ can be obtained through global optimization of the following MINLP problem:

$$\begin{aligned} \min \ \sigma = & \frac{2}{\|x\|^2} \left\{ \frac{x_1 v_1 + x_3 v_2}{1 + x_3^2} + x_2 \tanh x_2 + \left(1 + \frac{x_1^2}{\|x\|^2}\right) \left[(x_1 - x_2)^2 + (x_3 - x_2)^2\right] \right\} \\ \text{s.t.} \ & (8.27), \ \ \|x\|^2 \leq R^2. \end{aligned} \tag{8.28}$$

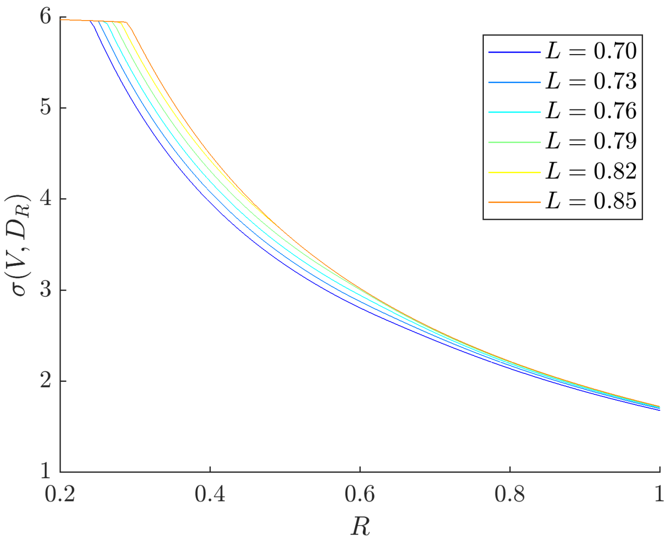


Figure 8.3: Convergence rate of $V(x)$ in D_R under different input bounds L.

We solve the above problem with the BARON solver [420] on GAMS 25.0.2 for a series of R values ranging from 0.2 to 1 under different parameters of L. The results are shown in Fig. 8.3. The convergence rate stays positive, i.e., there always exists a feasible control to stabilize the system from anywhere in the state space. On the other hand, we observe that tighter bounds on v_1 and v_2 (lower values of L) slow down the convergence rate of V.

8.7 Conclusion

The application of Lyapunov-based control requires a generic method of estimating convergence rate and domain of attraction of control-Lyapunov functions. In this work, we proposed bilevel optimization formulations for the convergence rate and domain of attraction estimation, which are amenable to a KKT condition treatment and the active set method that transforms the problems into single-layer MINLPs. This approach to the convergence analysis of control-Lyapunov functions can be applied to any system with input-nonaffinity or general input constraints satisfying a convexity assumption.

Chapter 9
A Stochastic Programming Approach to the Optimal Design of Control-Lyapunov Functions

9.1 Introduction

Lyapunov-based control is a proactive approach to enforce closed-loop convergence based on a pre-determined positive definite state-dependent function, called control-Lyapunov function [198]. Given the control-Lyapunov function and its desirable decay rate over time, one may construct a control policy, e.g., according to the formula of [230], that explicitly guarantees the decay of the control-Lyapunov function, or incorporate the decay as an additional constraint to the originally designed control strategy. For example, Lyapunov-based model predictive control and economic model predictive control formulations have been proposed to directly enforce stability [263, 234, 150], differently from typical techniques for stabilization such as imposing terminal constraints or adding penalty terms [368].

Despite the wide application of Lyapunov-based control, systematic approaches to the design of control-Lyapunov functions are scarce. Typically, special forms or restrictive conditions satisfied by the system dynamics are needed to construct control-Lyapunov functions [371, 247]. For polynomial systems under unconstrained polynomial feedback control laws, sum-of-squares (SOS) programming has been proposed for optimizing control-Lyapunov functions as SOS polynomials [175, 402, 61]. Moreover, since a cost or performance index is typically associated with control policies, the choice of control-Lyapunov function is expected to have a significant impact on the cost of the resulting Lyapunov-based control. Therefore, it is desirable that the control-Lyapunov function be optimally designed so that the resulting control cost is minimized. This *optimal design problem of control-Lyapunov functions*, to our best knowledge, has not been considered for control systems.

In this work, we formulate the optimal design of control-Lyapunov functions as a *two-stage stochastic programming* problem, given a specified decay rate of the control-Lyapunov function and a domain of interest in the state space. The two-stage stochasticity originates from the expression of the cost associated with the Lyapunov function (first-stage decision) as the expected cost of the resulting Lyapunov-based control (second-stage decision) from different states (uncertainty) on the domain. We consider two different stochastic programming formulations, and propose corresponding algorithms for their solution, respectively.

- In the first case, we consider the accumulated control cost starting from the initial state until convergence to the origin (and hence the control is an infinite-horizon optimal policy), and the expected cost by averaging the accumulated cost over the domain of interest. A sample average approximation is used to convert the stochastic programming problem into a deterministic one, expressed as multiple dynamic programming problems linked by the control-Lyapunov function. For its solution, we adopt a *decomposition-based optimization algorithm* that partition the monolithic problem into subproblems, each corresponding to a single sample.
- In the second case, the control cost is defined based only on the instantaneous states and inputs. Without a prediction as in the first case, the trajectories can only be guaranteed to remain in the smallest Lyapunov sublevel set covering the domain of interest, and hence the cost associated with the Lyapunov function is defined as the averaged instantaneous cost within such a Lyapunov sublevel set. For the solution of such a problem, we develop an online optimization algorithm, called *stochastic proximal algorithm*, which allows the Lyapunov function to be updated continually with new samples of uncertainty.

The remainder of this chapter is organized as follows. In Section 9.2, the preliminaries of Lyapunov stability and Lyapunov-based control are introduced. The stochastic programming framework and the two specific formulations of optimally designing Lyapunov functions are established in Section 9.3. The algorithms for solving these stochastic programs are described in Section 9.4. A case study on a chemical reactor example is shown in Section 9.5. Conclusions are made in Section 9.6.

Remark 9.1. *This work is motivated by the works on flexibility analysis and flexible process design in the field of chemical engineering [137], which consider the following three problems:*

1. *Flexibility test problem – Given the design of the chemical process and the range of uncertain parameters, can the operations of the designed process always stay feasible (satisfy certain constraints)?*
2. *Flexibility index problem – Given the process design and the constraints for feasible*

operations, what is the magnitude of variation of the parameters, under which the operations remain feasible?

3. *Flexible design problem – Given the range of the uncertain parameters and the constraints for feasible operations, what is the design that always satisfy the constraints under the uncertainties, and minimizes the expected economic cost?*

In the previous chapter [410], we studied the convergence rate estimation and domain of attraction estimation problems as analogues of the flexibility test and flexibility index problems, respectively, with a conceptual correspondence between the feasibility of process operations and the decay rate of control-Lyapunov functions, and between the parameter uncertainty set and the domain of attraction. The optimal Lyapunov function design problem to be addressed in this work is an analogue of the flexible design problem. The sample average approximation technique for solving stochastic programs to be used in this chapter has been commonly employed in the works of flexible process design in the name of multi-period design [430, 312].

9.2 Preliminaries

We consider a nonlinear control system:

$$\dot{x} = f(x, u) \tag{9.1}$$

in which $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ represent the states and control inputs, respectively. f is Lipschitz with the origin being an equilibrium point ($f(0, 0) = 0$). The goal is to stabilize the system (9.1) towards the origin. We also assume that any admissible control policy $u = u(x)$ should be subject to some operational constraints:

$$\pi(x, u) \leq 0. \tag{9.2}$$

For any control policy $u = u(x)$, the following theorem relates the closed-loop stability to the existence of a Lyapunov function [193, Section 4.5].

Fact 9.1. *Let $D \subseteq \mathbb{R}^n$ be a domain containing the origin, and $x \mapsto u(x)$ be a given control policy on D. If there exists a continuously differentiable function, called a Lyapunov function, $V : D \to \mathbb{R}$, such that there exist some continuous positive definite functions*

w_1, w_2 on D satisfying

$$w_1(x) \leq V(x) \leq w_2(x), \quad \frac{dV(x)}{dt} = \nabla V(x)^\top f(x, u(x)) \leq 0 \tag{9.3}$$

for all $x \in D$, then $x = 0$ is stable. If the latter inequality is enhanced by a positive definite function $w_3(x)$ on D:

$$\frac{dV(x)}{dt} \leq -w_3(x), \tag{9.4}$$

then $x = 0$ is asymptotically stable.

Stability is not always guaranteed by an admissible control. To enforce closed-loop stability, a positive definite function V, called *control-Lyapunov function*, can be artificially assigned, with a constraint on its decay imposed on the control law. Typically, the constraint requires that the control-Lyapunov function should have an exponential decaying rate of $\sigma > 0$:

$$\frac{dV(x)}{dt} = (\nabla V(x))^\top f(x, u) \leq -\sigma V(x). \tag{9.5}$$

In this way, for any control policy $u = u(x)$, as long as the operational constraints (9.2) and the Lyapunov constraint (9.5) remain feasible for all positive times starting from the initial point, closed-loop stability of the system can be guaranteed with V acting as a Lyapunov function monotonically decreasing with time. Such a method of determining the control policy is called *Lyapunov-based control*.

9.3 Optimal Design of Control-Lyapunov Functions

9.3.1 Stochastic programming framework

Now we consider the optimal design problem of control-Lyapunov functions. The control-Lyapunov function itself does not explicitly have a cost; its associated cost is based on the corresponding Lyapunov-based control. We assume that a domain of interest D in the state space and an exponential decay rate $\sigma > 0$ are given, so that the control cost can be specified by the Lyapunov-based control that attracts any state $x \in D$ towards the origin with the control-Lyapunov function exponentially decaying at a rate of σ.

Let U the denote the control decisions (in a general sense, to be made specific later) to be made when the system states are at point x, and let $\ell(x, U)$ be a function standing for the cost of control decisions U. Due to the existence of the Lyapunov constraint (9.5) and the operational constraints (9.2), the control decisions U will be subject to corresponding constraints, denoted by $\Psi(U, V, x, \sigma) \leq 0$ and $\Pi(U, x) \leq 0$, respectively. For a given state x, we first define a pointwise cost of control.

Definition 9.1. *The pointwise Lyapunov-based control cost $c_0(V, x; \sigma)$ at point x is the minimum achievable cost $\ell(x, U)$ over the control decisions U:*

$$c_0(V, x; \sigma) = \min_U \ \ell(x, U) \quad \text{s.t. } \Psi(U, V, x, \sigma) \leq 0, \ \Pi(U, x) \leq 0. \tag{9.6}$$

Let $K_D(V)$ be a domain containing D and possibly dependent on the control-Lyapunov function V. By averaging the pointwise cost over $K_D(V)$, the Lyapunov-based control cost on the domain D is obtained, which we define as the cost associated with V.

Definition 9.2. *The cost associated with the control-Lyapunov function V is the integral average*

$$c(V; D, \sigma) = \frac{1}{\Omega(K_D(V))} \int_{K_D(V)} c_0(V, x; \sigma) dx, \tag{9.7}$$

where $\Omega(K_D(V))$ is the volume of $K_D(V)$:

$$\Omega(K_D(V)) = \int_{K_D(V)} 1 dx. \tag{9.8}$$

The optimal design of control-Lyapunov function is thus formally stated as the following problem.

Problem 9.0. *Determine the optimal control-Lyapunov function V, given a required decay rate σ and domain D where the initial state is located, with the minimized associated cost $c(V; D, \sigma)$ specified by Definitions 9.1 and 9.2:*

$$\min_V \ c(V; D, \sigma). \tag{9.9}$$

Practically, the control-Lyapunov function needs to be parameterized and the optimization performed within the corresponding parametric forms. Denote by P the corresponding parameters, $\mathcal{P}$ the domain of P, and V_P the corresponding control-Lyapunov function, e.g., an ellipsoidal parameterization is used for quadratic functions:

$V_P(x) = x^\top Px$, $P \in \mathcal{P} = \mathbb{S}_{++}$ (the set of positive definite matrices). Thus the optimal design problem is also written as

$$\min_{P\in\mathcal{P}} \; c(V_P; D, \sigma). \tag{9.10}$$

If x is seen as a random variable uniformly distributed over $K_D(V)$, denoted as $x \sim K_D(V)$, the integral average can be expressed as an expectation:

$$c(V; D, \sigma) = \mathbb{E}_{x\sim K_D(V)} c_0(V, x; \sigma). \tag{9.11}$$

Therefore, the optimal design of control-Lyapunov function is essentially a two-stage stochastic programming problem:

$$\begin{aligned} \min_V \;\; & c(V; D, \sigma) = \mathbb{E}_{x\sim K_D(V)} c_0(V, x; \sigma) \\ & c_0(V, x; \sigma) = \min_U \;\; \ell(x, U) \quad \text{s.t. } \Psi(U, V, x, \sigma) \le 0, \;\; \Pi(U, x) \le 0. \end{aligned} \tag{9.12}$$

Here the control-Lyapunov function V (or its parameterization P) is the first-stage decision, which is made independently of any specific value of the state x. x is viewed as the uncertainty. If $K_D(V)$ is dependent on V, the random distribution of the uncertainty is affected by the first-stage decision, and x is said to be an endogenous uncertainty; otherwise x is called an exogenous uncertainty [130]. The control decision U is the second-stage decision, which is determined after the random variable x is observed (the uncertainty is realized) by minimizing the control cost $\ell(x, U)$.

The specific formulation of the optimal design problem (9.12) depends on the control decision U to be made at x, the function forms of ℓ, Ψ, Π, and the domain $K_D(V)$ on which the pointwise cost is averaged. In the following subsections we consider two different ways of specializing these objects, which result in two different stochastic programming formulations for (9.12).

Remark 9.2. *The essence of stochastic programming is to optimally shape the distribution of the objective function under uncertainty with respect to a risk measure [373]. Besides expectation, there are other indices that account for variance or quantile of the distribution and are suitable as the risk measure, e.g., the conditional value-at-risk (CVaR) [319]. As an alternative, robust optimization [34], which seeks to minimize the maximum cost under uncertainty, leads to a tri-level programming formulation for two-stage problems and requires more restrictive algorithms [479].*

9.3.2 Optimal design of control-Lyapunov function with respect to the control cost over infinite horizon

In the first case, the control decision U made at state x is a predictive one, i.e., the input trajectory for an infinite horizon $\hat{u}(\cdot) : [0, \infty) \to \mathbb{R}^m$. The cost of this control trajectory is defined as

$$\ell(x, \hat{u}(\cdot)) = \int_0^\infty l(\hat{x}(t), \hat{u}(t))dt, \tag{9.13}$$

where $l(\hat{x}, \hat{u})$ is a given function, and $\hat{x}(\cdot)$ is the predicted state trajectory satisfying

$$\dot{\hat{x}}(t) = f(\hat{x}(t), \hat{u}(t)), \ \ t \geq 0, \ \ \hat{x}(0) = x. \tag{9.14}$$

The constraints on the infinite-horizon control decision (Ψ and Π) are the Lyapunov constraint and operational constraints imposed on the entire prediction horizon:

$$\begin{aligned} \nabla V(\hat{x}(t))^\top f(\hat{x}(t), \hat{u}(t)) \leq -\sigma V(\hat{x}(t)), \ t \geq 0, \\ \pi(\hat{x}(t), \hat{u}(t)) \leq 0, \ t \geq 0. \end{aligned} \tag{9.15}$$

Since the cost of control decisions ℓ is evaluated based on the predicted trajectory until convergence to the origin, we only need to average the pointwise cost over the domain of starting points D, i.e., $K_D(V) = D$. The distribution of uncertainty x is independent of the first-stage decision V and thus exogenous. Hence we have the following statement for the optimal design of control-Lyapunov functions with respect to the infinite-horizon control cost.

Problem 9.1. *Determine the optimal control-Lyapunov function $V(x)$, given a decay rate σ and domain D, by solving the following two-stage stochastic program under exogenous uncertainty:*

$$\begin{aligned} \min_V \ \ c(V; D, \sigma) &= \mathbb{E}_{x\sim D} c(V, x; \sigma) \\ c(V, x; \sigma) &= \min_{\hat{u}(\cdot)} \ \int_0^\infty l(\hat{x}(t), \hat{u}(t))dt \quad \text{s.t. } (9.14), (9.15). \end{aligned} \tag{9.16}$$

To make the Problem 9.1 computationally tractable, the integral over the infinite horizon is first approximated by an integral over a finite horizon with a terminal constraint that guarantees convergence, i.e.,

$$\begin{aligned} c_0(V, x; \sigma) = \min_{\hat{u}(\cdot)} \; & \int_0^T l(\hat{x}(t), \hat{u}(t)) dt \\ \text{s.t. } & \dot{\hat{x}}(t) = f(\hat{x}(t), \hat{u}(t)) \\ & \nabla V(\hat{x}(t))^\top f(\hat{x}(t), \hat{u}(t)) \leq -\sigma V(\hat{x}(t)) \\ & \pi(\hat{x}(t), \hat{u}(t)) \leq 0, \;\; t \in [0, T] \\ & \hat{x}(0) = x, \;\; \hat{x}(T) = 0. \end{aligned} \tag{9.17}$$

The continuous-time formulation can be discretized, e.g., using a Gauss-Legendre collocation [124]. Specifically, the collocation points $0 = t_0 < t_1 < \cdots < t_N < t_{N+1} = T$ are generated by the roots $-1 \leq \xi_1 < \cdots < \xi_N \leq 1$ of the N-th order Legendre polynomials $P_N(t) = \frac{1}{2^N N!} \frac{d^N}{dt^N} (t^2 - 1)^N$ through $t_k = (\xi_k + 1)T/2$. The states and inputs at the collocation points are used to interpolate $\hat{x}(t)$ and $\hat{u}(t)$, respectively (the summations need not involve $k = N + 1$ since $x_{N+1} = 0$ and hence $u_{N+1} = 0$):

$$\hat{x}(t) \approx \sum_{k=0}^{N} x_k Q_k(t), \;\; \hat{u}(t) \approx \sum_{k=0}^{N} u_k Q_k(t) \tag{9.18}$$

where $Q_k(t)$ is the $(N + 2)$-th order Lagrangian polynomial

$$Q_k(t) = \prod_{0 \leq j \leq N+1, j \neq k} \frac{t - t_j}{t_k - t_j},$$

satisfying $Q_k(t_j) = 1$ if $k = j$ and 0 otherwise, and hence $x_k = \hat{x}(t_k)$, $u_k = \hat{u}(t_k)$. Any integral over $[0, T]$ is approximated by an average with quadrature weights $\omega_k = T/(1 - \xi_k^2)[dP_N(\xi_k)/dt]^2$, $k = 1, \ldots, N$, and hence

$$\begin{aligned} & \int_0^T l(\hat{x}, \hat{u}) dt \approx \sum_{k=1}^{N} \omega_k l(x_k, u_k), \\ & \hat{x}(T) = \hat{x}(0) + \int_0^T f(\hat{x}, \hat{u}) dt \approx x + \sum_{k=1}^{N} \omega_k f(x_k, u_k). \end{aligned} \tag{9.19}$$

Thus we convert (9.16) into the following form with finite and discretized second-stage decision variables:

$$\begin{aligned}
\min_{V} \ \ c(V; D, \sigma) =& \mathbb{E}_{x\sim D} c_0(V, x; \sigma) \\
c_0(V, x; \sigma) = \min_{u_0,\dots,u_N} \ \ & \sum_{k=1}^{N} \omega_k l(x_k, u_k) \\
\text{s.t.} \ \ & \sum_{j=0}^{N} x_j \dot{Q}_j(t_k) = f(x_k, u_k) \\
& \nabla V(x_k)^\top f(x_k, u_k) \leq -\sigma V(x_k) \\
& \pi(x_k, u_k) \leq 0 \ \ (k = 0, \dots, N) \\
& x_0 = x, \ \ x + \sum_{k=1}^{N} \omega_k f(x_k, u_k) = 0.
\end{aligned} \tag{9.20}$$

9.3.3 Optimal design of control-Lyapunov function with respect to the instantaneous control cost

In the second case, the control decision involves only the here-and-now control input, i.e., $U = u$, and the control cost $\ell(x, u)$ depends only on the instantaneous state and input. Hence the control policy is determined in a non-predictive fashion. Nevertheless, under the Lyapunov-based control, the control-Lyapunov function V becomes a Lyapunov function monotonically decreasing with time, which guarantees that the trajectories starting from any point in D will stay inside the smallest sublevel set of V that contains D, namely

$$K_D(V) = \{x | V(x) \leq \max_{x'\in D} V(x')\}. \tag{9.21}$$

In other words, $K_D(V)$ is the smallest positive invariant set for any control policy resulting from the control-Lyapunov function V that one can ensure *a priori*. We therefore define the cost associated with the control-Lyapunov function $c(V; D, \sigma)$ as the pointwise control cost $c_0(V, x; \sigma)$ averaged within such a Lyapunov sublevel set $K_D(V)$, in the sense that an optimal control-Lyapunov function leads to a lower control cost on average when the control signal is decided based only on instantaneous information. The optimal design of control-Lyapunov function with respect to the instantaneous control cost is stated as the following problem.

Problem 9.2. *Determine the optimal control-Lyapunov function V, given decay rate σ and domain D, by solving the following two-stage stochastic program under endogenous* *uncertainty:*

$$\begin{aligned}\min_V \ & c(V; D, \sigma) = \mathbb{E}_{x\sim K_D(V)} c_0(V, x; \sigma) \\ & c_0(V, x; \sigma) = \min_u \ \ell(x, u) \\ & \text{s.t. } \nabla V(x)^\top f(x, u) \leq -\sigma V(x), \ \ \pi(x, u) \leq 0. \end{aligned} \tag{9.22}$$

where $K_D(V)$ is the smallest Lyapunov sublevel set containing D as defined in (9.21)*.*

Problems 9.1 and 9.2 differ in the following two aspects. First, the uncertainty is exogenous in Problem 9.1 with a fixed distribution of x on D, while it is endogenous in Problem 9.2, as the range of uncertainty $K_D(V)$ varies with the first-stage decision V. Second, the second-stage decision in Problem 9.1 has a dynamic optimization formulation, which is a nonlinear program with a larger size and hence computationally more difficult than an instantaneous decision in Problem 9.2. In the following section, we propose two different algorithms for their solution, respectively.

9.4 Solution of the Stochastic Programs

9.4.1 Offline optimization of Problem 9.1

Consider Problem 9.1. Since the integral average over D is difficult to evaluate, a sample average is typically used to replace the expectation in stochastic programming [373]. Specifically, we first generate Monte Carlo samples $x^{(1)}, x^{(2)}, \dots, x^{(S)}$ from a uniform distribution in D, and approximate the expectation with the sample average

$$c(V; D, \sigma) \approx \frac{1}{S}\sum_{s=1}^{S} c_0(V, x^{(s)}; \sigma), \tag{9.23}$$

Thus we have transformed Problem 9.1 from a stochastic program to a deterministic one. This method is said to be offline since the samples are fixed once chosen, in contrast to the online optimization in the next subsection, where new samples are generated throughout the iterations. For each sample $x^{(s)}$, $s = 1, \dots, S$, we adopt a finite time approximation in the form of (9.16), and the collocation method which approximates the continuous-time formulation in the form of (9.22).

Problem 9.3. *Given Monte Carlo samples $x^{(1)}, \dots, x^{(S)} \sim D$ and the decay rate σ, find*

P with minimized $\sum_{s=1}^{S} c_0(V_P, x^{(s)}; \sigma)$ by solving the nonlinear programming problem

$$\begin{aligned}
\min_{P\in\mathcal{P}} \ \sum_{s=1}^{S}\sum_{k=1}^{N}\omega_k l(x_k^{(s)}, u_k^{(s)}) \quad \text{s.t.} \ & \sum_{j=0}^{N} x_j^{(s)}\dot{Q}_j(t_k) = f(x_k^{(s)}, u_k^{(s)}) \\
& \nabla V_P(x_k^{(s)})^\top f(x_k^{(s)}, u_k^{(s)}) \leq -\sigma V_P(x_k^{(s)}) \\
& \pi(x_k^{(s)}, u_k^{(s)}) \leq 0 \ \ (k=0,\dots,N, \ \ s=1,\dots,S), \\
& x_0^{(s)} = x^{(s)}, \ \ x_0^{(s)} + \sum_{k=1}^{N}\omega_k f(x_k^{(s)}, u_k^{(s)}) = 0 \ \ (s=1,\dots,S)
\end{aligned} \tag{9.24}$$

where the optimization variables include control-Lyapunov function parameters (first-stage decisions) $P \in \mathcal{P}$, control signals under all scenarios (second-stage decisions) $u_k^{(s)}$ and predicted states associated with second-stage decisions $x_k^{(s)}$.

The formulation of (9.24) is a large-scale nonlinear programming problem. We note that the bordered block diagonal structure between the variables and constraints, resulting from the dependence of all scenarios on the same control-Lyapunov function:

$$\begin{array}{c|c|ccccc}
c & P & u^{(1)}, x^{(1)} & u^{(2)}, x^{(2)} & \cdots & u^{(S)}, x^{(S)} \\
 & * & * & & & \\
 & * & & * & & \\
 & \vdots & & & \ddots & \\
 & * & & & & * \\
\hline
* & & * & * & \cdots & *
\end{array}$$

can be exploited to tackle the problem either by decomposing the linear algebraic operations involved in the optimization [187] or by decomposing the monolithic formulation into subproblems corresponding to individual samples [356]. Here we adopt the latter approach for its relative ease of implementation. We use the *alternating direction of multipliers* (ADMM), also known as progressive hedging for two-stage stochastic programs [350], which is an augmented Lagrangian-based algorithm [44].

Specifically, the decision variables P are duplicated for S times into $P^{(s)}$ with constraints on their consensus, namely the non-anticipativity constraints $P = P^{(s)}$, $s = 1, \dots, S$, and the constraints involving V_P and $u_k^{(s)}, x_k^{(s)}$ become

$$\nabla V_{P^{(s)}}(x_k^{(s)})^\top f(x_k^{(s)}, u_k^{(s)}) \leq -\sigma V_{P^{(s)}}(x_k^{(s)}). \tag{9.25}$$

The problem (9.24) is composed of S subproblems:

$$\begin{aligned}\min\ \sum_{k=1}^{N}\omega_k l(x_k^{(s)}, u_k^{(s)}) \quad \text{s.t.}\ & \sum_{j=0}^{N} x_j^{(s)}\dot{Q}_j(t_k) = f(x_k^{(s)}, u_k^{(s)}) \\ & \nabla V_{P^{(s)}}(x_k^{(s)})^\top f(x_k^{(s)}, u_k^{(s)}) \le -\sigma V_{P^{(s)}}(x_k^{(s)}) \\ & \pi(x_k^{(s)}, u_k^{(s)}) \le 0 \ \ (k = 0, \dots, N) \\ & x_0^{(s)} = x^{(s)},\ \ x_0^{(s)} + \sum_{k=1}^{N}\omega_k f(x_k^{(s)}, u_k^{(s)}) = 0\end{aligned} \tag{9.26}$$

each corresponding to a sample s, connected by the non-anticipativity constraints. The augmented Lagrangian is then constructed by introducing dual variables $\Lambda^{(s)}$ and a penalty parameter $\tau > 0$:

$$\mathcal{L} = \sum_{s=1}^{S}\mathcal{L}_s = \sum_{s=1}^{S}\left[\sum_{k=1}^{N}\omega_k l(x_k^{(s)}, u_k^{(s)}) + \langle \Lambda^{(s)}, P - P^{(s)}\rangle + \frac{\tau}{2}\|P - P^{(s)}\|^2\right] \tag{9.27}$$

where $\langle\cdot,\cdot\rangle$ is an inner product and $\|\cdot\|$ is the correspondingly specified norm (e.g., in the case that P and Λ are matrices, $\langle \Lambda, P\rangle = \mathrm{tr}(\Lambda^\top P)$ is the trace of their matrix product with Λ transposed and $\|P\| = \sqrt{\mathrm{tr}(P^\top P)}$ is the Frobenius norm). The optimization of (9.24) is thus equivalent to

$$\max_{\Lambda^{(s)}}\ \min_{P, P^{(s)}, u^{(s)}, x^{(s)}}\ \mathcal{L}\left(P, \{P^{(s)}, u^{(s)}, x^{(s)}\}_{s=1}^{S}, \{\Lambda^{(s)}\}_{s=1}^{S}\right) \tag{9.28}$$

subject to the constraints involving $P^{(s)}$ and $u^{(s)}, x^{(s)}$.

In the ADMM algorithm, each iteration involves the following 3 update steps (in which the first and the third step can be parallelized with respect to s), namely distributed primal update, consensus update and dual update:

$$\begin{aligned}(P^{(s)}, u^{(s)}, x^{(s)}) :=& \arg\min \mathcal{L}_s\left(P, P^{(s)}, u^{(s)}, x^{(s)}, \Lambda^{(s)}\right) \\ P :=& \arg\min \mathcal{L}\left(P, \{P^{(s)}, u^{(s)}, x^{(s)}\}_{s=1}^{S}, \{\Lambda^{(s)}\}_{s=1}^{S}\right) \\ \Lambda^{(s)} :=& \Lambda^{(s)} + \tau(P - P^{(s)}).\end{aligned} \tag{9.29}$$

With the definition of augmented Lagrangian (9.27), the ADMM algorithm (9.29) can be formulated as follows.

Algorithm 9.1 (ADMM)**.** *Given the Monte Carlo samples $x^{(1)}, \dots, x^{(S)} \sim D$, initialize P and $\Lambda^{(1)}, \dots, \Lambda^{(s)}$. Set the penalty constant $\mu > 0$. Iterate the following three steps in sequence until convergence:*

- *Distributed primal update, i.e., obtain $P^{(s)}$ by solving the following problem, for $s = 1, \dots, S$ in parallel:*

$$\begin{aligned} \min \quad & \sum_{k=1}^{N} \omega_k l(x_k^{(s)}, u_k^{(s)}) + \frac{\tau}{2} \left\| P - P^{(s)} + \Lambda^{(s)}/\tau \right\|^2 \\ \text{s.t.} \quad & \sum_{j=0}^{N} x_j^{(s)} \dot{Q}_j(t_k) = f(x_k^{(s)}, u_k^{(s)}) \\ & \nabla V_{P^{(s)}}(x_k^{(s)})^\top f(x_k^{(s)}, u_k^{(s)}) \le -\sigma V_{P^{(s)}}(x_k^{(s)}) \\ & \pi(x_k^{(s)}, u_k^{(s)}) \le 0 \ \ (k = 0, \dots, N) \\ & x_0^{(s)} = x^{(s)}, \ \ x_0^{(s)} + \sum_{k=1}^{N} \omega_k f(x_k^{(s)}, u_k^{(s)}) = 0 \end{aligned} \tag{9.30}$$

  This can be seen as an optimal control problem involving $P^{(s)}$ as static parameters with an associated quadratic cost, in addition to the predicted states and inputs.

- *Consensus update*

$$P := \frac{1}{S} \sum_{s=1}^{S} \left(P^{(s)} - \frac{1}{\tau} \Lambda^{(s)} \right) \tag{9.31}$$

- *Dual update for $s = 1, \dots, S$ in parallel*

$$\Lambda^{(s)} := \Lambda^{(s)} + \tau (P - P^{(s)}) \tag{9.32}$$

Remark 9.3. *ADMM in its usual form is known to converge linearly for convex problems [148, 161] and some nonconvex problems under specific assumptions [443]. The acceleration of ADMM has been discussed recently [131]. However, the applicability of accelerated ADMM algorithms are so far guaranteed only for convex problems.*

9.4.2 Online optimization of Problem 9.2

For Problem 9.2, due to the endogeneity of the uncertainty x, i.e. the distribution of x on $K_D(V)$, it is impossible to obtain samples before the optimization. Instead, the Lyapunov function needs to be optimized in an online manner, i.e., the first-stage decision variables P are continually optimized by new samples from the updated distribution $x \sim K_D(V_P)$. This type of algorithms are called *stochastic approximation* ones since the objective function and/or its gradient are stochastically approximated by their evaluations at online samples, and are known to require little data storage and converge to the true optimum rather than an approximation [373, Chapter 5].

Specifically, we adopt the idea of stochastic gradient descent (SGD), which successively updates the decision variable P in the reverse direction of $G(V_P; x, \sigma)$ for a random sample $x \in K_D(V_P)$, where $G(V_P; x, \sigma)$ is a stochastic estimate of the gradient direction $\nabla_P c(V_P; D, \sigma)$ based on the information of x, i.e., $\nabla_P c(V_P; D, \sigma) = \mathbb{E}_{x \sim K_D(V_P)}[G(V_P; x, \sigma)]$. The SGD algorithm is written as

$$P^{(s+1)} = P^{(s)} - \eta_s G(V_{P^{(s)}}; x^{(s)}, \sigma), \;\; s = 0, 1, \dots, \tag{9.33}$$

where $P^{(s)}$ is the result after the s-th update, and $\eta_s > 0$ is the step size of the s-th update. The average of all intermediate results weighted by the corresponding step sizes

$$\hat{P}^{(s)} = \frac{\sum_{r=1}^{s} \eta_r P^{(r)}}{\sum_{r=1}^{s} \eta_r} \tag{9.34}$$

is used to approximate the optimal value of P. A step size of the form $\eta_s = \eta_1/\sqrt{s}$ has been suggested for fast convergence [289]. However, the availability of the stochastic gradient $G(V_P; x, \sigma)$ as a prerequisite of implementing (9.33) is usually a "stochastic oracle" and taken as an assumption in the literature [373]. To execute (9.33), we need to overcome the following two issues.

The first issue is the endogeity of x. For stochastic programming problems with exogenous uncertainties, from the definition of $c(V_P; D, \sigma) = \mathbb{E}_{x \sim K_D}[c_0(V_P, x; \sigma)]$ it can be inferred that $\nabla_P c_0(V_P, x; \sigma)$ for any $x \sim K_D$ is an unbiased estimate of the gradient $\nabla_P c(V_P; D, \sigma)$ and can be taken as $G(V_P; x, \sigma)$. In the presence of endogenous uncertainty, additional terms related to the boundary and volume changes in $K_D(V_P)$ should be considered. For a specific type of $K_D(V_P)$, we have the following proposition.

Proposition 9.1. *Under the definition of cost function in* (9.22)*, assume that $K_D(V_P)$ is dependent on P through*

$$K_D(V_P) = \{x | x = T_P y, y \in S\} \tag{9.35}$$

for a fixed set S and a positive definite linear transformation T_P dependent on P. Then *the gradient of the control cost with respect to P can be expressed as:*

$$\begin{aligned}&\nabla_P c(V_P; D, \sigma) = \mathbb{E}_{x\sim K_D(V_P)}[\nabla_P c_0(V_P, x; \sigma)\\&+ \frac{1}{2} x \nabla_x^\top c_0(V_P, x; \sigma)(\nabla_P \ln T_P) + \frac{1}{2}(\nabla_P \ln T_P)\nabla_x c_0(V_P, x; \sigma)x^\top].\end{aligned} \tag{9.36}$$

Proof. Denote $K_P = K_D(V_P)$. We have

$$c(V_P; D, \sigma) = \int_{K_P} \frac{c_0(V_P, x; \sigma)}{\Omega(K_P)} dx. \tag{9.37}$$

When P varies, the differentiation with respect to P gives

$$\begin{aligned}d_P c(V_P; D, \sigma) &= \int_{\partial K_P} \frac{c_0(V_P, x; \sigma)}{\Omega(K_P)}(d_P x \cdot n)dx\\&\quad+ \int_{K_P} \left[\frac{d_P c_0(V_P, x; \sigma)}{\Omega(K_P)} - \frac{c_0(V_P, x; \sigma) d_P \Omega(K_P)}{\Omega^2(K_P)}\right] dx\\&= \int_{\partial K_P} \frac{c_0(V_P, x; \sigma)}{\Omega(K_P)}(d_P x \cdot n)dx\\&\quad+ \mathbb{E}[d_P c_0(V_P, x; \sigma)] - \mathbb{E}[c_0(V_P, x; \sigma)] d_P \ln \Omega(K_P).\end{aligned} \tag{9.38}$$

where ∂K_P is the boundary of K_P and n is the outward unit normal vector on ∂K_P. Using the Gauss-Ostrogradsky divergence theorem,

$$\int_{\partial K_P} \frac{c_0(V_P, x; \sigma)}{\Omega(K_P)}(d_P x \cdot n)dx = \int_{K_P} \left[\frac{c_0(V_P, x; \sigma)\nabla \cdot (d_P x)}{\Omega(K_P)} + \frac{\nabla c_0(V_P, x; \sigma) \cdot d_P x}{\Omega(K_P)}\right] dx \tag{9.39}$$

With $x = T_P y$, $d_P x = d_P T_P y = (d_P T_P)T_P^{-1}x$, hence $\nabla \cdot (d_P x) = \mathrm{tr}((d_P T_P)T_P^{-1})$, and

$$\text{R.H.S. of (9.39)} = \mathbb{E}[c_0(V_P, x; \sigma)]\mathrm{tr}((d_P T_P)T_P^{-1}) + \mathbb{E}[\nabla_x^\top c_0(V_P, x; \sigma)(d_P T_P)T_P^{-1}x]. \tag{9.40}$$

Substituting the above expression into (9.38), we obtain

$$\begin{aligned}d_P c(V_P; D, \sigma) =& \mathbb{E}[d_P c_0(V_P, x; \sigma)] - \mathbb{E}[c_0(V_P, x; \sigma)] d_P \ln \Omega(K_P)\\&+ \mathbb{E}[c_0(V_P, x; \sigma)]\mathrm{tr}((d_P T_P)T_P^{-1}) + \mathbb{E}[\nabla_x^\top c_0(V_P, x; \sigma)(d_P T_P)T_P^{-1}x]\\=& \mathbb{E}[d_P c_0(V_P, x; \sigma)] - \mathbb{E}[c_0(V_P, x; \sigma)] d_P[\ln \Omega(K_P) - \mathrm{tr}(\ln T_P)]\\&+ \mathrm{tr}\left(\mathbb{E}[x \nabla_x^\top c_0(V_P, x; \sigma)](d_P \ln T_P)\right).\end{aligned} \tag{9.41}$$

Since $K_P = \{T_P y | y \in S\}$, $\Omega(K_P) = \Omega(S) \det T_P$, hence

$$\ln \Omega(K_P) - \operatorname{tr}(\ln T_P) = \ln \Omega(S) = \text{const.} \tag{9.42}$$

The second term on the right-hand side of (9.41) is thus 0. This gives the following relation in the form of gradients

$$\begin{aligned}\nabla_P c(V_P; D, \sigma) =& \mathbb{E}[\nabla_P c_0(V_P, x; \sigma)] + \frac{1}{2}\mathbb{E}[x \nabla_x^\top c_0(V_P, x; \sigma)](\nabla_P \ln T_P) \\ &+ \frac{1}{2}(\nabla_P \ln T_P)\mathbb{E}[\nabla_x c_0(V_P, x; \sigma) x^\top].\end{aligned} \tag{9.43}$$

This concludes the proof. $\square$

We show that the ellipsoidal parameterization $V_P(x) = x^\top P x, P \in \mathbb{S}_{++}$ satisfies the assumption of Proposition 9.1. Denote by $M_P = \max_{x \in D} x^\top P x$. Then we have $K_P = \{x | x^\top P x \le M_P\}$. We have the representation that $x = T_P y = \sqrt{M_P} P^{-1/2} y$ with $y \in S = \{y | \|y\| \le 1\}$. Hence

$$\nabla_P \ln T_P = \nabla_P \ln(\sqrt{M_P} P^{-1/2}) = \frac{1}{2}(\nabla_P \ln M_P - P^{-1}). \tag{9.44}$$

Specifically, if D is a unit ball, $M_P = \mu_{\max}(P)$ is the largest eigenvalue of P, then $\nabla_P \mu_{\max}(P) = e_1(P) e_1(P)^\top$, where $e_1(P)$ is the unit eigenvector associated with the largest eigenvalue $\mu_{\max}(P)$. Therefore

$$\nabla_P \ln T_P = \frac{1}{2}\left(\frac{e_1(P) e_1(P)^\top}{\mu_{\max}(P)} - P^{-1}\right). \tag{9.45}$$

It follows that a stochastic gradient is

$$\begin{aligned}G(V_P; x, \sigma) =& \nabla_P c_0(V_P, x; \sigma) + \frac{1}{2} x \nabla_x^\top c_0(V_P, x; \sigma)(\nabla_P \ln T_P) \\ &+ \frac{1}{2}(\nabla_P \ln T_P) \nabla_x c_0(V_P, x; \sigma) x^\top,\end{aligned} \tag{9.46}$$

and hence the SGD is

$$\begin{aligned}P^{(s+1)} =& P^{(s)} - \eta_s \nabla_P c_0(V_{P^{(s)}}, x^{(s)}; \sigma) - \frac{\eta_s}{2}(x^{(s)}) \nabla_x^\top c_0(V_{P^{(s)}}, x^{(s)}; \sigma)(\nabla_P \ln T_{P^{(s)}}) \\ &- \frac{\eta_s}{2}(\nabla_P \ln T_{P^{(s)}}) \nabla_x c_0(V_{P^{(s)}}, x^{(s)}; \sigma)(x^{(s)})^\top.\end{aligned} \tag{9.47}$$

The second issue in the implementation of the SGD algorithm is that, due to the two-stage nature of the problem, the dependence of $c_0(V_P, x^{(s)}; \sigma)$ on P and $x^{(s)}$ is implicit through the optimization problem (9.22) involving u, and that the gradients $\nabla_P c_0(V_P, x; \sigma)$ and $\nabla_x c_0(V_P, x; \sigma)$ in (9.47) can not be analytically obtained. Therefore, we approximate the gradients by solving the so-called proximal point of $P^{(s)}$ with respect to function $\eta_s c_0$ [313, Chapter 4]:

$$\bar{P}^{(s)} = \arg\min_P \left[c_0(V_P, x^{(s)}; \sigma) + \frac{1}{2\eta_s} \|P - P^{(s)}\|^2 \right]. \tag{9.48}$$

With a small choice of η_s, by linearizing the first term around P_0, the minimizer $\bar{P}^{(s)}$ approximates $P^{(s)} - \eta_s \nabla_P c_0(V_{P^{(s)}}, x^{(s)}; \sigma)$. An additional trust region constraint is imposed on $P^{(s)}$ to prevent large deviations: $\|\bar{P}^{(s)} - P^{(s)}\|_\infty \leq \eta_s^{1/2}$. Hence $P^{(s)} - \bar{P}^{(s)}$ is used to replace the $\eta_s \nabla_P c_0$ term in (9.47). In a similar manner, we solve the proximal point of $x^{(s)}$ with respect to $\eta_s c_0$:

$$\bar{x}^{(s)} := \arg\min_x \left[c_0(V_{P^{(s)}}, x^{(s)}; \sigma) + \frac{1}{2\eta_s} \|x - x^{(s)}\|^2 \right], \tag{9.49}$$

and approximate the $\eta_s \nabla_x c$ term with $x^{(s)} - \bar{x}^{(s)}$. The SGD (9.47) is thus approximated with

$$P^{(s+1)} = \bar{P}^{(s)} - \frac{1}{2} x^{(s)} (x^{(s)} - \bar{x}^{(s)})^\top (\nabla_P \ln T_{P^{(s)}}) - \frac{1}{2} (\nabla_P \ln T_{P^{(s)}}) (x^{(s)} - \bar{x}^{(s)}) x^{(s)\top}. \tag{9.50}$$

Finally, we note that the updated $P^{(s+1)}$ may escape from its domain $\mathcal{P}$. Hence we follow with a projection:

$$P^{(s+1)} := \mathrm{proj}_{\mathcal{P}} P^{(s+1)} = \arg\min_{P \in \mathcal{P}} \|P - P^{(s+1)}\|^2. \tag{9.51}$$

For example, under $V(x) = x^\top P x$ where P must be positive semidefinite, the projection is performed by a spectral decomposition $P^{(s+1)} = \sum_i \mu_i v_i v_i^\top$ followed by the removal of negative eigenmodes $P^{(s+1)} := \sum_i \max(\mu_i, 0) v_i v_i^\top$ [45, Page 399]. In summary, the online optimization of (9.22) is solved through the following algorithm.

Algorithm 9.2 (Stochastic Proximal Algorithm)**.** *Initialize $P^{(0)}$. For $s = 1, 2, \ldots$, iterate the following steps until convergence.*

- *Determine $K_D(V_{P^{(s)}})$ and sample $x^{(s)} \sim K_D(V_{P^{(s)}})$.*

- *Solve the proximal points*

$$\begin{aligned}\bar{P}^{(s)} :=& \arg\min_{P,u} \left[\ell(x^{(s)}, u) + \frac{1}{2\eta_s}\|P - P^{(s)}\|^2\right] \\ \text{s.t. } & \|P - P^{(s)}\|_\infty \leq \eta_s^{1/2} \\ & \nabla V_P(x^{(s)})^\top f(x^{(s)}, u) \leq -\sigma V_P(x^{(s)}) \\ & \pi(x^{(s)}, u) \leq 0\end{aligned} \tag{9.52}$$

  and

$$\begin{aligned}\bar{x}^{(s)} :=& \arg\min_{x,u} \left[\ell(x, u) + \frac{1}{2\eta_s}\|x - x^{(s)}\|^2\right] \\ \text{s.t. } & \|x - x^{(s)}\|_\infty \leq \eta_s^{1/2} \\ & \nabla V_{P^{(s)}}(x)^\top f(x, u) \leq -\sigma V_{P^{(s)}}(x) \\ & \pi(x, u) \leq 0\end{aligned} \tag{9.53}$$

- *Compute* $\nabla_P \ln T_{P^{(s)}}$*.*
- *Perform the SGD update according to* (9.50)*.*
- *Project the updated* P *according to* (9.51)*.*

Record the total iteration S*. Then obtain the step size-weighted average* $\hat{P}^{(S)}$ *according to* (9.34)*.*

Remark 9.4. *The convergence rate of SGD towards the optimum is* $O(1/k)$ *and* $O(1/\sqrt{k})$ *for smooth and general convex problems, respectively, where* k *is the iteration number [41]. Adaptive variants of SGD that improve its computational efficiency, e.g., the Adam algorithm [195], have been widely applied in the practice of machine learning. SGD and its accelerated algorithms for non-convex optimization have also been studied [125]. In this work, we use the SGD algorithm in its basic form* (9.33) *for simplicity.*

Remark 9.5. *Both Problems 9.1 and 9.2 in the present chapter fall into the category of "simulation optimization" [132], where the objective functions and the constraints are evaluated through stochastic simulation (of Lyapunov-based control under random initial states). There exist several categories of algorithms other than gradient-based methods, e.g., response surface, pattern search, and metaheuristics [5]. However, it appears that these methods either behave well only on very small-scale problems or are computationally prohibitive.*

9.5 Case Study on a Chemical Reactor

With the algorithms for Problems 9.1 and 9.2 introduced in the previous section, we now apply the proposed methods on the optimal Lyapunov function design of a continuously stirred tank reactor system (CSTR). In the reactor, an exothermic first order reaction $\mathrm{A} \to \mathrm{B}$ takes place. The dynamic model is

$$\dot{x}_1 = 1 - x_1 - \alpha x_1 \mathrm{e}^{\gamma(1-1/x_2)}, \quad \dot{x}_2 = 1 - x_2 + \alpha\beta x_1 \mathrm{e}^{\gamma(1-1/x_2)} + \omega(u - x_2) \tag{9.54}$$

where x_1, x_2, u are the dimensionless reactant concentration, reactor temperature, and coolant temperature, respectively. The parameter values are $\alpha = 0.0637$, $\beta = 0.167$ ($\beta > 0$ corresponds to an exothermal reaction), $\gamma = 37.7$, $\omega = 0.1356$. For the system (9.54) under constant control input $u^{\mathrm{ss}} = 1.048$, three steady states exist – $x = [0.8498 \;\; 1.0278]^\top$ (low conversion, low temperature), $x = [0.1996 \;\; 1.1234]^\top$ (high conversion, high temperature), and $x = [0.5196 \;\; 1.0764]^\top$(medium conversion and temperature), among which the first two are open-loop stable and the third one is open-loop unstable.

We consider the state regulation problem that aims to steer the reactor towards the open-loop unstable steady state $x_{\mathrm{ss}} = [0.5196 \;\; 1.0764]^\top$ under $u_{\mathrm{ss}} = 1.048$. The system equations (9.54) are translated and scaled by $x_1 := (x_1 - x_1^{\mathrm{ss}})/0.20$, $x_2 := (x_2 - x_2^{\mathrm{ss}})/0.03$ and $u := (u - u_{\mathrm{ss}})/0.20$ to obtain a form of $\dot{x} = f_0(x) + g(x)u$ for which $x = 0, u = 0$ represents the steady state to be regulated at. We assume the absence of any operational constraints π for simplicity. The optimization of the Lyapunov function is performed under a parameterization of $V_P(x) = x^\top P x$, $P \in \mathcal{P} = \mathbb{S}_{++}$, under which Proposition 9.1 holds. We seek to stabilize the reactor from any initial condition in $D = \{x | x^\top x \leq 1\}$ with an exponential convergence rate faster than $\sigma = 1$. This has specified the optimal control-Lyapunov function design problem (9.9).

We first consider Problem 9.1 with a stage cost of $l(\hat{x}, \hat{u}) = \hat{x}_1^2 + \hat{x}_2^2 + 2.5\hat{u}^2$. The problem is solved by sample average approximation and ADMM (Algorithm 9.1), for which $S = 100$ random samples are generated on D (see the first subplot of Fig. 9.1). The finite horizon is set as $T = 5$ (since $\mathrm{e}^{-5} < 0.01$), and the number of collocation points inside $[0, T]$ is $N = 9$. For the ADMM algorithm, the penalty parameter is set as $\tau = 10$ for the first 50 iterations, multiplied by $\sqrt{10}$ at the 51st iteration and for every 5 subsequent iterations in order to enforce the consensus constraints [300]. The second subplot of Fig. 9.1 shows the convergence of the components of P, i.e., p_{11} and

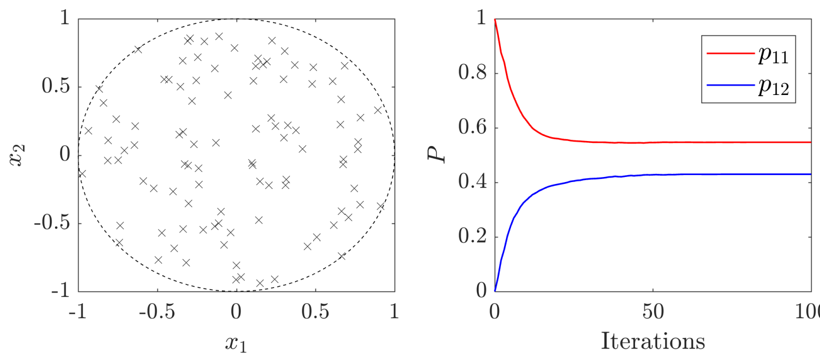


Figure 9.1: The random samples $x^{(1)}, \dots, x^{(S)} \sim D$ (left) and the evolution of the two parameters of the Lyapunov function throughout 100 ADMM iterations (right).

$p_{12} = p_{21}$, within 100 iterations (p_{22} is fixed at 1 without loss of generality). After 86 iterations, the consensus constraints are satisfied below a tolerance of 10^{-4}. Hence we have approximately obtained the optimal control-Lyapunov function with respect to the infinite horizon control cost: $V(x) = x^\top P_* x$ where $P_* = [0.5471, 0.4299; 0.4299, 1]$.

To illustrate the optimality of P_*, we compare the costs associated with the control-Lyapunov functions of different P values throughout the iterations. For this purpose, we generate another batch of 100 random samples on D, and evaluate the corresponding values of $c_0(V_P, x; \sigma)$ for all the samples under the P values at iterations 0, 5, 10 and 86, where iteration 0 corresponds to the initial guess $P_0 = I_2$ (the unit 2×2 matrix), and iteration 86 corresponds to the optimum. The result is shown by the stem plot in Fig. 9.2. It can be clearly observed that for the samples, the control cost gradually decreases as the control-Lyapunov function proceeds from its intermediate solutions at iterations 0, 5, 10 to the optimum.

Now we consider Problem 9.2 with an instantaneous control cost of $\ell(x, u) = x_1^2 + x_2^2 + 2.5u^2$, which is solved through the stochastic proximal algorithm (Algorithm 9.2). The step size is of the form $\eta_s = \eta_1/\sqrt{s}$ where $\eta_1 = 10^{-2}$ is empirically determined. For faster convergence, the weighted averages (9.34) are evaluated only after 10^3 iterations. Fig. 9.3 shows the solution trajectories of $P^{(s)}$ and the weighted average of $P^{(1)}, \dots, P^{(s)}$ given by (9.52) throughout 10^5 iterations. It is observed from Fig. 9.3 that during iterations 10^3–10^5, the oscillation of $P^{(s)}$ has been stabilized in a certain interval, leading to a well-converging sequence of the weighted average $\hat{P}^{(s)}$. At the 15731st iteration, the components of $\hat{P}$ vary less than 10^{-4} for the previous 1000 iterations. We take the $\hat{P}$ at this iteration as the optimal control-Lyapunov function: $P_* \approx \hat{P}^{(15731)} =$

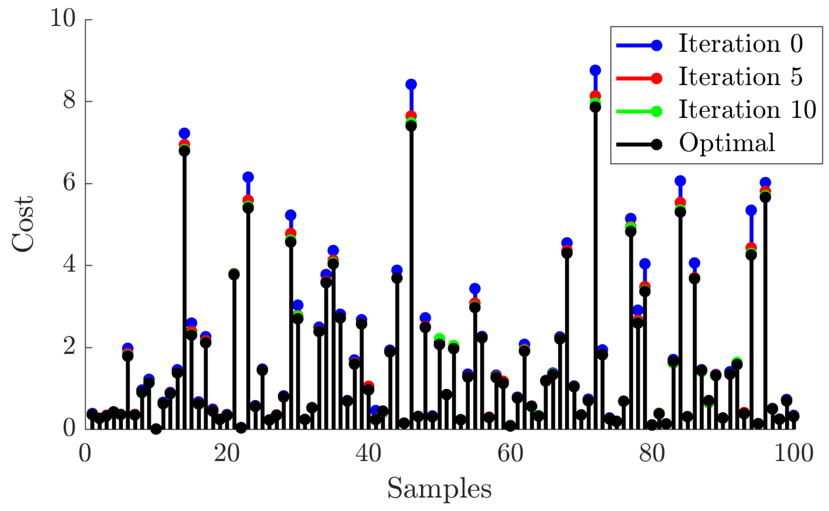


Figure 9.2: Cumulative control cost $c_0(V_P, x; \sigma)$ for 100 samples for different P values throughout the ADMM iterations.

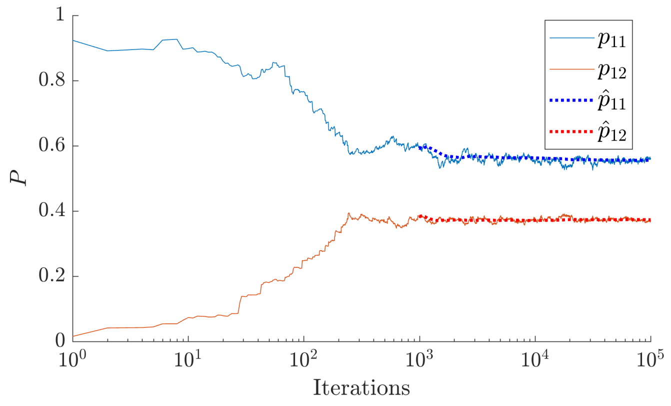


Figure 9.3: Evolution of $P^{(s)}$ (upper) and weighted average $\hat{P}^{(s)}$ (lower) by stochastic proximal algorithm.

$[0.5608, 0.3721; 0.3721, 1]$.

The optimum value of P obtained here is compared with the intermediate results $P^{(s)}$ for $s = 1, 10, 10^2, 10^3$ in terms of the resulting instantaneous control cost using 100 random samples for each. Fig. 9.4 shows the empirical cumulative density function of the control cost $c_0(V_P, x; \sigma)$ under these P values. From the higher and lower parts of the lines, a left shifting and a right shifting can be observed respectively, implying that throughout the iterations, the expected control cost is decreased through diminishing

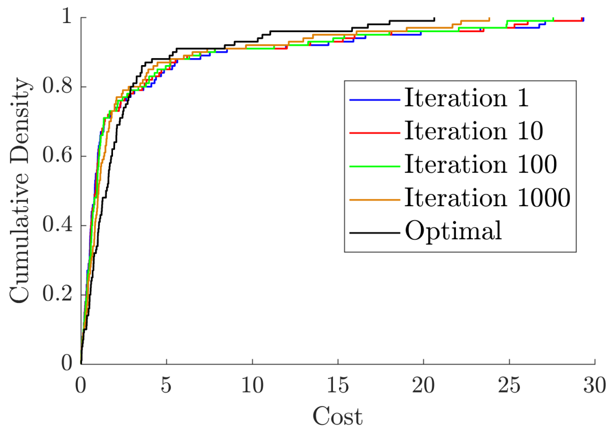


Figure 9.4: Distribution of the instantaneous control cost for different P values throughout the stochastic proximal iterations.

the possibility of containing states that require the higher control costs (higher than 3), at the expense of increasing the cost of the low-cost states that compose 80% of the samples.

9.6 Conclusion

The wide application of Lyapunov-based control requires a method of systematically designing a control-Lyapunov function that is optimal with respect to a cost of the resulting Lyapunov-based control strategy. In this work, we propose a two-stage stochastic programming framework for the optimal design of control-Lyapunov functions, and two different specific formulations depending on whether the control cost is defined over an infinite prediction horizon or only on instantaneous values of the states and inputs. We also propose solution algorithms to these stochastic programs, respectively. Specifically, the first formulation is approximated via sample average approximation and solved with a decomposition-based optimization algorithm – ADMM. For the second formulation where the uncertainty is endogenous in nature, we establish a stochastic proximal algorithm based on stochastic gradient descent, which allows online optimization of the control-Lyapunov function. Through this study, we have provided a generic approach to the design of control-Lyapunov functions, thus facilitating the implementation of Lyapunov-based control strategies.

Chapter 10
Lyapunov Flexibility Analysis for Integrated Design and Control of Nonlinear Processes

10.1 Introduction

The existence of uncertainties in chemical processes has motivated analyses of flexibility, resiliency, operability or controllability, and process design with consideration of these properties [137]. Due to the dependence of process dynamics on the design and possible conflicts between an economic objective and process operability, traditional design procedures where the design and control are separated may result in suboptimal or undesirable solutions; hence, *integrated design and control* approaches have attracted attention [370, 348, 473, 374]. Key to these works is the characterization of achievable attenuation of the uncertainties by manipulating the control inputs, used as the basis of evaluating and optimizing process design. The approaches to integrated design and control can be classified into two major categories [473], namely controllability index-based and mixed-integer dynamic optimization (MIDO)-based ones. In the first type of approaches, given the process model under any specific design, an index is defined to capture the operability of the dynamic system and used as a part of the design objective or constraint [217, 242]. However, such indices are typically extracted from a linearized model and do not properly and sufficiently capture the dynamic features of nonlinear processes, while those defined based on nonlinear analysis [224] are used mainly for analysis and are difficult to be integrated into the optimal design problem.

In the MIDO-based approaches, the dynamic flexibility analysis is directly formulated in the parlance of nonlinear optimal control [88, 112]. Such formulations are rigorous in that they involve detailed descriptions of all the uncertain scenarios, which, however, becomes the very reason for their computational difficulty, despite significant progress in dynamic and stochastic programming. This issue can be partly relieved by embedding simplified forms of control such as LQG [246] or multi-parametric MPC [85].

In this chapter, we propose an efficient framework for dynamic flexibility analysis of nonlinear processes based on the concept of *Lyapunov function* [193], which is a state-dependent function capturing the deviation of the system states from the designed steady state. Specifically,

- Given any Lyapunov function, a hypothetical Lyapunov-based controller can be specified in terms of the descent of the Lyapunov function. Under this control policy, the effect of the uncertainties on the system states can be described by the resulting average level of the Lyapunov function value, which is compared to the boundary of operational feasibility to give a characterization of dynamic flexibility.

- The operating cost under uncertainties is related to the average level of the Lyapunov function value under Lyapunov-based control. By considering the choice of the Lyapunov function as a part of the process design decisions, the optimal process design subject to a dynamic flexibility constraint can be found.

Following these lines, we propose in this work formulations for the flexibility test, flexibility index, and optimal flexible process design problems. The framework is illustrated by a case study on a two-reactor system.

Compared to the MIDO approaches, the Lyapunov perspective gives an alternative to large-scale optimization problems resulting from a multi-scenario description of the dynamic process variables; compared to the controllability index-based approaches, the Lyapunov flexibility analysis proposed here explicitly accounts for nonlinear dynamics and can be easily combined with optimal design. We note that since the Lyapunov function only plays an auxiliary role in deriving the optimization problems, and correction factors based on simulations are introduced to redeem any resulting inexactness, such an approach does not limit itself to a Lyapunov-based controller.

10.2 Lyapunov Function and L_2-Stability

We consider nonlinear dynamic systems of the form:

$$\dot{x} = f(x) + g(x)u + b(x)w \tag{10.1}$$

where x and u are the vectors of states and control inputs respectively. The vector of uncertainties w may include parametric uncertainties and time-varying disturbances. The origin is the operational steady state ($f(0) = 0$). Lyapunov theory is widely used in the stability analysis of nonlinear systems. It refers to the construction of a positive definite function $V(x)$ acting as the distance metric of the states to the desired point.

Suppose that the inputs are constrained by $\|u\|_\infty \leq 1$, and that $V(x)$ is continuously differentiable. Among all the feasible control policies satisfying the input constraints, $u(x) = -\text{sign}(g(x)^\top \nabla V(x))$ gives the fastest descent of $V(x)$:

$$\dot{V} = \nabla V(x)^\top f(x) - \|\nabla V(x)^\top g(x)\|_1 + \nabla V(x)^\top b(x) w. \tag{10.2}$$

Under this best possible policy, we consider any output vector in the form of $y = h(x)$. If there exists a positive number γ and a positive time constant τ, such that for any x in a domain of the state space X, the Hamilton-Jacobi inequality [427]

$$\nabla V(x)^\top f(x) - \|\nabla V(x)^\top g(x)\|_1 + \frac{\tau}{4\gamma^2} \nabla V(x)^\top b(x) b(x)^\top \nabla V(x) + \frac{1}{\tau} h(x)^\top h(x) \leq 0 \tag{10.3}$$

is satisfied, then within this domain X, the L_2-gain from the disturbances to the outputs is finite and upper bounded by γ^2. In other words, for any time $T \geq 0$ and uncertainty signal $w(t)$, any trajectory starting from the origin satisfies

$$\int_0^T \|y(t)\|^2 dt \leq \gamma^2 \int_0^T \|w(t)\|^2 dt. \tag{10.4}$$

For the purpose of Lyapunov flexibility analysis, we consider for simplicity a scalar output $h(x) = V(x)^{1/2}$ to capture the effect of the uncertainties on the Lyapunov function value $V(x)$. Then within the domain on which

$$\nabla V(x)^\top f(x) - \|\nabla V(x)^\top g(x)\|_1 + \frac{\tau}{4\gamma^2} \nabla V(x)^\top b(x) b(x)^\top \nabla V(x) + \frac{1}{\tau} V(x) \leq 0 \tag{10.5}$$

holds, under any uncertainty that has a bounded magnitude Δ in time-averaged squared norm $\frac{1}{T}\int_0^T \|w(t)\|^2 dt \leq \Delta$ (e.g., a sinusoidal or static signal), the Lyapunov function value will be bounded in time average $\frac{1}{T}\int_0^T V(t) dt \leq \gamma^2 \Delta$. Hence, given the uncertainty level $\Delta \geq 0$, the response of the system will result in an estimated upper bound $\gamma^2 \Delta$ of the Lyapunov function value, as long as the Hamilton-Jacobi inequality (10.5) always holds at any point within the Lyapunov sublevel set $X = \{x | V(x) \leq \gamma^2 \Delta\}$. With the third term relaxed and combined with (10.2), (10.5) also implies that in the absence of uncertainties ($w = 0$), the fastest descent of $V(x)$ is such that $dV(x)/dt \leq -V(x)/\tau$, i.e., faster than an exponential decay with time constant τ.

Based on the above discussion, given the dynamic model, Lyapunov function V(x) and the uncertainty level Δ, by choosing a time constant of interest τ for the Lyapunov function in the uncertainty-free case, we can estimate an invariant set $\{x|V(x) \leq \gamma^2\Delta\}$ by finding the minimum L_2-gain γ^2 such that (10.5) holds for any x in the corresponding invariant set. We note that such an estimated invariant set is not strict, since the time-average bound on the Lyapunov function value is a relaxation of its strict upper bound, while the form of X as a sublevel set is conservative. However, these issues cannot be circumvented for general nonlinear dynamics without computationally expensive procedures. This compromise for easier computation in the dynamic flexibility analysis and flexible process design will be validated and amended by dynamic simulations during the application of the proposed method.

10.3 Lyapunov Dynamic Flexibility Analysis

Dynamic flexibility analysis considers two problems, namely (i) given the process design and the magnitude of the uncertainties, test whether the process operation remains feasible; (ii) given the process design, estimate the allowed magnitude of the uncertainties such that the feasibility constraints on process operation are not violated. Now we formulate these problems in the proposed Lyapunov framework. We denote by d the vector of process design decisions. Generally the functions f, g and b in the process model are dependent on d and denoted as $f(x;d)$, $g(x;d)$ and $b(x;d)$, respectively. The constraints for feasible process operation are denoted by the inequality $c(x;d) \leq 0$.

10.3.1 Flexibility test problem

For a given uncertainty level represented by Δ, the smallest L_2-gain is estimated by finding the smallest value of γ^2 such that (10.5) holds for any x with $V(x) \leq \gamma^2\Delta$:

$$\begin{aligned}\max_{x,\gamma} \ \gamma^2 \quad \text{s.t. } & V(x) \leq \gamma^2\Delta \\ & \nabla V(x)^\top f(x;d) - \|\nabla V(x)^\top g(x;d)\|_1 \\ & \quad + \frac{\tau}{4\gamma^2}\nabla V(x)^\top b(x;d)b(x;d)^\top \nabla V(x) + \frac{1}{\tau}V(x) \leq 0\end{aligned} \tag{10.6}$$

where the minimum of γ such that (10.5) holds for any x is equivalently substituted by the maximum of γ such that (10.5) is violated for some x in the corresponding Lyapunov sublevel set $\{x|V(x) \leq \gamma^2\Delta\}$.

By solving the problem in (10.6), we obtain an estimate of the invariant set $\{x|V(x) \leq$

$\gamma^2\Delta\}$. On the other hand, the upper bound of the Lyapunov function value to guarantee operational feasibility, Φ, is solved by

$$\min_{x,\Phi} \ \Phi \quad \text{s.t. } V(x) \leq \Phi, \ \ c(x;d) \not\leq 0. \tag{10.7}$$

The ratio $\Phi/\gamma^2\Delta$ represents the gap between the feasibility boundary and the estimated invariance set of the system, and thus a higher (lower) ratio indicates that the system is more (less) flexible in the presence of parametric uncertainties and disturbances. However, due to the non-exactness of the invariance set construction, and the considerations of the best possible control policy and worst-case uncertainty in the L_2 analysis, it does not necessarily follow that $\Phi/\gamma^2\Delta = 1$ is a watershed between flexibility and inflexibility. Simulations will be needed to validate the process flexibility by sampling multiple scenarios for the uncertainties under a realistic controller.

10.3.2 Flexibility index problem

For the Lyapunov flexibility index problem, we aim to find the maximum allowed extent of uncertainties Δ such that the smallest achievable L_2-gain determined by (10.6) is guaranteed to keep the process operation feasible, with a given gap of R from the feasibility boundary captured by Φ in (10.8). This problem can be expressed as the following bilevel programming problem:

$$\begin{aligned} \max \ \Delta \quad \text{s.t. } & \gamma^2\Delta \leq \Phi/R \\ & \gamma^2 = \max_{x,\gamma} \ \gamma^2\Delta \\ & \qquad \text{s.t. } V(x) \leq \gamma^2\Delta \\ & \qquad\qquad \nabla V(x)^\top f(x;d) - \|\nabla V(x)^\top g(x;d)\|_1 \\ & \qquad\qquad + \frac{\tau}{4\gamma^2}\nabla V(x)^\top b(x;d)b(x;d)^\top \nabla V(x) + \frac{1}{\tau}V(x) \leq 0 \end{aligned} \tag{10.8}$$

or equivalently a single-level program:

$$\begin{aligned} \min_{x,\gamma,\Delta} \ \Delta \quad \text{s.t. } & \gamma^2\Delta \leq \Phi/R, \ \ V(x) \leq \gamma^2\Delta \\ & \nabla V(x)^\top f(x;d) - \|\nabla V(x)^\top g(x;d)\|_1 \\ & + \frac{\tau}{4\gamma^2}\nabla V(x)^\top b(x;d)b(x;d)^\top \nabla V(x) + \frac{1}{\tau}V(x) \leq 0. \end{aligned} \tag{10.9}$$

In (10.9), the parameter R is viewed as a *correction factor* accounting for the gap between the realized upper bound of the Lyapunov function values and the one from the L_2 analysis. After solving (10.9), we can perform simulations under different scenarios of uncertainties of magnitude Δ. If the simulated trajectories are yet deep inside the feasible region, we reduce R to allow larger uncertainties. If a significant portion of the trajectories have escaped beyond the feasibility boundary, the ratio R should be increased to restrict the uncertainties to be smaller.

10.4 Optimal Lyapunov Flexible Design Problem

The goal of the flexibility analysis is to pave the way for the formulation of optimally designing a dynamically flexible process. Given the uncertainty level Δ, we aim to determine the optimal process design d and Lyapunov function $V(x)$ such that the resulting process is dynamically flexible with minimized expected cost under uncertainties. The formulation is given below:

$$\begin{aligned} \min_{d,V} \; & Q(d,V) + \mathbb{E}_{w(t)}\left[\min_{u(t)} \frac{1}{T}\int_0^T P(d,x(t),u(t))dt\right] \\ \text{s.t. } & d \in \mathbb{D},\; V \in \mathbb{V},\; \gamma^2\Delta \le \Phi/R \\ & \Phi = \min_{x,\Phi} \; \Phi, \quad \gamma^2 = \max_{x,\gamma} \; \gamma^2. \end{aligned} \tag{10.10}$$

In (10.10), Q is the capital cost of the process determined by the design decisions d as well as the parameters V that specify the Lyapunov function $V(x)$, whose form may affect the cost of the controller; P is the operating cost depending on the design decisions d and states x, minimized by the control inputs u and averaged over a long time horizon T; $\mathbb{E}$ is the symbol for expectation over uncertainty scenarios. $\mathbb{D}$ stands for the constraints of d, and $\mathbb{V}$ is the range of the parameters V, e.g., if the Lyapunov function is restricted in quadratic forms ($V(x) = x^\top Vx$), then the range $\mathbb{V} = \mathbb{S}_+$ (the set of positive semidefinite matrices). The constraints in the inner level problems are omitted for brevity.

The involvement of an expected operating cost defined based on dynamic trajectories in (10.10)results in a mixed-integer dynamic optimization problem that is also stochastic and bilevel in nature. To reduce the complexity of (10.10), we consider the expected operating cost as a "baseline" operating cost $P(d,V)$ multiplied by a *correction factor* α. The baseline cost is chosen as such that (i) the control input is determined by the formula of Lin and Sontag [230] to guarantee the decay of $V(x)$ faster than $dV(x)/dt \leq -V(x)/\tau$:

$$\kappa(x; d, V) = \begin{cases} -\frac{L_f^* V(x;d) + \sqrt{(L_f^* V(x;d))^2 + \|L_g V(x;d)\|^4}}{\|L_g V(x;d)\|^2 \left(1 + \sqrt{1 + \|L_g V(x;d)\|^2}\right)} L_g V(x; d)^\top, & L_g V(x; d) \neq 0 \\ 0, & L_g V(x; d) = 0 \end{cases}, \tag{10.11}$$

where the Lie derivative notations are defined as $L_g V(x; d) = \nabla V(x)^\top g(x; d)$ and $L_f^* V(x; d) = \nabla V(x)^\top f(x; d) + V(x)/\tau$ respectively, and (ii) the cost is evaluated as the highest possible value in the corrected invariant set. In other words, we reformulate (10.10) into the following form:

$$\begin{aligned} \min_{d,V} \ & Q(d, V) + \alpha P \\ \text{s.t. } & d \in \mathbb{D}, \ \ V \in \mathbb{V}, \ \ \gamma^2 \Delta \leq \Phi/R \\ & P = \max_{\bar{x}} \ \ P(d, \bar{x}, \kappa(\bar{x}; d, V)) \\ & \Phi = \min_{\check{x}, \Phi} \ \ \Phi \ \text{ s.t. } \ V(\check{x}) \leq \Phi, \ \ c(\check{x}; d) \nleq 0 \\ & \gamma^2 = \max_{\hat{x}, \gamma} \ \ \gamma^2 \ \text{ s.t. } \ V(\hat{x}) \leq \gamma^2 \Delta \\ & \qquad\qquad\qquad V(\hat{x})^\top f(\hat{x}; d) - \|\nabla V(\hat{x})^\top g(\hat{x}; d)\|_1 \\ & \qquad\qquad\qquad\quad + \frac{\tau}{4\gamma^2} \nabla V(\hat{x})^\top b(\hat{x}; d) b(\hat{x}; d)^\top \nabla V(\hat{x}) + \frac{1}{\tau} V(\hat{x}) \leq 0. \end{aligned} \tag{10.12}$$

The bilevel programming problem (10.12) can then be solved via a typical transformation into a mixed-integer nonlinear program using the Karush-Kuhn-Tucker (KKT) optimality conditions involving dual and slack variables [141]. We note that since (10.12) is a bilevel program with nonconvex inner problems, global optimality is not guaranteed by solving the transformed problem. However, due to the use of correction factors R and α, the suboptimality can be accounted for. The quality of the solution to (10.12) is critically dependent on the choice of the correction factors. We propose that R and α should be iteratively updated with a posteriori simulations. Specifically, in each iteration, the optimal flexible process design problem (10.12) is first solved under the given R and α to determine the optimal design decisions d and V. Then, a number of scenarios of the uncertainties are sampled (given the magnitude level Δ) and for each scenario, the system is simulated under the user-specified desirable control strategy (which can differ from (10.11), e.g., MPC or distributed MPC) to obtain the corresponding trajectories. The Lyapunov function values of the states on the simulated trajectories are compared to the Φ value given by (10.8) to update R, and the average cost of the simulated trajectories is compared to the "baseline" operating cost $P(d, V)$ to update α. The iterations proceed until the correction factors have converged.

Compared to the typical formulations of the integrated design and control problem, in our proposed approach, the optimization problem (10.12) has a deterministic and static form. The dynamic simulations under uncertain scenarios are separated from the optimization formulation and serve only as a means of adjusting the correction factors that account for the inexactness of L_2-gain analysis, deviations of actual control from the baseline policy, and bilevel programming sub-optimality. The proposed formulation is therefore computationally efficient. Also, since we do not require model linearization or limit our scope to any specific type of controllers, this approach is generally applicable to nonlinear process systems under any nonlinear control.

10.5 Illustrating Example

We examine the proposed framework with a case study on a two-reactor system with one or two feed streams (split from a total feed). The detailed model was given in [359] and adapted here without considering the dynamics of holdup volume. Here the existence of the second reactor ($d_1 = 0$ or 1), the existence of the second feed stream ($d_2 = 0$ or 1), the portion of second feed stream in the total feed ($0 \leq d_3 \leq 1$), the reactor volumes (d_4, d_5) and the operating steady state are the design variables. The 4 states include the deviations of the reactor temperatures and reactant concentrations from the steady state. The deviations of the cooling rates are used as 2 control inputs. A disturbance exists in the feed temperature. We scale the control inputs by 1 MJ/min, states by 20 K, and 0.01 mol/L and the disturbance by 20 K. The control inputs are assumed to be constrained within ± 1. The time constant τ for the fastest convergence of the nominal system is chosen as 3 sec.

We first perform a dynamic flexibility analysis given the design as $d_1 = 1$, $d_2 = 1$, $d_3 = 0.52$ and the Lyapunov function $V(x) = x^\top x$ ($\Phi = 0.6289$). For $\Delta = 1$, we obtain from the dynamic flexibility test problem (10.6) that $\gamma^2 = 6.45 \times 10^{-4}$, giving an estimated invariant set with the value of $V(x)$ upper bounded by 6.45×10^{-4}, which implies that the process is highly flexible under the disturbance level $\Delta = 1$ if the

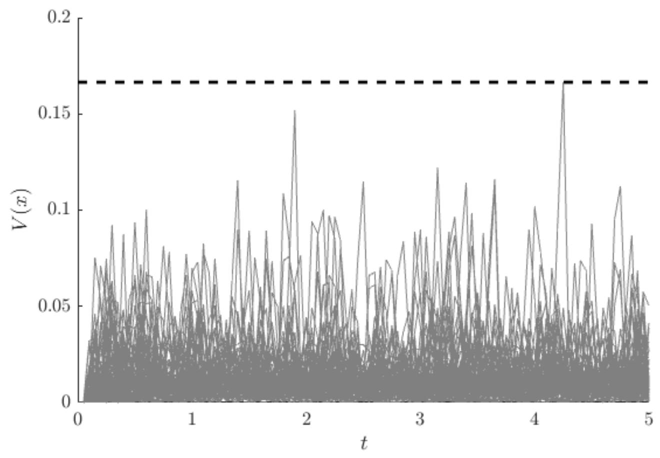


Figure 10.1: Lyapunov function values and the practical upper bound under disturbance level $\Delta = 1$.

control policy is the ideal one that perfectly results in the fastest achievable decay of the Lyapunov function. However, such a perfect controller is unrealistic since it is noncontinuous and the control inputs are almost always saturated. Here we will use a well-tuned model predictive controller combined with moving horizon estimation using the reactor temperatures as the observations [8]. Fig. 10.1 shows the simulated values of $V(x)$ under such a controller for 100 independent bounded disturbance signals ($|w| \leq \Delta$) in 5 time scales. It can be observed that the practical upper bound of $V(x)$ is approximately 0.1640. This suggests a correction factor $R = 0.1664/(6.45\times10^{-4}) = 258$. Still, the process remains highly feasible as the upper bound of $V(x)$ is much lower than Φ.

Next, we consider the flexibility index problem that determines the largest allowable disturbance level without affecting the feasibility of process operation. The process design d and the Lyapunov function $V(x)$ are the same as before. The initial guess of the correction factor is the one from the feasibility test problem $R = 258$. After every iteration of solving Δ from (10.9), we update the correction factor through the simulation under the afore-mentioned model predictive control with 100 independent samples of the disturbance signal using the shrinking step-size rule $R_{i+1} := R_i + (R_{\text{new}} - R_i)/i$ (i is the iteration number). The changes of disturbance level Δ and the Lyapunov function value $V(x)$ with increasing number of iterations are shown in Fig. 10.2. Within 20 iterations, the simulated upper bound of $V(x)$ approaches close to the level corresponding to the

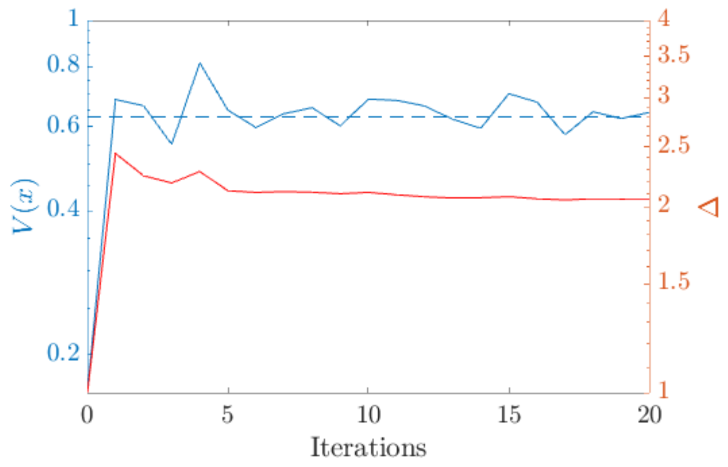


Figure 10.2: Maximum allowable disturbance level and Lyapunov function value.

boundary of operational feasibility $\Phi = 0.6289$, which determines the maximum allowed disturbance level for maintaining operational feasibility as approximately $\Delta = 2.06$.

Finally, we consider the problem of finding the optimal design of the process d and the Lyapunov function $V(x)$ by solving (10.12) under the given disturbance level $\Delta = 1$. For simplicity we consider quadratic Lyapunov functions $V(x) = x^\top V x$, where V is a positive semidefinite matrix without terms involving terms from two different reactors:

$$V = \begin{bmatrix} v_{11} & v_{13} & 0 & 0 \\ v_{13} & v_{12} & 0 & 0 \\ v_{21} & v_{23} & 0 & 0 \\ v_{23} & v_{22} & 0 & 0 \end{bmatrix}$$

satisfying $v_{i1}, v_{i2} \geq 0$ and $v_{i1}v_{i2} - v_{i3}^2 \geq 0$, $i = 1, 2$. In the economic objective function, the capital cost Q is specified by the reactor volumes (d_4, d_5), and the operating cost P is specified by the substrate concentrations (C_1, C_2) in the reactors as in [359], but with an additional term accounting for the utility cost of the heat exchange rates with the reactors (Q_1, Q_2):

$$\begin{aligned} Q &= 1.375(d_4^{0.6227} + d_1 d_5^{0.6227}), \\ P &= 10^4|(1-d_1)C_1 + d_1 C_2 - 0.01| + 2.5((1-d_1)Q_1 + d_1 Q_2). \end{aligned} \tag{10.13}$$

The steady state design is constrained so that the conversion equals 99%, i.e., $(1-d_1)C_1+$

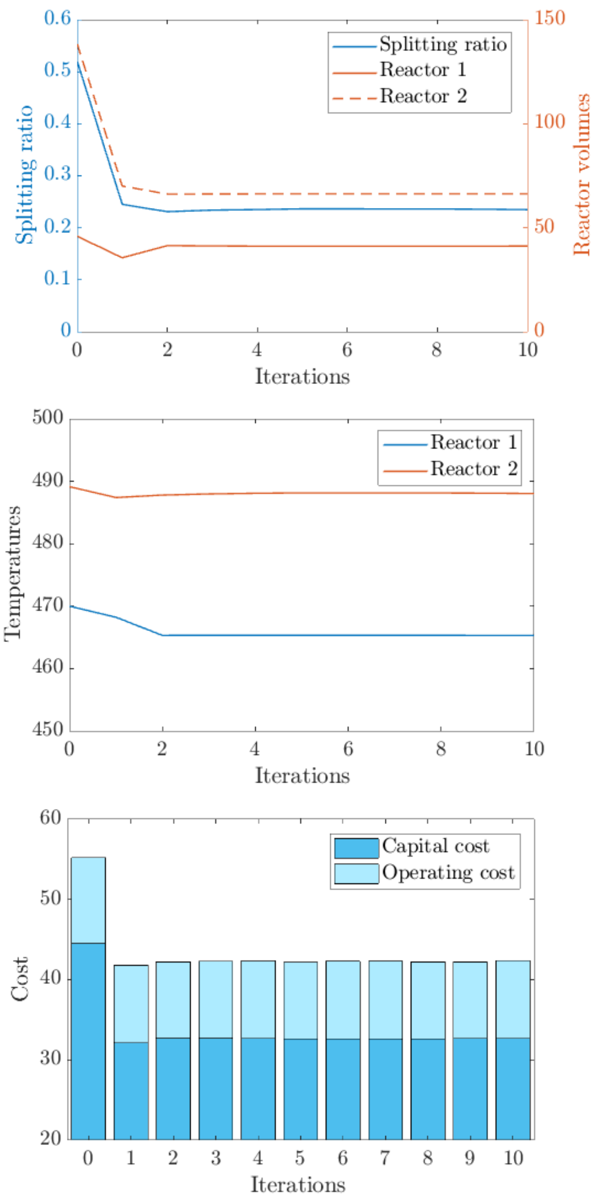


Figure 10.3: Splitting ratio, reactor volumes (left), steady-state reactor temperatures (middle), capital and operating costs (right).

$d_1 C_2 = 0.01$. The initial guess of the correction factors are chosen as $R = 300$ and $\alpha = 1$, and during the iterations they are updated with shrinking step-sizes. Fig. 10.3 shows the change in the solution of the optimal design problem throughout the iterations. A converged solution is obtained within a few iterations. This optimal solution requires that both reactors should be in use, the splitting ratio should be changed to about 0.236, and the second reactor should be reduced to a half smaller volume. Compared to the initial guess, the optimal design is found to have approximately 23% less total cost.

10.6 Conclusions

In this work, we have proposed a novel flexibility analysis framework based on the concepts of Lyapunov control theory, where the effect of the uncertainties on the system states are evaluated based on the resulting averaged Lyapunov function value. This approach is computationally efficient compared to the classical MIDO-based approach, while accurately accounting for the intrinsic nonlinearity of process systems dynamics compared to the controllability index-based approaches. Especially, we adopt correction factors to redeem any inexactness and suboptimality that may arise in the derivations and computations, which makes our Lyapunov flexibility analysis generally applicable regardless of the actual controller type although based on hypothetical Lyapunov-based controllers.

While the proposed framework shows a promising direction of integrated process design and control, for process systems on a larger scale, the implementation of our proposed approach is not yet apparent due to the following two reasons. First, a detailed nonlinear dynamic model is required in the current method but can be implicit (i.e., a black-box model) in a flowsheet simulation environment. Second, it becomes difficult to appropriately parameterize and optimize the Lyapunov function involving states in high dimensions. These issues will be further investigated in our future studies.

Part III
Intelligence: Utilizing Process Data to Learn Essentially Control-Relevant Information

The old man had his zeal of youth revived,	老夫聊發少年狂，
leashing a yellow hound on the left side,	左牽黃，
and a falcon in gray on the right.	右擎蒼，
Dressed in silk hats and sable coats,	錦帽貂裘，
thousands of horsemen swept the mountain roads.	千騎卷平岡。
In response to the whole town of people joining the mayor,	爲報傾城隨太守，
He is to hunt down himself a tiger,	親射虎，
to present his young General's vigor.	看孫郎。

Su Shi, "*A Tune of Jiang Cheng Zi: Hunting at Mizhou*" (excerpt), 1075.
蘇軾《江城子·密州出獵》

Chapter 11
Lie-Sobolev Nonlinear System Identification and Its Effect on Nonlinear Control

11.1 Introduction

The development of nonlinear control methods has been one of the most important topics in process control due to the intrinsic nonlinearity of process systems. Examples include input–output linearization [173], which uses state feedback to cancel out the nonlinearity and shape the output response, and model predictive control (MPC) [344], which generates control signals by optimizing a cost associated with the predicted trajectory. It is self-evident that the successful application of these nonlinear model-based control methods is intrinsically dependent on high-quality dynamic models. Process systems may be represented as white-box first-principles models, black-box models of completely unknown dynamics, or grey-box models in between [384]. Whenever a perfect white-box model is unavailable, the unknown parts of the underlying dynamics must be inferred from *system identification* procedures. Despite the complexity of identification-control interaction and integration (see e.g., discussions on separation principle [14], adaptive control [106], or dual control [111]), the quality of control strategies generally benefits from more accurate identification methods.

The specific meaning of system identification may vary with the context. In general, system identification may refer to any regression or data-driven characterization of the unknown parts in dynamic models, such as state-space dynamics, transfer functions, and autoregressive models [367]. A wide spectrum of approaches have been developed in this sense in the process control literature [91, 489, 108, 382]. In a broader sense, the identification can be performed in a model-free manner only to learn useful control-relevant information from data, such as optimal value/policy functions or dissipativity parameters [412, 411, 414]. Considering nonlinear chemical processes, the aim of system identification is typically to estimate the unknown parameters, usually physical and chemical properties, in models of certain a priori structures derived from first principles or approximations [99, 475]. Also, for chemical processes there usually exist states that are not directly measurable and the parameter estimation should be combined with state observation, i.e., both dynamic states and model parameters should be estimated. In this chapter, we consider system identification as the problem of designing such an *observer-estimator*.

State observer design for dynamic systems is a classical problem in process control [387, 90, 203], which can in principle be extended to combined observer-estimator design by viewing parameters as invariant states (see e.g., [208]). The most traditional approach is to modify the Kalman filter into an extended or unscented Kalman filter (EKF, UKF) [383]. For specific types of nonlinear systems, the observer-estimator can be designed through backstepping or following the linear Luenberger observer procedures [207, 107, 425]. Kazantzis and Kravaris [191] formulated a generic nonlinear observer, which guarantees convergence but demands computationally intractable solution of partial differential equations. Implicit schemes based on nonlinear optimization, such as maximum likelihood estimator (MLE) [366] and especially moving horizon estimator (MHE) [341], have gained increasing applications in nonlinear processes.

In this work we focus on the role of derivatives of the model functions in system identification. We argue that they constitute essential *control-relevant information* that is of decisive importance to the resulting control performance. This is apparent in input–output linearizing control, where the control laws are directly constructed by *Lie derivatives*. For MPC, the impact of derivatives can be implied from the local Chen-Fliess series expansions [173, Section 3.2] of the predicted trajectories, whose coefficients rely on the Lie derivatives of the nonlinear model. In fact, under certain constructions these two control strategies can be considered as equivalent [389]. Identification procedures that only seek to match the estimated model with the actual model in the directly measured input and output values may not be effective for matching the Lie derivatives. Moreover, since structural errors generally exist, i.e., the true dynamics may not be exactly in the presumed model structure, explicitly accounting for Lie derivatives can help suppress the effect of structural errors through the estimation of Lie derivatives.

Motivated by the above, in this chapter, we propose a *Lie-Sobolev* framework for system identification, which accounts for the differences between the estimated model and the actual model in their direct input–output trajectories *as well as* Lie derivatives. We will start in Section 11.2 with a simple example to motivate system identification with derivative considerations, give the general definition of Lie-Sobolev system identification, and prove its improved performance with a Luenberger observer for linear systems. The Lie-Sobolev formulations of an explicit gradient descent observer-estimator and an MHE are derived, and their convergence properties as well as their effects on nonlinear control are discussed in Section 11.3 and Section 11.4, respectively. The advantages of the Lie-Sobolev approaches are demonstrated by the application to simple numerical examples and a glycerol etherification reactor with complex dynamics in Section 11.5. Regarding related prior works, we note that the identification of linearly parametrized models with measurable states accounting for first-order derivatives was discussed in [302], and the Sobolev training of neural networks, where the errors in direct and derivative values together contribute to the back-propagation, was proposed in [333, 72].

11.2 Lie-Sobolev System Identification

11.2.1 A motivating example

We motivate our Lie-Sobolev perspective with a simple dynamic system example:

$$\dot{x} = f(x) + u, \tag{11.1}$$

where $f : \mathbb{R} \to \mathbb{R}$ is an unknown odd function defined by $f(x) = e^{-x} \sin 2\pi x$ on $x \in [0, +\infty)$, which we want to drive from an open-loop stable equilibrium point $x = 1$ to an open-loop unstable equilibrium point $x = 0$. We approximate the model function f with a 5th-order polynomial with a zero constant term on $[0, 1]$, whose derivative function is expected to be an approximation of $f'(x) = \frac{df}{dx}(x)$:

$$\hat{f}(x|\theta) = \theta_5 x^5 + \theta_4 x^4 + \cdots + \theta_1 x, \;\; \hat{f}'(x|\theta) = \frac{d\hat{f}}{dx}(x|\theta) = 5\theta_5 x^4 + 4\theta_4 x^3 + \cdots + \theta_1. \tag{11.2}$$

For model identification, $n = 100$ sample points $\{x^{(i)}\}$ are drawn under the uniform distribution on $[0, 1]$ and the corresponding f values are measured with independently identically distributed (i.i.d.) Gaussian noises from $\mathcal{N}(0, 0.2^2)$. First-order derivative values are also measured with i.i.d. noises from $\mathcal{N}(0, 0.2^2)$. The ordinary least-squares regression gives an estimation of $\theta = (\theta_1, \dots, \theta_5)$ by minimizing the sum of squared distances between the measured and predicted function values,

$$\hat{\theta}_L = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \hat{f}(x^{(i)}|\theta) \right)^2. \tag{11.3}$$

The objective function can be viewed as an estimation of the squared L^2-norm of $f - \hat{f}$:

$$\|f - \hat{f}\|^2_{L^2([0,1])} = \int_0^1 (f(x) - \hat{f}(x|\theta))^2 dx. \tag{11.4}$$

With a Sobolev perspective, the least-squares regression is modified by taking into account the distance between the measured and predicted derivatives with a weighted sum:

$$\hat{\theta}_W = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left[\left(y^{(i)} - \hat{f}(x^{(i)}|\theta) \right)^2 + \left(y'^{(i)} - \hat{f}'(x^{(i)}|\theta) \right)^2 \right]. \tag{11.5}$$

This objective function is now an estimation of the squared norm of $f - \hat{f}$ on the Sobolev space $W^{1,2}([0,1])$[1]:

$$\|f - \hat{f}\|^2_{W^{1,2}([0,1])} = \int_0^1 \left[(f(x) - \hat{f}(x|\theta))^2 + (f'(x) - \hat{f}'(x|\theta))^2 \right] dx. \tag{11.6}$$

The performance of $\hat{\theta}_L$ and $\hat{\theta}_W$ is shown by the comparison between the estimated $\hat{f}(x|\theta)$ and true $f(x)$ and between their derivatives in Fig. 11.1(a) and 11.1(b), respectively. In contrast to $\hat{\theta}_L$ which leaves considerably large errors in the derivative, the Sobolev estimator significantly reduces the derivative errors. In fact, an estimation procedure that considers only the original function values may not provide any guarantee for the resulting derivatives, due to the lack of an upper bound of Sobolev $W^{k,p}$-norms ($k \geq 1$) on L^p-, namely $W^{0,p}$-norms.[2]

We specify the ideal control law as $u = -f(x) - 0.5x$, so that the closed-loop system is exponentially convergent to the origin by $\dot{x} = -0.5x$. The actual closed-loop system with the identified model is $\dot{x} = f(x) - \hat{f}(x|\theta) - 0.5x$. Under the ordinary least-squares estimator $\hat{\theta}_L$ and Sobolev estimator $\hat{\theta}_W$, we simulate the closed-loop system from $x(0) = 1$. The resulting trajectories are shown in Fig. 11.1(c). It can be seen that the Sobolev estimation gives trajectories that are closer to the target trajectory, while the ordinary estimation may not be able to guarantee asymptotic convergence to the origin. This is due to the role of derivative in guaranteeing asymptotic convergence to the origin,

[1]In general, a Sobolev space $W^{k,p}(\Omega)$ defined on a set $\Omega \subseteq \mathbb{R}^d$ is the collection of all functions on Ω that are differentiable up to k-th order, giving derivatives that are L^p-integrable on Ω, i.e.,

$$W^{k,p}(\Omega) = \Bigg\{ g : \Omega \to \mathbb{R} | \forall \alpha_1, \dots, \alpha_d \in \mathbb{N} \text{ with } \alpha_1 + \cdots + \alpha_d \leq k,$$
$$\frac{\partial^{\alpha_1 + \cdots + \alpha_d} g}{\partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}} \text{ exists and } \int_\Omega \left| \frac{\partial^{\alpha_1 + \cdots + \alpha_d} g}{\partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}}(x) \right|^p dx < +\infty \Bigg\}.$$

[2]For example, let $g_m(x) = \frac{1}{m} \sin 2\pi m x$ ($m \in \mathbb{N}$, $m \geq 1$) with $g'_m(x) = 2\pi \cos 2\pi m x$. Then we have $\|g_m\|^2_{L^2([0,1])} = \frac{1}{2m^2}$ and $\|g_m\|^2_{W^{1,2}([0,1])} = \frac{1}{2m^2} + 2\pi^2$, leading to a ratio of $W^{1,2}$-norm and L^2-norm of $\|g_m\|_{W^{1,2}([0,1])} / \|g_m\|_{L^{1,2}([0,1])} \geq 2\pi m$, which can become arbitrarily large with varying m.

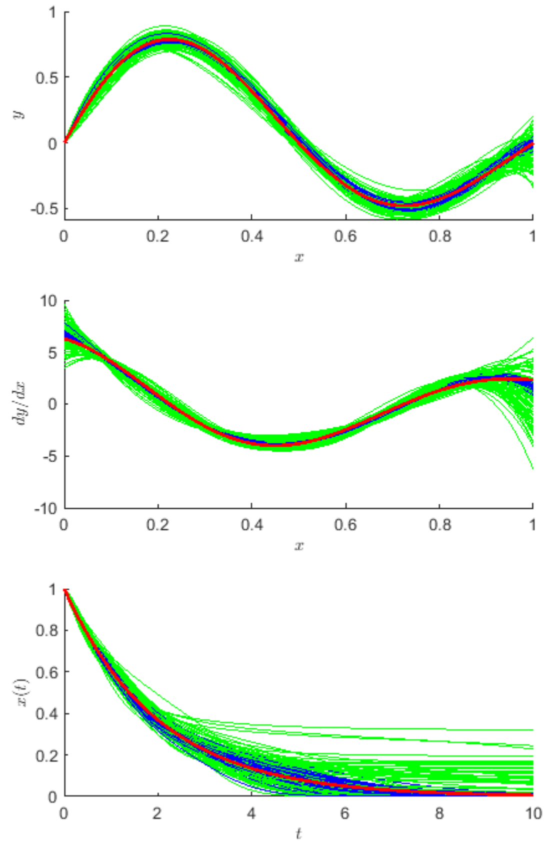


Figure 11.1: The identified $\hat{f}(x|\theta)$, $\hat{f}'(x|\theta)$, and simulated closed-loop trajectories under ordinary and Lie-Sobolev estimation in 100 independent experiments. The green and blue curves correspond to $\hat{\theta}_L$ and $\hat{\theta}_W$, respectively, and the red curves stand for the true functions $f(x)$, $f'(x)$ and the target trajectory.

i.e., the approximation error in the derivative should satisfy $|f'(x)-\hat{f}'(x|\theta)| < 0.5$, and a smaller derivative approximation error implies smaller deviation from the target system.

It should be highlighted that the combination of the original L^2-objective and the derivative term does not necessarily result in a tradeoff of two conflicting criteria, as shown by the comparison of green and blue curves in Fig. 11.1(a). In other words, a better derivative estimation can contribute to a better estimation of the original function. This can be partly understood by the classical Poincaré inequality, which establishes the existence of a positive constant c such that $\|g\|_{L_2} \leq c\|g'\|_{L^2}$ for any $g \in W^{1,2}$, thus bounding the L^2-norm with the $W^{1,2}$-norm for functions g equal to zero on the boundary. Even considering any function g on $[0,1]$ with only one side $g(0) = 0$

fixed, we have

$$\|g\|_{L^2}^2 = \int_0^1 \left(\int_0^x g'(s)ds \right)^2 dx \leq \int_0^1 \|g'\|_{L^2}^2 x dx = \frac{1}{2}\|g'\|_{L^2}^2 \tag{11.7}$$

where the inequality results from Cauchy-Schwarz, and hence

$$\|g\|_{L^2}^2 \leq \|g\|_{W^{1,2}}^2/3. \tag{11.8}$$

Therefore, a Sobolev estimator can be seen as an "indirect" estimator in the L^p-sense, which has potential to improve the identification performance by exploiting the global relations linking the direct L^p-approximation error and approximation errors in the higher-order derivative functions.

11.2.2 System identification with observer-estimator

Different from the above motivating example where we estimate the function $f(x)$ from static sample data, for control purposes we use historical measurement data to identify a nonlinear dynamic model:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \;\; y(t) = h(x(t)), \tag{11.9}$$

where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^{d_x}$, $u(t) \in \mathcal{U} \subseteq \mathbb{R}^{d_u}$ and $y(t) \in \mathbb{R}^{d_y}$ are the vectors of states, inputs and outputs, respectively. $f : \mathcal{X} \to \mathbb{R}^{d_x}$, $g : \mathcal{X} \to \mathbb{R}^{d_x \times d_u}$ and $h : \mathcal{X} \to \mathbb{R}^{d_y}$ are supposed to be smooth functions but may not be completely known, and hence need to be approximated within parameterized families of smooth functions $\{(\hat{f}(x|\theta), \hat{g}(x|\theta), \hat{h}(x|\theta))|\theta \in \Theta\}$, where $\theta \in \Theta \subseteq \mathbb{R}^{d_\theta}$ is a vector of parameters. Suppose that θ is estimated in an adaptive way. That is, excitation signals are imposed after time 0, and the inputs u and outputs y are measured throughout time. At any specific time t, the historical measurements in the time interval $[0, t]$ are available for deriving an estimation $\hat{\theta}(t)$, which is updated with new measurements. Such an *adaptive parameter estimator* is generally formulated as

$$\hat{\theta}(t) = \pi(\{y(s), u(s), \hat{\theta}(s)|0 \leq s < t\}) \tag{11.10}$$

where $\hat{\theta}$ is an estimation of θ varying with time, and π is an adaptation law (also a functional) to update the parameter estimate according to the past estimates and historical measurements in the inputs and outputs [459]. Once the parameter estimation is completed, a control law (in the form of a functional κ for generality)

$$u(t) = \kappa(\{y(s), u(s), \hat{\theta} | 0 \leq s < t\}) \tag{11.11}$$

can be designed to shape the closed-loop trajectory.

In a more concrete form, for the system (11.9) as a state-space model, the parameter estimation is described by differential equations on $t \geq 0$, and is often realized with an accompanying state observer as a dynamic system that imitates the plant. For nonlinear systems, it is more convenient to observe $\dot{x}(t)$ in addition to $x(t)$, and hence the observer is allowed to have a second-order dynamics (denoted as σ). Here we consider the case where the state observer σ and the parameter estimator π are both ordinary differential equations (ODEs) using the historical information of y derivatives and u and the current observations and estimates:

$$\begin{aligned}
\dot{\hat{x}}(t) &= \hat{v}(t) \\
\dot{\hat{v}}(t) &= \sigma\left(\hat{x}(t), \hat{v}(t), \hat{\theta}(t), \left\{\{y^{(r)}(s)\}_{r=0}^{r_y}, u(s), \middle| 0 \leq s < t\right\}\right) \\
\dot{\hat{\theta}}(t) &= \pi\left(\hat{x}(t), \hat{v}(t), \hat{\theta}(t), \left\{\{y^{(r)}(s)\}_{r=0}^{r_y}, u(s), \middle| 0 \leq s < t\right\}\right).
\end{aligned} \tag{11.12}$$

As the control law κ is usually constructed based on the estimated model $(\hat{f}, \hat{g}, \hat{h})$, key to the construction of the observation and estimation laws (σ, π) is the matching of the trajectory of $\hat{x}(t)$ and $\hat{\theta}(t)$ to the behavior of the plant (11.9) in terms of the measured historical data. A perfect identification refers to a pair (σ, π) that makes the following ODEs hold for all $t \geq 0$:

$$\dot{\hat{x}}(t) = \hat{f}(\hat{x}(t)|\hat{\theta}(t)) + \hat{g}(\hat{x}(t)|\hat{\theta}(t))u(t), \;\; y(t) = \hat{h}(\hat{x}(t)). \tag{11.13}$$

11.2.3 Lie-Sobolev identification

For a Sobolev-type identification of the dynamic system (11.9) from input and output historical trajectories, we review the definition of Lie derivatives in nonlinear control [173, Chapter 4].

Definition 11.1 (Lie derivative)**.** *The Lie derivative of the i-th component of h, h_i, with respect to f and g are defined as*

$$L_f h_i = \frac{\partial h_i}{\partial x} f, \;\; L_f g = \frac{\partial h_i}{\partial x} g, \tag{11.14}$$

respectively, where $\partial h_i/\partial x \in \mathbb{R}^{1\times d_x}$. The Lie differentiation operators can be recursively composed to generate high-order Lie derivatives, e.g., $(L_f L_g)h_i = L_f(L_g h_i)$. Denote $L_f^{k+1} = L_f L_f^k$, $k = 0, 1, \dots$, with L_f^0 being identity.

The Lie derivatives capture the instantaneous response of the outputs, as $\dot{y}_i = L_f h_i + (L_g h_i)u$, $\ddot{y}_i = L_f \dot{y}_i + (L_g \dot{y}_i)u$, ...

Definition 11.2 (relative degree)**.** *The relative degree ρ_i for the i-th output is the smallest positive integer r such that $L_g L_f^{r-1} h_i \not\equiv 0$.*

With relative degree ρ_i known, we have

$$\begin{aligned} y_i(t) &= L_f^0 h_i(x(t)) \\ \dot{y}_i(t) &= L_f^1 h_i(x(t)) \\ &\cdots \\ y_i^{(\rho_i)}(t) &= L_f^{\rho_i} h_i(x(t)) + L_g L_f^{\rho_i - 1} h_i(x(t))u(t) \end{aligned} \tag{11.15}$$

which implies that the direct effect of inputs u falls on the ρ_i-th time derivative of t. For controlling the system (11.9) by shaping the responses of $y_i^{(\rho_i)}$, $y_i^{(\rho_i-1)}$, ..., y_i, accurately evaluating or managing the errors in approximating the Lie derivatives $L_f^r h_i(x)$, $r = 0, 1, \dots, \rho_i$ and $L_g L_f^{\rho_i-1} h_i(x)$, $i = 1, \dots, n_y$ is thus of crucial importance. Therefore, we propose that the estimator-observer (π, σ) should be defined in a Lie-Sobolev manner, aiming at matching not only the output values according to (11.13) but also the Lie derivatives by:

$$\begin{aligned} y_i(t) &= L_{\hat{f}}^0 \hat{h}_i(\hat{x}(t)|\hat{\theta}(t)) \\ \dot{y}_i(t) &= L_{\hat{f}}^1 \hat{h}_i(\hat{x}(t)|\hat{\theta}(t)) \\ &\cdots \\ y_i^{(\rho_i)}(t) &= L_{\hat{f}}^{(\rho_i)} \hat{h}_i(\hat{x}(t)|\hat{\theta}(t)) + L_{\hat{g}} L_{\hat{f}}^{\rho_i-1} \hat{h}_i(\hat{x}(t)|\hat{\theta}(t))u(t). \end{aligned} \tag{11.16}$$

Definition 11.3 (Lie-Sobolev identification)**.** *Lie-Sobolev identification refers to the design of an observer-estimator law (σ, π) based on not only $\hat{f}$, $\hat{g}$ and $\hat{h}$ evaluated at the observed states and estimated parameters, but also Lie derivatives including $L_{\hat{f}}^r \hat{h}_i$, $r = 1, \dots, \rho_i$ and $L_{\hat{g}} L_{\hat{f}}^{\rho_i-1} \hat{h}_i$, $i = 1, \dots, d_y$, assuming that all the relative degrees are known a priori.*

As motivated in the previous subsection, the purpose of using Lie-Sobolev identification instead of an ordinary one is two-fold. First, the Lie derivatives are explicitly accounted for and their approximation errors can be better restricted, which may be necessary in constructing control laws κ. Second, there is the potential of improving the identification performance of the original model by taking a roundabout route through a Sobolev space. Let us show this advantage through a Luenberger observer for linear systems.

11.2.4 Lie-Sobolev Luenberger observer for linear systems

Consider a linear system with a Luenberger observer[3]

$$\dot{x} = Ax + Bu, \;\; y = Cx, \;\; \dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x}), \tag{11.17}$$

where the observer gain matrix L is given by $L = YC^\top V^{-1}$ with $Y \succeq 0$ being the solution of the algebraic Riccati equation

$$YA^\top + AY - YC^\top V^{-1}CY + W = 0. \tag{11.18}$$

Here $W \succ 0$ and $V \succ 0$ are tunable matrices of appropriate dimensions. Let the observation error be $e = x - \hat{x}$. Then we have $\dot{e} = (A - LC)e$, in which $A - LC \prec 0$. The performance of the observer can be characterized by the eigenvalues of $\bar{A} + \bar{A}^\top$ with $\bar{A} = A - LC$, which reflects the rate of decay of the observation errors. The Lie-Sobolev modification of the Luenberger observer is to augment the outputs with their time derivatives, i.e., to supplement the matrix C with rows $c_i A^r$ (c_i is the i^{th} row of C) and V with additional diagonal entries $v_i^r > 0$ for $r = 1, \ldots, \rho_i$, $i = 1, \ldots, d_y$.

Denote the modified C and V by $\tilde{C}$ and $\tilde{V}$, respectively. Then $\tilde{C}^\top \tilde{V}^{-1}\tilde{C} - C^\top V^{-1} C = \sum_{i=1}^{d_y} \sum_{r=1}^{\rho_i} c_i^\top (A^{r-1})^\top A^{r-1} c_i / v_i^r \succeq 0$. One can show that in (11.18), when $\Upsilon = C^\top V^{-1} C$ is perturbed to $\Upsilon + \delta\Upsilon$ with $\delta\Upsilon$ being an infinitesimally small symmetric matrix (by setting v_i^r at sufficiently large values), the corresponding perturbation in $\bar{A}$, denoted by $\delta\bar{A}$, satisfies the following algebraic Lyapunov equation

$$\delta\bar{A} \cdot \Upsilon^{-1}\bar{A}^\top + \bar{A}\Upsilon^{-1} \cdot \delta\bar{A}^\top = \bar{A}\Upsilon^{-1} \cdot \delta\Upsilon \cdot \Upsilon^{-1}\bar{A}^\top - A\Upsilon^{-1} \cdot \delta\Upsilon \cdot \Upsilon^{-1}A^\top \tag{11.19}$$

[3]We consider the parameters as time-invariant states implicitly, and due to the linearity, the dynamics of $\hat{x}$ in the Luenberger observer is first-order, different from the second-order dynamics in (11.12).

From (11.18) it can be inferred that $\bar{A}\Upsilon^{-1}\bar{A} - A\Upsilon^{-1}A = W \succ 0$. If $m_1\Upsilon \preceq \delta\Upsilon \preceq m_2\Upsilon$ for some $m_1, m_2 > 0$ with $m_2/m_1 \leq 1 + \lambda_{\min}(W)/\lambda_{\max}(A\Upsilon^{-1}A)$, then the right-hand side above is positive semidefinite. Since $\Upsilon^{-1}\bar{A}^\top$ is Hurwitz, $\delta\bar{A} + \delta\bar{A}^\top \preceq 0$. The Lie-Sobolev modification therefore leads to a higher-gain observer and hence improves the convergence of the observation errors. For the case with structural errors, the readers are referred to a brief discussion on an analogous condition for the improved performance of Lie-Sobolev Luenberger observer in Appendix C.1.

In the next two sections, we formulate the Lie-Sobolev observer-estimator for nonlinear systems based on gradient descent, which underlies the majority of adaptive parameter estimation schemes [117]. Specifically, a real-valued criterion J is defined based on the state observations $\hat{x}(t)$, their time derivatives $\hat{v}(t) = \dot{\hat{x}}(t)$ and parameter estimates $\hat{\theta}(t)$, given the measured inputs, outputs and output derivatives. The update rules are designed such that the time derivative of J is made as negative as possible. Depending on the way that $J(t)$ is defined, we formulate two different Lie-Sobolev identification schemes.

- In the first type, $J(t)$ is defined based only on the current estimates and measurements. We view this explicit gradient-based identification scheme as a prototype approach, for which theoretical properties can be analyzed with classical Lyapunov arguments.
- The second type – MHE, is an implicit observer-estimator formulated as an optimization problem involving the current and past measurements. Its convergence properties are established in a similar way to the ordinary MHE whose analysis has been covered in the recent literature.

In nonlinear cases, the advantage of Lie-Sobolev nonlinear system identification is difficult to prove, and is examined in numerical studies in Section 11.5. In the sequel it is always assumed that the relative degrees are known and unaffected by the parameterization of f, g and h.

11.3 Lie-Sobolev Gradient Descent Observer-Estimator

11.3.1 Derivation

Consider the following function $J(t)$, which accounts for the residuals on the equalities for the perfectly identified model as characterized by (11.13) and (11.16) in a Lie-Sobolev setting:

$$\begin{aligned}
J(t) &= Q(\hat{x}(t), \dot{\hat{x}}(t), \hat{\theta}(t)) \\
&= \frac{1}{2}\left\|\dot{\hat{x}}(t) - \hat{f}(\hat{x}(t)|\hat{\theta}(t)) - \hat{g}(\hat{x}(t)|\hat{\theta}(t))u(t)\right\|^2 + \frac{1}{2}\sum_{i=1}^{d_y}\left[\sum_{r=0}^{\rho_i-1} w_i^r \left\|y_i^{(r)}(t) - L_{\hat{f}}^r \hat{h}_i(\hat{x}(t)|\hat{\theta}(t))\right\|^2\right. \\
&\quad \left. + w_i^{\rho_i}\left\|y_i^{(\rho_i)}(t) - L_{\hat{f}}^{\rho_i}\hat{h}_i(\hat{x}(t)|\hat{\theta}(t)) - L_{\hat{g}}L_{\hat{f}}^{\rho_i-1}\hat{h}_i(\hat{x}(t)|\hat{\theta}(t))u(t)\right\|^2\right]
\end{aligned} \tag{11.20}$$

where the weights w_i^r corresponding to the response of $y_i^{(r)}$, $r = 0, 1, \dots, \rho_i$, $i = 1, \dots, d_y$ are positive constants. If the summation over r contains only $r = 0$ and the last term is not contained either, then the observer-estimator is non-Lie-Sobolev. The criterion function $J(t)$ is analogous to the squared Sobolev norm objective to be minimized in the motivating example (11.6), which equals zero whenever the identified model parameters is equivalent to the true model and state observation is error-free. We call this measure for the model identification performance as the Lie-Sobolev norm.

Definition 11.4 (Lie-Sobolev norm). *For the identified model $\hat{f}(x|\hat{\theta})$, $\hat{g}(x|\hat{\theta})$, $\hat{h}(x|\hat{\theta})$, the squared Lie-Sobolev norm of the model error $(\Delta f, \Delta g, \Delta h) = (f, g, h) - (\hat{f}, \hat{g}, \hat{h})$ is*

$$\begin{aligned}
\|\Delta(f, g, h)\|_W^2 = \int_{\mathcal{X}\times\mathcal{U}} \|\Delta f + \Delta g u\|^2 + \sum_{i=1}^{d_y}\left[\sum_{r=0}^{\rho_i-1} w_i^r \left\|L_f^r h_i - L_{\hat{f}}^r \hat{h}_i\right\|^2\right. \\
\left. + w_i^{\rho_i}\left\|L_f^{\rho_i} h_i + L_g L_f^{\rho_i-1} h_i u - L_{\hat{f}}^{\rho_i}\hat{h}_i - L_{\hat{g}}L_{\hat{f}}^{\rho_i-1}\hat{h}_i u\right\|^2\right] d\mathbb{P}(x, u),
\end{aligned} \tag{11.21}$$

where $\mathbb{P}(x, u)$ is a nonnegative measure on $\mathcal{X} \times \mathcal{U}$ with $\int_{\mathcal{X}\times\mathcal{U}} d\mathbb{P} = 1$, i.e., a probability measure.

Assuming that $\mathcal{U}$ is full-dimensional and that the characteristic matrix

$$A(x) = \begin{bmatrix} L_{g_1}L_f^{\rho_1}h_1(x) & \dots & L_{g_{d_u}}L_f^{\rho_1}h_1(x) \\ \vdots & \ddots & \vdots \\ L_{g_1}L_f^{\rho_{d_y}}h_{d_y}(x) & \cdots & L_{g_m}L_f^{\rho_{d_y}}h_{d_y}(x) \end{bmatrix} \tag{11.22}$$

is nonsingular, $\|(\Delta f, \Delta g, \Delta h)\|_W = 0$ only when $\Delta f(x) = 0$, $\Delta g(x) = 0$, $\Delta h(x) = 0$ almost everywhere in $\mathcal{X}$. The observer-estimator should be designed with an aim to reduce $Q(t)$. Assuming that the input signals and the output signals up to the order of relative degrees are time differentiable, and writing the part of Q after the first term in

(11.20) as S, it can be shown that

$$
\begin{aligned}
\dot{Q} = (\dot{\hat{x}} - \hat{f} - \hat{g}u)^\top \left[\ddot{\hat{x}} - \left(\frac{\partial f}{\partial \hat{x}} + \sum_{j=1}^{n_u} u_j \frac{\partial \hat{g}_j}{\partial \hat{x}} \right) \dot{\hat{x}} + \left(\frac{\partial S}{\partial \hat{x}} \right)^\top - \hat{g}\dot{u} \right] + \frac{\partial S}{\partial \hat{x}} (\hat{f} + \hat{g}u) \\
+ \left[\frac{\partial S}{\partial \hat{\theta}} - (\dot{\hat{x}} - \hat{f} - \hat{g}u)^\top \left(\frac{\partial f}{\partial \hat{\theta}} + \sum_{j=1}^{n_u} u_j \frac{\partial \hat{g}_j}{\partial \hat{\theta}} \right) \right] \dot{\hat{\theta}} + \frac{\partial S}{\partial u} \dot{u} + \sum_{i=1}^{d_y} \sum_{i=0}^{\rho_i} \frac{\partial S}{\partial y_i^{(r)}} y_i^{(r+1)}.
\end{aligned}
\tag{11.23}
$$

Therefore we construct the laws of state observer σ and parameter estimator π as follows:

$$
\begin{aligned}
\sigma &= -\Gamma_\sigma (v - \hat{f} - \hat{g}u) + \left(\frac{\partial f}{\partial \hat{x}} + \sum_{j=1}^{n_u} u_j \frac{\partial \hat{g}_j}{\partial \hat{x}} \right) v - \left(\frac{\partial S}{\partial \hat{x}} \right)^\top + \hat{g}\dot{u} \\
\pi &= -\Gamma_\pi \left[\frac{\partial S}{\partial \hat{\theta}} - (\dot{\hat{x}} - \hat{f} - \hat{g}u)^\top \left(\frac{\partial f}{\partial \hat{\theta}} + \sum_{j=1}^{n_u} u_j \frac{\partial \hat{g}_j}{\partial \hat{\theta}} \right) \right].
\end{aligned}
\tag{11.24}
$$

where $\Gamma_{\hat{x}}$ and $\Gamma_{\hat{\theta}}$ are positive definite matrices of order d_x and d_θ, respectively.

11.3.2 Convergence properties

Substituting the identification law (11.24) into the expression of $\dot{Q}$ (11.23) according to (11.12), we can characterize the convergence property of the observer-estimator (11.24) based on Lyapunov stability analysis. We first consider the case when there exits a "true" value of parameters θ such that the errors in functions f, g, h and Lie derivatives vanish at θ in the absence of observation errors. The following proposition gives the conditions for the nominal convergence of (11.24).

Proposition 11.1 (Nominal convergence of the gradient descent observer-estimator). *Assume that*

- *$\hat{f}, \hat{g}, \hat{h}$, the Lie derivative functions and their partial derivatives with respect to $\hat{x}$ and $\hat{\theta}$ are bounded, strongly convex in $\hat{\theta}$ and Lipschitz in $\hat{x}$;*
- *$u, \dot{u}$ and $\hat{g}(\hat{x})$ are bounded;*
- *The errors in state observations are linearly bounded by the errors in their estimated dynamics and the parameter estimations, i.e., $\|\hat{x} - x\| \leq c_1 \|\dot{\hat{x}} - \hat{f} - \hat{g}u\| + c_2 \|\hat{\theta} - \theta\|$ for some $c_1, c_2 > 0$.*

Under the observer-estimator (11.24), *if the tunable matrices Γ_σ and Γ_π are chosen large enough, then $\dot{Q} \leq 0$ and the equality holds when $\dot{\hat{x}} = \hat{f}(\hat{x}) + \hat{g}(\hat{x})u$ and $\hat{\theta} = \theta$.*

In general, when there exists structural errors, i.e., there is non-vanishing errors between the estimated and true dynamics even when $\hat{\theta} = \theta$ and $\hat{x} = x$, the Lyapunov stability will be replaced with ultimate boundedness, if the structural errors are also bounded. This is stated as the following proposition.

Proposition 11.2 (Boundedness of errors of the gradient descent observer-estimator under structural uncertainties)**.** *Assume that*

- *$\hat{f}, \hat{g}, \hat{h}$, Lie derivatives and their partial derivatives are bounded linearly in the nominal parameter errors, state observation errors plus bounded quantities, i.e., for $\psi = f, g, h, L_f^r h_i, L_g L_f^{\rho_i - 1} h_i$, $r = 0, 1, \dots, \rho_i$, $i = 1, \dots, d_y$, there exists $m_\psi, \ell_\psi, c_\psi > 0$, such that*

$$\|\psi(x) - \hat{\psi}(\hat{x}|\hat{\theta})\| \leq m_\psi \|\theta - \hat{\theta}\| + \ell_\psi \|x - \hat{x}\| + c_\psi; \tag{11.25}$$

- *Q is strongly convex in $\hat{\theta}$ with a bounded deviation, i.e., $\partial S/\partial\hat{\theta} = (\partial Q/\partial\hat{\theta})_0 + \epsilon_Q^\top$, with $(\partial Q/\partial\hat{\theta})_0(\hat{\theta} - \theta) \geq \mu\|\hat{\theta} - \theta\|^2$ for some $\mu > 0$ and $\|\epsilon_Q\| \leq c_Q$ for some $c_Q > 0$;*
- *$u, \dot{u}$ and $\hat{g}(\hat{x})$ are bounded;*
- *The errors in state observations are linearly bounded by the errors in their estimated dynamics and the parameter estimations, i.e., $\|\hat{x} - x\| \leq c_1\|\dot{\hat{x}} - \hat{f} - \hat{g}u\| + c_2\|\hat{\theta} - \theta\| + c_0$ for some $c_1, c_2, c_0 > 0$.*

Under (11.24)*, the errors in state observations and parameter estimations will be ultimately bounded.*

Assuming that the trajectory is informative enough in the sense that it can be rendered close enough to any point in $\mathcal{X} \times \mathcal{U}$, and that the sensitivity of errors to such a distance is limited, the boundedness on the trajectory can be generalized to $\mathcal{X} \times \mathcal{U}$.

Proposition 11.3 (Boundedness of identification error under the gradient descent observer-estimator)**.** *Suppose that the assumptions in Proposition 11.2 hold, and assume that*

- *For some $\epsilon > 0$ and the corresponding $T_\epsilon > 0$, there exists $t_1, \dots, t_{n_k} > T_\epsilon$ and $\eta > 0$ such that for any $(x, u) \in \mathcal{X} \times \mathcal{U}$, $\|(x(t_k) - x, u(t_k) - u\| \leq \eta$;*
- *$L_f^r h_i(x) - L_{\hat{f}}^r \hat{h}_i(x|\hat{\theta})$ and $L_g L_f^{\rho_i - 1} h_i(x) - L_{\hat{g}} L_{\hat{f}}^{\rho_i - 1} \hat{h}_i(x|\hat{\theta})$, $r = 0, 1, \dots, \rho_i$, $i = 1, \dots, d_y$ are uniformally Lipschitz in x for all $\hat{\theta} \in \Theta$.*

Then there is an upper bound $B(\epsilon, \eta) > 0$ such that for any probability measure $\mathbb{P}(x, u)$,

$$\|(\Delta f, \Delta g, \Delta h)\|_W^2 \leq B(\epsilon, \eta). \tag{11.26}$$

The proofs of the 3 propositions in this subsection are provided in Appendix C.2. Although it is desirable to establish the non-Lie-Sobolev performance of the Lie-Sobolev observer-estimator (as (11.7) for the motivating example), this appears to be challenging. Generally for dynamical systems of equal dimensions, even if the relative degrees coincide and the Lie derivatives of the outputs up to the order of relative degrees are equal, the remaining dynamics, known as the *zero dynamics*, can be different. Since the system identification is oriented at control, instead of pursuing the coincidence of zero dynamics, we consider its effect on input–output linearizing control.

11.3.3 Effect on input–output linearizing control

Assuming that the characteristic matrix $A(x)$ is nonsingular, then there exists vector fields $\zeta_k(x)$, $k = 1, \dots, d_\zeta = d_x - \sum_{i=1}^{d_y} \rho_i$ to complement the Lie derivatives, satisfying $L_g \zeta_k = 0$ ($k = 1, \dots, d_\zeta$), such that the nonlinear transformation of coordinates Φ: $\Phi(x) = [\zeta_1(x), \dots, \zeta_{d_\zeta}(x), L_f^0 h_1(x), \dots, L_f^{\rho_{d_y}-1} h_{d_y}(x)]^\top$ is nonsingular [173], and thus an inverse transformation $x = \Phi^{-1}(\zeta, \xi)$ exists. Under the transformed coordinates, the original system (11.9) is expressed as

$$\dot{\zeta} = Z(\zeta, \xi), \;\; \dot{\xi}_i^0 = \xi_i^1, \;\; \dots, \;\; \dot{\xi}_i^{\rho_i - 1} = L_f^{\rho_i} h_i(x) + L_g L_f^{\rho_i - 1} h_i(x) u, \tag{11.27}$$

where $\xi_i^0, \dots, \xi_i^{\rho_i-1}$ correspond to $y_i, \dots, y_i^{\rho_i-1}$, respectively. In the zero dynamics, ξ can be considered as the states to the partial states ζ. For the input–output linearizing control of system (11.27), if the states are accurately known, the inputs are specified as such that the output trajectories are shaped to satisfy

$$\sum_{i=1}^{d_y} \sum_{r=0}^{\rho_i} \beta_{ir} \frac{d^r y_i}{dt^r}(t) = \omega(t) \tag{11.28}$$

for a d_y-dimensional reference signal ω, in which $\beta_{ir} \in \mathbb{R}^{d_y}$ with $[\beta_{1\rho_1}, \dots, \beta_{d_y \rho_{d_y}}]$ being a nonsingular matrix. In other words, the control law is designed as

$$\begin{aligned} u = &\left[\textstyle\sum_{i=1}^{d_y} \beta_{i\rho_i} L_{g_1} L_f^{\rho_i-1} h_i(x) \quad \dots \quad \textstyle\sum_{i=1}^{d_y} \beta_{i\rho_i} L_{g_{d_u}} L_f^{\rho_i-1} h_i(x) \right]^{-1} \\ &\left[\omega - \sum_{i=1}^{d_y} \sum_{r=0}^{\rho_i} \beta_{ir} L_f^r h_i(x) \right] =: B(x)^{-1} b(x). \end{aligned} \tag{11.29}$$

With observation and estimation errors, the terms in the ideal control law above are then replaced with the corresponding observed and estimated terms, which results in an additional term $\Delta\omega = B(B+\Delta B)^{-1}\Delta b - \Delta B(B+\Delta B)^{-1}b$ to the right-hand side of (11.28). If the outputs and Lie derivatives are subject to bounded errors, then $\Delta\omega$ is bounded by their errors, as long as the nonsingularity of $B(x)$ is retained. Usually the reference response should be shaped such that the eigenvalues of the left-hand side of (11.28) should be negative in real parts. Hence the errors in outputs and output derivatives are bounded in terms of $\Delta\omega$ and thereafter in terms of the errors. That is, there exist positive constants c_ψ for $\psi = L_f^r h_i(x)$, $r = 0, \dots, \rho_i$ and $L_g L_f^{\rho_i - 1} h_i(x)$, $i = 1, \dots, d_y$, such that a bound on the deviation of ξ from the reference trajectory (11.28) can be written as $\|\Delta\xi\| \le \sum_\psi c_\psi \|\psi(x) - \hat{\psi}(\hat{x}|\hat{\theta})\|$. Then consider the deviation of the zero dynamics states ζ from its reference trajectory. Since the zero dynamics takes ξ as its input, as long as the incremental zero dynamics

$$\Delta\dot{\zeta} = Z(\zeta + \Delta\zeta, \xi + \Delta\xi) - Z(\zeta, \xi) \tag{11.30}$$

is input–state stable (ISS), then there exists $c'_\psi > 0$ such that

$$\|\Delta\zeta\| \le \sum_\psi c'_\psi \|\psi(x) - \hat{\psi}(\hat{x}|\hat{\theta})\|. \tag{11.31}$$

The above discussion is summarized as follows.

Proposition 11.4 (Performance of input–output linearizing control under the gradient descent observer-estimator)**.** *Suppose that the assumptions in Proposition 11.2 hold, and also assume that*

- *The eigenvalues for the left-hand side of reference trajectory* (11.28) *are all negative in the real part.*
- *The incremental zero dynamics with $\Delta\xi$ as inputs and $\Delta\zeta$ as states is ISS.*

Then the input–output linearizing control based on the observer-estimator given by (11.24) *gives a trajectory with bounded deviations from the reference* (11.28)*.*

11.4 Lie-Sobolev Moving Horizon Estimator

11.4.1 Formulation

Different from the previous explicit observer-estimator, the Lie-Sobolev MHE determines the state observation $\hat{x}(t)$ and parameter estimation $\hat{\theta}(t)$ based on the historical measurements in the past time period of length T, by solving the following dynamic optimization problem about $\hat{\theta}$ and $\hat{x}(s)$ for $s \in [t-T, t]$:

$$\min \int_{t-T}^{t} Q(\hat{x}(s), \hat{\theta} | u(s), y(s)) ds + R(\hat{x}(t-T), \hat{\theta}) \tag{11.32}$$

The objective accounts for the discrepancies between the estimated outputs along with their time derivatives and the measured values. A regulation term $R(\hat{x}(t-T), \hat{\theta})$ accounts for the truncation before the time instant $t-T$ and the allowed range of parameters θ. The constraints are the estimated model equations. By solving the above problem, the obtained $\hat{x}(t)$ and $\dot{\hat{x}}(t)$ are the observed states and observed time derivatives of the states at the current time t, and $\hat{\theta}$ is the current estimated parameters $\hat{\theta}(t)$. Although the practical approximate solution of the problem requires discretization, here we use the original formulation for analysis.

Applying variational calculus, one can easily verify that the optimal solution of the observation-estimation pair $(\{\hat{x}(s) | s \in [t-T, t]\}, \hat{\theta})$ should be specified by the following first-order optimality conditions:

$$\begin{aligned}
0 =& \frac{d}{ds}(\dot{\hat{x}}(s) - \hat{f}(\hat{x}(s)|\hat{\theta}) - \hat{g}(\hat{x}(s)|\hat{\theta})u(s)) - (\dot{\hat{x}}(s) - \hat{f}(\hat{x}(s)|\hat{\theta}) - \hat{g}(\hat{x}(s)|\hat{\theta})u(s))^\top \\
& \left(\frac{\partial \hat{f}}{\partial \hat{x}}(\hat{x}(s)|\hat{\theta}) + \sum_j u_j \frac{\partial \hat{g}_j}{\partial \hat{x}}(\hat{x}(s)|\hat{\theta}) \right), \quad s \in [t-T, t] - \frac{\partial S}{\partial \hat{x}}(\hat{x}(s), \hat{\theta}) \\
0 =& \dot{\hat{x}}(t-T) - \hat{f}(\hat{x}(t-T)|\hat{\theta}) - \hat{g}(\hat{x}(t-T)|\hat{\theta})u(t-T) - \frac{\partial R}{\partial \hat{x}}(\hat{x}(t-T), \hat{\theta}) \\
0 =& \dot{\hat{x}}(t) - \hat{f}(\hat{x}(t)|\hat{\theta}) - \hat{g}(\hat{x}(t)|\hat{\theta})u(t) \\
0 =& \int_{t-T}^{t} \frac{\partial S}{\partial \hat{\theta}}(\hat{x}(s), \hat{\theta}) + (\dot{\hat{x}}(s) - \hat{f}(\hat{x}(s)|\hat{\theta}) - \hat{g}(\hat{x}(s)|\hat{\theta})u(s))^\top \\
& \left(\frac{\partial \hat{f}}{\partial \hat{\theta}}(\hat{x}(s)|\hat{\theta}) + \sum_j u_j \frac{\partial \hat{g}_j}{\partial \hat{\theta}}(\hat{x}(s)|\hat{\theta}) \right) ds + \frac{\partial R}{\partial \hat{\theta}}(\hat{x}(t-T), \hat{\theta}).
\end{aligned} \tag{11.33}$$

As the time t flows for an infinitesimal time δt, there will be infinitesimal changes in the inputs, outputs and output derivatives in the time horizon $[t-T,t]$, which are the parameters needed by the optimization problem (11.32). This will then result in changes $\delta\ddot{\hat{x}}, \delta\dot{\hat{x}}, \delta\hat{x}, \delta\hat{\theta}$ to guarantee that the first-order optimality conditions still hold. The procedure to find the changes in the optimal solution originating from changes in the parameters is known as *sensitivity analysis* [110].

Specifically, under appropriate regularity assumptions, the variation of the first equation in (11.33) can be expressed as

$$\delta\hat{\theta} = \pi\left(\left\{\delta\dot{\hat{x}}(s), \delta\hat{x}(s), \delta u(s), \delta Y(s) | s \in [t-T,t]\right\}\right) \tag{11.34}$$

for some linear functional π, and the variation of the third equation is

$$\delta\dot{\hat{x}}(t) = \sigma\left(\delta\hat{x}(t), \delta\hat{\theta}, \delta u(t), \delta Y(t)\right) \tag{11.35}$$

for some linear functional σ, in which $Y(s)$ is the collection of $y_i^{(r)}(s)$ for $r = 0, 1, \dots, \rho_i$, $i = 1, \dots, d_y$. The variations in the states and their derivatives during the horizon are determined by the variations of the fourth equation of (11.33) as second-order ODEs, and the variations of the second and third equations of (11.33) as two boundary conditions. This results in $\delta\hat{x}(s)$ and $\delta\dot{\hat{x}}(s)$ as linear functionals of $\{\delta u(s), \delta Y(s) | s \in [t-T,t]\}$ depending on the current solution. Therefore we can rewrite the above two formulas as

$$\begin{aligned} \delta\hat{\theta} &= \pi\left(\left\{\dot{\hat{x}}(s), \hat{x}(s), u(s), Y(s), \delta u(s), \delta Y(s) | s \in [t-T,t]\right\}\right), \\ \delta\dot{\hat{x}}(t) &= \sigma\left(\hat{x}(t), \hat{\theta}, , u(s), Y(s), \delta u(t), \delta Y(t)\right), \end{aligned} \tag{11.36}$$

with functionals π and σ linear in $\delta u(s)$ and $\delta Y(s)$. Since $\delta u(s) = \dot{u}(s)\delta t$ and $\delta Y(s) = \dot{Y}(s)\delta t$, we finally have

$$\begin{aligned} \dot{\hat{\theta}} &= \pi\left(\left\{\dot{\hat{x}}(s), \hat{x}(s), u(s), Y(s), \dot{u}(s), \dot{Y}(s) | s \in [t-T,t]\right\}\right), \\ \ddot{\hat{x}}(t) &= \sigma\left(\hat{x}(t), \hat{\theta}, , u(s), Y(s), \dot{u}(t), \dot{Y}(t)\right). \end{aligned} \tag{11.37}$$

This indicates that the MHE is an implicit dynamic system of an observer-estimator.[4]

[4]In many works on MHE (e.g., [208]), the consistency with the identified dynamic model is imposed as a hard constraint, which is the case with $w_i^r \to 0$. Bounds for $\hat{x}$ and $\hat{\theta}$ can also be imposed. Under such formulations, the above analysis of variations and sensitivities can be extended by incorporating the flow

11.4.2 Convergence properties

Due to the implicitness of MHE, the convergence properties need to be considered in a roundabout way. The key idea is to establish the conditions under which the objective function of (11.32) is a Lyapunov function, and the descent or boundedness of such a Lyapunov function implies convergence or boundedness of the distance to nominal parameters and true states. Such properties have been well studied in the literature [341, 176, 280]. Here we rephrase the conditions for the convergence of MHE for state observation given in [280, Theorem 14]. Compared to the original theorem, we simply augment the outputs with their derivatives, augment the states with time-invariant model parameters, and reformulate the conditions in continuous time.

Definition 11.5. *A function α is said to belong to class $\mathcal{K}_\infty$ if it is defined on $[0, +\infty)$, strictly increasing, and such that $\alpha(0) = 0$ and $\alpha(r) \to \infty$ as $r \to \infty$. A function $\beta : [0, +\infty) \times [0, +\infty) \to [0, +\infty]$ is said to belong to class $\mathcal{KL}$ if it is strictly increasing in the first variable with $\beta(0, s) = 0$ for any s, and decreasing in the second variable with $\beta(r, s) \to 0$ as $s \to +\infty$.*

Proposition 11.5 (Convergence properties of Lie-Sobolev MHE)**.** *Assume that*

- *The system represented as the estimation model under nominal parameter values θ with structural errors w, v:*

$$\dot{x} = \hat{f}(x|\theta) + \hat{g}(x|\theta)u + w$$
$$y_i^{(r)} = \begin{cases} L_{\hat{f}}^r \hat{h}_i(x|\theta) + v_i^r, \;\; r = 0, 1, \dots, \rho_i - 1 \\ L_{\hat{f}}^{\rho_i} \hat{h}_i(x|\theta) + L_{\hat{g}} L_{\hat{f}}^{\rho_i} \hat{h}_i(x|\theta)u + v_i^{\rho_i}, \;\; r = \rho_i \end{cases} \tag{11.38}$$

  is incrementally input-to-state stable (ISS) and output-to-state stable (OSS). That is, there exists a $\mathcal{KL}$-class function β and two $\mathcal{K}_\infty$-class functions α_w and α_v, such that if the two systems (11.38) starts at time 0 from two different states $x(0)$ and $x(0)'$, then at any time t, their states $x(t)$ and $x'(t)$ have a distance bounded by

$$\|x(t) - x'(t)\| \leq \beta(\|x(0) - x'(0)\|, t) + \alpha_w(\|w - w'\|_{[0,t]}) + \alpha_v(\|v - v'\|_{[0,t]}). \tag{11.39}$$

- *The function β satisfies $\beta(r, s) \leq c_\beta r^{p_r} s^{-p_s}$ for some $c_\beta > 0$, $p_r \geq 1$, and $p_s > 0$.*

of the Lagrangian multipliers of equality constraints, known as co-states $\lambda(s)$, $s \in [t - T, t]$. Optimality conditions can be derived in a similar appearance to the equations in the Pontryagin maximum principle.

The functions α_w and α_v satisfy $\alpha_w(r) \leq c_w r^{q_w}$ and $\alpha_v(r) \leq c_v r^{q_v}$, respectively, for some $c_w, c_v, q_w, q_v > 0$.

- *The regulation term $R(\hat{x}, \hat{\theta})$ is restricted by*

$$R(\hat{x}, \hat{\theta}) \in \left\| \begin{bmatrix} \hat{x}(t-T) - \hat{x}^*(t-T) \\ \hat{\theta} - \hat{\theta}^* \end{bmatrix} \right\|^q [m_R, M_R], \tag{11.40}$$

where $\left(\{\hat{x}^*(s) | s \in [t-T, t]\}, \hat{\theta}\right)$ is the optimal solution to (11.32)*, $M_R \geq m_R > 0$, and* $\max(1/q_w, 1/q_v, 2p_r/p_s) \leq q \leq 2/\max(q_w, q_v)$.
- *The initial observation-estimation error and structural errors are bounded.*

Then the observation-estimation errors of the MHE (11.32) *remain bounded. Furthermore in the absence of structural error in the model, $\hat{x}(t) - x(t) \to 0$, $\hat{\theta}(t) - \theta \to 0$.*

11.4.3 Effect on nonlinear MPC

The nonlinear MPC of the system (11.9) is such a control strategy where at each time instant t, the control signal $u(t)$ is determined by solving the optimization problem:

$$\begin{aligned} \min \ & \int_t^{t+T} \ell(\tilde{x}(s), \tilde{u}(s)) ds + \ell_{\mathrm{f}}(\tilde{x}(t+T)) \\ \text{s.t. } & \dot{\tilde{x}}(s) = f(\tilde{x}(s)) + g(\tilde{x}(s))\tilde{u}(s), \ \ s \in [t, t+T], \quad \tilde{x}(t) = x(t) \end{aligned} \tag{11.41}$$

and extracting the first piece of the input signal $\tilde{u}(s)$ in the receding horizon $[t, t+T]$ (the horizon length T may be different from the one in MHE). The functions ℓ and ℓ_{f} are called stage cost and terminal cost, respectively. For simplicity we consider the case without process constraints. The stability conditions for nonlinear MPC have been well established in the literature, where the objective function of (11.41), denoted as V, is considered as a control-Lyapunov function. Specifically, if there exists a control policy $u = \kappa(x)$ such that

$$\frac{d\ell_{\mathrm{f}}}{dx}(f(x) + g(x)\kappa(x)) \leq -\ell(x, \kappa(x)) \tag{11.42}$$

then the asymptotic stability follows from the descent property $\dot{V}(t) \leq -\ell(x(t), u(t))$ under appropriate assumptions on the choice of the relevant functions [255].

When the model and the states are not precisely known, $(f(\cdot), g(\cdot))$ in (11.41) should be replaced by $(\hat{f}(\cdot|\hat{\theta}), \hat{g}(\cdot|\hat{\theta})$, and $x(t)$ should be replaced by the observed state $\hat{x}(t)$. The effect of the accuracy of Lie derivatives on the solution of the MPC problem is implicit.

First, the predicted states are related to the predicted outputs, output derivatives and states of the zero dynamics through the nonlinear transformation Φ: $\tilde{x} = \Phi^{-1}(\tilde{\zeta}, \tilde{\xi})$, which transforms the MPC problem into the following form:

$$
\begin{aligned}
\min \ & \int_t^{t+T} \ell(\tilde{\zeta}(s), \tilde{\xi}(s), \tilde{u}(s))ds + \ell_{\mathrm{f}}(\tilde{\zeta}(t+T), \tilde{\xi}(t+T)) \\
\text{s.t. } & \dot{\tilde{\zeta}}(s) = Z(\tilde{\zeta}(s), \tilde{\xi}(s)), \ \ s \in [t, t+T] \\
& \dot{\tilde{\xi}}_i^r(s) = \tilde{\xi}_i^{r+1}(s), \ \ r = 0, \dots, \rho_i - 2, \ \ s \in [t, t+T] \\
& \dot{\tilde{\xi}}_i^{\rho_i - 1}(s) = L_f^{\rho_i} h_i(\tilde{x}(s)) + L_g L_f^{\rho_i - 1} h_i(\tilde{x}(s))\tilde{u}(s), \ \ s \in [t, t+T] \\
& \tilde{x}(t) = x(t)
\end{aligned}
\tag{11.43}
$$

where the Lie derivatives appear in the prediction of the transformed state trajectories. Alternatively, if only the outputs are accounted for in the objective function, one may use Chen-Fliess series expansion to express the outputs [173, Section 3.2]:

$$
y_i(s) = h_i(x(t)) + \sum_{k=0}^{\infty} \sum_{j_0, \dots, j_k = 0}^{d_u} L_{g_{j_0}} \dots L_{g_{j_k}} h_i(x(t)) \int_t^s d\chi_{j_k} \dots d\chi_{j_0}, \tag{11.44}
$$

where $g_0 = f$, $\chi_0(s) = s$, $\chi_j(s) = \int_t^s u_j(s')ds'$ ($j = 1, \dots, d_u$), and $\int_t^s d\chi_{j_k} \dots d\chi_{j_0} = \int_t^s d\chi_{j_k}(s') \int_t^{s'} d\chi_{j_{k-1}} \dots d\chi_{j_0}$. Again in (11.44) we see the presence of Lie derivatives.

We formalize the impact of the observation-estimation errors on the MPC performance by considering its perturbation on the stability conditions (11.42) and hence on the Lyapunov descent. The coordinate-transformed MPC formulation is used, with the dynamics abbreviated as $(\dot{\zeta}, \dot{\xi}) = \Lambda_0(\zeta, \xi) + \Lambda(\zeta, \xi)u$. Suppose that there exists a κ such that

$$
\left(\frac{d\ell_{\mathrm{f}}}{d\zeta}, \frac{d\ell_{\mathrm{f}}}{d\xi} \right) (\Lambda_0(\zeta, \xi) + \Lambda(\zeta, \xi)\kappa(\zeta, \xi)) \leq -\ell(\zeta, \xi, \kappa(\zeta, \xi)). \tag{11.45}
$$

When the information of Λ_0 and Λ is erroneous, the right-hand side above needs to be augmented with a term to bound such errors. We may assume that structural errors are linearly bounded by $\|\hat{\theta} - \theta\|$ plus a positive constant. Then the change in the Lyapunov function becomes

$$
\dot{V}(t) \leq -\ell(\hat{x}(t), u(t)) + c_\theta \|\hat{\theta} - \theta\| + c_0 \tag{11.46}
$$

for some $c_\theta, c_0 > 0$. Then, as long as the $\hat{x}(t)$ entry in the ℓ term above can be replaced by $x(t)$ with an additional term linear in the observation error $\|x(t) - \hat{x}(t)\|$, and the observation-estimation errors are ultimately bounded, it follows that

$$\dot{V}(t) \leq -\ell(x(t), u(t)) + \epsilon \tag{11.47}$$

for some $\epsilon > 0$, which implies the ultimate boundedness of the states in the closed-loop system [255]. The conditions are summarized in the following proposition.

Proposition 11.6 (Performance of MPC under the Lie-Sobolev MHE)**.** *Assume that*

- *the state observation and parameter estimation errors* $(x - \hat{x}, \theta - \hat{\theta})$ *are ultimately bounded, e.g., under the conditions of Proposition 11.5;*
- *the stage cost* $|\ell(x) - \ell(\hat{x})| \leq c_x \|x - \hat{x}\|$ *for some* $c_x > 0$*;*
- *the structural errors in the transformed dynamics* (11.27) *are bounded by* $c_\theta \|\hat{\theta} - \theta\| + c_0$ *for some* $c_\theta, c_0 > 0$*;*
- *there exist* $\mathcal{K}_\infty$ *functions* α *and* α_f *such that* $\ell_\mathrm{f}(x) \leq \alpha_\mathrm{f}(x)$, $\ell(x, u) \geq \alpha(x)$.

Then $\|x\|$ *is ultimately bounded.*

11.5 Applications

We apply the proposed Lie-Sobolev system identification approaches to several examples. First, we use the explicit observer-estimator proposed in Section 11.3 on two numerical examples, one with nominal parameter values that perfectly matches the true dynamics, and the other with structural error in the estimated model. Then we apply the Lie-Sobolev MHE in Section 11.4 to a complex chemical reactor system. Through these case studies, improved convergence of the estimated parameters, smaller structural errors, and better performance of the resulting nonlinear control can be seen in Lie-Sobolev approaches compared to the non-Lie-Sobolev ones.

11.5.1 Example without structural errors

Consider the following system [173, Example 4.1.5]:

$$\dot{x} = \begin{bmatrix} \theta_1 x_1 x_2 - x_1^3 \\ x_1 \\ -x_3 \\ \theta_3 x_1^2 + x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \theta_2(1 + x_3) \\ 1 \\ 0 \end{bmatrix} u, \quad y = x_4 \tag{11.48}$$

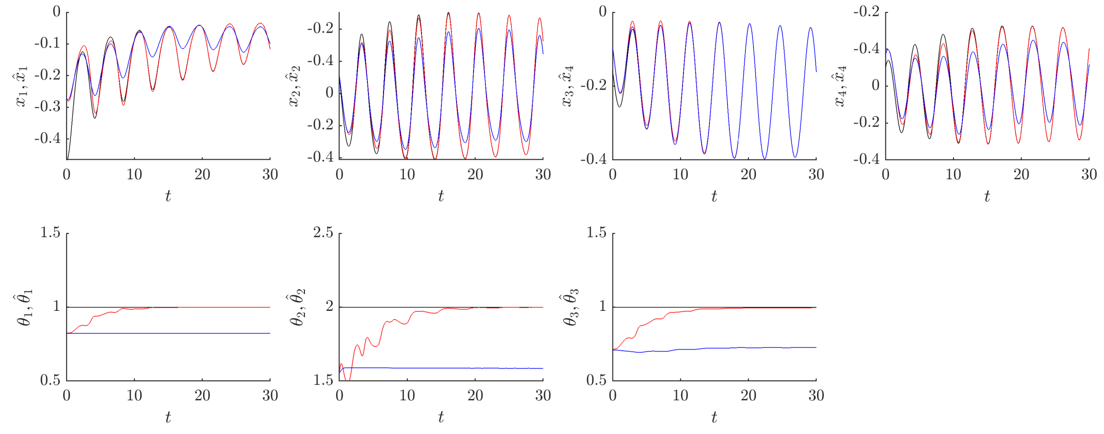


Figure 11.2: Observed states and estimated parameters for (11.48) under Lie-Sobolev (red) and non-Lie-Sobolev (blue) approaches compared to the true values (black).

with true parameter values $\theta_1 = 1$, $\theta_2 = 2$ and $\theta_3 = 1$. By choosing the control input as a proportional feedback from the output, the resulting state trajectories appear to be oscillatory around the origin for a considerable time span, and hence this is considered as a suitable condition to perform identification.

For the Lie-Sobolev approach, the weight constants w^0, w^1, w^2 are all set as 1. For the non-Lie-Sobolev one, w^1 and w^2 become 0. The tunable semidefinite matrices in the observer-estimator are empirically determined as $\Gamma_\sigma = \text{diag}(5, 2, 2, 1)$, $\Gamma_\pi = \text{diag}(1, 5, 1.5)$, under which the observed states and estimated parameters of the Lie-Sobolev approach converge to the true values.[5] The trajectories of $(\hat{x}, \hat{\theta})$ during a simulation time span of $T = 30$ are shown in Fig. 11.2. For non-Lie-Sobolev observation and estimation, the lack of convergence to the nominal values does not appear to be improved by choosing different tunings of Γ_σ and Γ_π. The reason for this limitation of the non-Lie-Sobolev approach is that when the output derivatives are not explicitly considered, the objective function can be insensitive to some of the parameters. After a short incipient time, the trajectories of the observed states become consistent with the estimated parameters that deviate from the true values. Afterwards, the updates on the estimated parameters are driven only by the difference between the output y and the observed $\hat{x}_4$, which results in changes significant only in $\hat{\theta}_3$. Therefore, we conclude that

[5]The initial state for simulation $x(0)$ is chosen according to a uniform distribution in the hypercube $[-0.5, 0.5]^4$. The initialized observation error for x, $\hat{x}(0) - x(0)$ is randomized in $[-0.25, 0.25]^4$, and for $\dot{x}$, $\dot{\hat{x}}(0) - \dot{x}(0)$ is randomized in $[-0.25^2, 0.25^2] \times [-0.25, 0.25]^3$. The initial guess for the parameters, $\hat{\theta}(0)$, is randomly chosen in $[0.5, 1.5] \times [1.5, 2.5] \times [0.5, 1.5]$.

an advantage of Lie-Sobolev identification is its improved sensitivity in parameters.

11.5.2 Example with structural errors

Consider the following system [173, Example 4.1.4]:

$$\dot{x} = \begin{bmatrix} -x_1 \\ x_1 x_2 \\ x_2 \end{bmatrix} + \begin{bmatrix} \phi(x_2) \\ 1 \\ 0 \end{bmatrix} u, \;\; y = x_3 \tag{11.49}$$

where $\phi(x_2) = \exp(x_2)$ unknown a priori and parametrized as $\hat{\phi}(x_2|\theta) = \theta_0 + \theta_1 x_2 + \frac{1}{2}\theta_2 x_2^2$. A nominal estimation according to the Maclaurin series would be $\theta_0 = 1$, $\theta_1 = 1$, $\theta_2 = 1$. The tuning of the observer-estimator (11.24) is determined as $\Gamma_\sigma = I$, and $\Gamma_\pi = \mathrm{diag}(4, 1, 10)$. The resulting trajectories[6] within $T = 20$ are shown in Fig. 11.3, where one can observe that the Lie-Sobolev approach results in apparently smaller state observation errors, while under the non-Lie-Sobolev identification, the state observation error does not appear to vanish. To quantify the performance of parameter identification, we may calculate the integrated errors on the identified function ϕ: $\int_0^T (\phi(x(t)) - \hat{\phi}(\hat{x}(t)|\hat{\theta}(t)))^2 dt$, which equals 0.7131 for Lie-Sobolev identification and 7.0140 for non-Lie-Sobolev identification. This result confirms the advantage of using Lie-Sobolev identification in the presence of structural errors.

A non-trivial problem involved in the identification is how to numerically find the derivatives of y_i up to order ρ_i at any time instant $t > 0$. Generally, numerical derivatives will contain noise inevitably, and typically higher-order derivatives will be noisier than lower-order ones. If the assumptions in the propositions presented earlier still hold, then the theoretical convergence properties can be established. Here we use a sliding-mode differentiator proposed by [220]:

$$\begin{aligned} \dot{z}_i^0 &= -\lambda_i^0 \gamma_i^{\frac{1}{\rho_i+1}} |z_i^0 - y_i|^{\rho_i/(\rho_i+1)} \mathrm{sign}(z_i^0 - y_i) + z_i^1 \\ \dot{z}_i^r &= -\lambda_i^r \gamma_i^{\frac{1}{\rho_i-r+1}} |z_i^r - \dot{z}_i^{r-1}|^{\frac{\rho_i-r}{\rho_i-r+1}} \mathrm{sign}(z_i^r - \dot{z}_i^{r-1}) + z_i^{r+1}, \;\; 1 \le r \le \rho_i - 1 \\ \dot{z}_i^{\rho_i} &= -\lambda_i^{\rho_i} \gamma_i \mathrm{sign}(z_i^{\rho_i} - \dot{z}_i^{\rho_i-1}) \end{aligned} \tag{11.50}$$

[6]The initial guess of parameters are chosen under a uniform distribution in $[0.5, 1.5]^3$. The initial state is chosen randomly in $[-0.5, 0.5]^3$, and the initial guess of states and state derivatives are perturbed from their true values respectively by $[-0.5, 0.5]^3$.

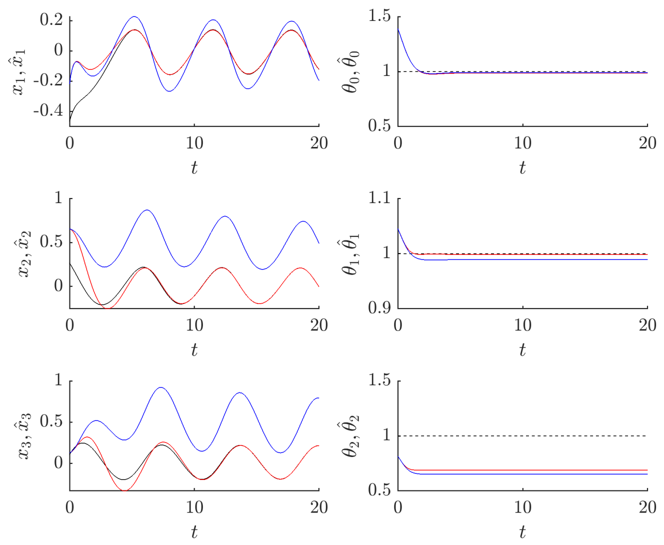


Figure 11.3: Observed states and estimated parameters for (11.49) under Lie-Sobolev (red) and ordinary (blue) identification, respectively. The solid curves and dashed lines in black are true states and nominal parameter values, respectively.

where the parameters are recommended as $\lambda_i^0 = 12$, $\lambda_i^1 = 8$, $\lambda_i^2 = 5$, with γ_i being a tunable Lipschitz estimate of the y_i signal. z_i^r is thus an estimate of $y_i^{(r)}$. For faster ODE solution, we approximate $\mathrm{sign}(\cdot)$ with $\tanh(\cdot/0.01)$.

Assuming an initial deviation of z from $(y, \dot{y}, \ddot{y})$ randomized according to a uniform distribution in $[0.5, 0.5]^3$, under the Levant sliding mode differentiator ($\gamma = 1$), the trajectories of the observed states and estimated parameters are shown in Fig. 11.4. It is observed that the trajectories under the sliding mode differentiator are asymptotically close to the trajectories with exact output derivatives. The integrated errors for the numerical differentiation-based identification is 0.8993, 26.1% larger than the exact case. It is thus concluded that due to the capability of sliding mode differentiator to give accurate estimations of output derivatives, the performance of numerical differentiation-based Lie-Sobolev identification remains satisfactory.

11.5.3 A glycerol etherification reactor

We consider a glycerol etherification reactor [233] with 6 states, 1 input and 1 output, and a relative degree of 1. The true dynamics involving the thermodynamics of non-ideal mixtures is approximated by a 6-parameter ideal mixture model. It is desirable

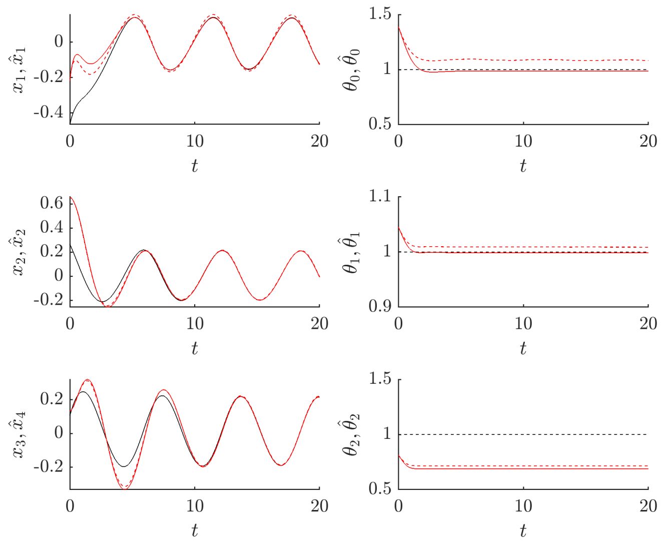


Figure 11.4: Observed states and estimated parameters for (11.49) under the Lie-Sobolev approach with exact output derivatives (red solid) and sliding mode numerical differentiation (red dash).

to handle this structural deviation with Lie-Sobolev identification, in which the time derivative of the output accounts for the sensitivity of reaction rates on component concentrations. A detailed description of the system is given in Appendix C.3.

For the identification, a sinusoidal excitation is imposed on the input F_1 with an amplitude of 50 kmol/h and a period of 1 hour. The output derivatives are obtained by Levant's sliding mode differentiator (11.50). The horizon length for MHE is set as 1.5 hour. The trajectories of observed states, estimated parameters, and the correspondingly inferred outputs and output derivatives of the Lie-Sobolev and non-Lie-Sobolev MHEs[7] are compared to the actual states and nominal parameters in Fig. 11.5. It is observed that due to the structural error in the parametric model assuming ideal liquid mixture, the observed states and estimated parameters inevitably have deviations from the true values of states and nominal kinetic parameters. By using the Lie-Sobolev MHE, the observation and estimation result in significantly smaller deviations in the

[7]The MHE is discretized by 30 finite elements. The weights for output and output derivative are 10^2 and 1, respectively. The MHEs are coded using the `pyomo.dae` module [297] in Python 3.6 in Anaconda 3 with IPOPT 3.11.1 as the solver. The computational performance is improved by providing a lower bound $(-5, -5, 1, 1, 5, 1)$ and an upper bound $(-0.5, -1, 10, 10, 15, 5)$ on the parameters. The observed states are bounded within a proportion of $(0.1, 0.025, 0.1, 0.1, 0.1, 0.1)$ of their respective steady-state values, and the quadratic inverse of these bounds are used as the weights of the states.

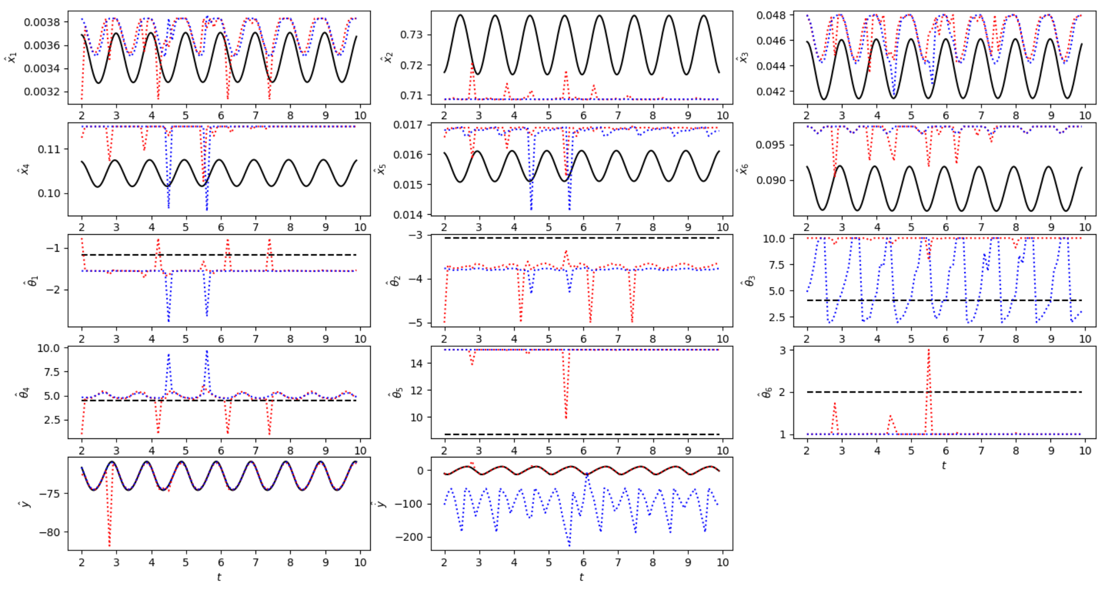


Figure 11.5: Observed states and estimated parameters for the reactor under Lie-Sobolev (red dotted) and ordinary (blue dotted) MHE, respectively. The solid curves and dashed lines in black are true states and nominal parameter values, respectively.

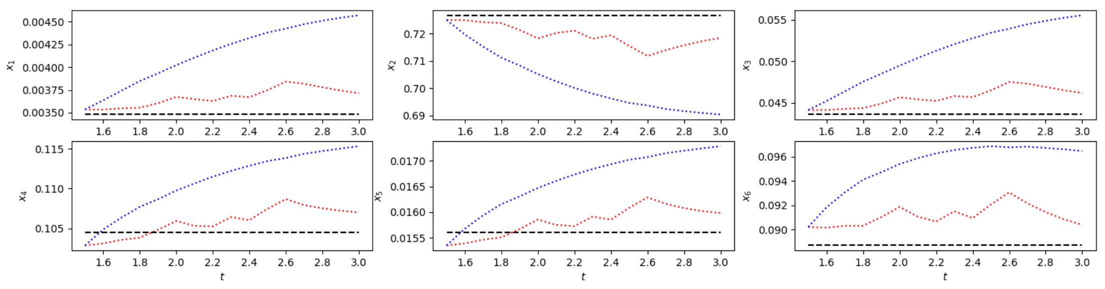


Figure 11.6: Closed-loop trajectories of MHE-MPC based on Lie-Sobolev (red) and non-Lie-Sobolev identification (blue) compared to the steady state (black).

output derivative. Comparing the trajectories of the Lie-Sobolev MHE to those of the non-Lie-Sobolev MHE, we note that the different decisions made by the two identification schemes include primarily the estimation of θ_3 and secondarily the estimation of θ_2, namely the pre-exponential factors of the two main reactions occurring in the system – the reactions of IB with DE (with a molar fraction of about 0.1045) and with ME (with a molar fraction of approximately 0.0436).

To examine the impact of Lie-Sobolev MHE on nonlinear control, MPC simulations are then performed based on the identified model starting from initial points randomly sampled around the steady state within the state bounds of the MHEs. The MPC is activated after one MHE horizon is passed, before which the control signal is fixed at zero. The prediction horizon length and the discretization scheme is the same as those of MHE, and the sampling time is 0.1 hours, during which the control input is held constant. Fig. 11.6 shows the closed-loop trajectories of inputs and states using MPC based on MHE as the state observer and the models determined by the Lie-Sobolev and non-Lie-Sobolev identification. Clearly, the model predictive controller using the Lie-Sobolev estimated model parameters and Lie-Sobolev MHE better stabilizes the process near the steady state in the presence of structural errors, while the non-Lie-Sobolev controller steers the molar fractions away from the steady state. In other words, nonlinear system identification using the Lie-Sobolev MHE results in improved MPC control performance.

It should be noted, however, that the incorporation of Lie derivative terms significantly increases the computational difficulty of solving MHE. The total CPU time in the above MPC simulation is 372.3 and 116.4 seconds with Lie-Sobolev and non-Lie-Sobolev state observation, respectively. Solver failures are also more frequently encountered.

11.6 Conclusions

In this chapter we have proposed that for nonlinear control, it is desirable to perform system identification following a Lie-Sobolev procedure, where the state observation and parameter estimation explicitly account for Lie derivatives. After showing a simple motivating example and the improved performance of Luenberger observer for linear systems, we have discussed the Lie-Sobolev formulations and their convergence properties for explicit gradient descent-based and implicit moving horizon-based identification schemes. Their effects on input–output linearizing control and MPC are also discussed, respectively. The improved performance of Lie-Sobolev identification is demonstrated by two numerical examples and a case study on a glycerol etherification reactor.

The scope of this chapter is restricted to the identification itself and the control performance is considered separately. The combination of Lie-Sobolev identification with robust and adaptive control will be investigated in our future studies.

Chapter 12
Distributed Adaptive Dynamic Programming for Data-Driven Optimal Control

12.1 Introduction

The analytics of big data or the doctrine of data science has become a prevalent research focus due to its potential of bringing a transformation to various industries [257, 336, 466]. Data-driven methods have been used for many different purposes, including process control and monitoring [170, 213, 215, 433]. For modern chemical and energy plants which are large-scale and complex [18], while model-based control is being extensively pursued, it remains difficult to generate an accurate control-oriented model based on first principles or system identification. Data-driven control methods are classified into offline, online, and combined offline-online approaches [164]. Online approaches [428, 62] are advantageous in terms of improving control performance over time. However, the stability of online designed controllers is difficult to predict and requires restrictive assumptions. This shortcoming is overcome by offline approaches [51, 222, 301]. Combined offline-online approaches [403] have been developed to incorporate both closed-loop stability and performance improvement with online data. This work focuses on the idea of adaptive dynamic programming (ADP), which can be either an offline or an online approach to solve for the optimal control policy [445], and has significant potential to overcome the curse of dimensionality prevalent in data-driven techniques [330].

It is well known that the solution to the classical dynamic programming problem is given by the Hamilton-Jacobi-Bellman (HJB) equations [28], which are partial differential equations unsolvable analytically for general nonlinear systems. The ADP approach of approximating the optimal control refers to an iterative algorithm (involving policy iteration or value iteration) of solving the HJB equations [360, 27]. Such an approach has been used for model-based optimal control (e.g., [481, 482]). In the context of data-driven optimal control without model information, neural networks or suitable basis functions are used to approximate the optimal control policy and cost [446, 33], with

©2018 Elsevier. Reprinted, with permission, from W. Tang, P. Daoutidis. Distributed adaptive dynamic programming for data-driven optimal control. *Systems & Control Letters* (Volume 120, October 2018). `DOI:10.1016/j.sysconle.2018.08.002`.

the parameters optimized (trained) by the operational data under appropriate algorithms such as reinforcement learning [401]. This idea has been successfully practiced in finite-state models, i.e., Markov decision processes [335], and recently explored for continuous state-space models [480, 178, 279, 231].

We note that in the existing works on ADP-based data-driven optimal control, the computation is performed in a centralized manner, where all data are processed by a single computing agent. On the other hand, distributed computing [32, 218] allows the computation to be performed by multiple agents coordinated through communication. Due to their parallelizability and applicability to large datasets collected and stored in a distributed manner, distributed algorithms have been recognized as being of central importance in statistics and machine learning for the big-data era [81, 235]. Studies have extended distributed optimization algorithms, including the alternating direction method of multipliers (ADMM) [44], from convex to nonconvex problems [162], and have accelerated the convergence from a linear to a quadratic rate [185, 458]. The recent 2017 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL) identified the application of distributed algorithms to data-driven optimal control as an open problem that should be pursued.

In this work, we consider input-affine nonlinear control systems, and point out that the *offline* data-based control with ADP formulations involves an optimization problem where one minimizes a regression objective function expressed as a sum of data-based terms. This type of problems, with applications in the fields of signal processing and wireless communications, has been reformulated as a multi-agent optimization problem (consensus optimization problem to be more specific) and handled with distributed optimization algorithms [44, Chapter 7]. Inspired by these results, we solve the data-driven optimal control problem with such distributed optimization algorithms. We call the resulting framework *distributed adaptive dynamic programming* (dADP) as it adaptively updates the regression parameters to approach the optimal control throughout the distributed optimization iterations. The remainder of this chapter is organized as follows. In Section 12.2, preliminaries on HJB equations and their data-driven version is given. Then in Section 12.3 the multi-agent optimization problem for the regression of offline data-driven HJB equations is formulated, and the ADMM algorithm and its accelerated variants for its solution is adopted. The input-constrained optimal control problem is further considered in Section 12.4. Conclusions are provided in Section 12.5.

12.2 HJB Equations and Data-Driven Solution

We consider a nonlinear input-affine system:

$$\dot{x} = f(x) + g(x)u \tag{12.1}$$

in which $u(t) \in \mathbb{R}^m$ is the vector of manipulated inputs, $x(t) \in \mathbb{R}^n$ is the vector of states, and f and g are continuous functions. We assume that the origin is an equilibrium of the system ($f(0) = 0$) and we consider the state regulation problem. For state feedback control $u(t) := u(x(t))$, the infinite-horizon control cost to drive the system to the origin from the initial position x_0, $V(x_0)$, is defined as

$$V(x_0) = \min_{u(t), t\in[0,\infty)} \int_0^\infty C(x(t), u(t))dt, \tag{12.2}$$

where $C(x, u)$ is a continuously differentiable utility or cost function. According to Bellman's principle of optimality, the optimal control $u^*(x)$ and the corresponding minimized control cost $V^*(x)$ should satisfy the following conditions, called Hamilton-Jacobi-Bellman (HJB) equations:

$$\begin{aligned} u^*(x) &= \arg\min_u \left[(\nabla V^*(x))^\top (f(x) + g(x)u) + C(x, u)\right] \\ 0 &= (\nabla V^*(x))^\top (f(x) + g(x)u^*(x)) + C(x, u^*(x)) \end{aligned} \tag{12.3}$$

The first equation can be written as

$$\left(\frac{\partial C}{\partial u}\right)_{(x,u^*(x))} + (\nabla V^*(x))^\top g(x) = 0. \tag{12.4}$$

Now we seek an equation that correlates $u^*(x)$ and $V^*(x)$ without system model information (f, g) but with process data. Assume that we have data of the inputs and the states $(u(t), x(t))$ during a time interval $[t', t'']$. Then

$$\begin{aligned} &\frac{dV^*(x(t))}{dt} = (\nabla V^*(x(t)))^\top (f(x(t)) + g(x(t))u(t)) \\ &= (\nabla V^*(x(t)))^\top (f(x(t)) + g(x(t))u^*(x(t))) + (\nabla V^*(x(t)))^\top g(x(t))(u(t) - u^*(x(t))) \\ &= -C(x(t), u^*(x(t))) - \left(\frac{\partial C}{\partial u}\right)_{(x(t),u^*(x(t)))} (u(t) - u^*(x(t))) \end{aligned} \tag{12.5}$$

and by integration on $[t', t'']$, we have

$$V^*(x(t')) - V^*(x(t'')) = \int_{t'}^{t''} C(x(t), u^*(x(t))) + \left(\frac{\partial C}{\partial u}\right)_{(x(t),u^*(x(t)))} (u(t) - u^*(x(t)))dt. \tag{12.6}$$

The left-hand side is the decrease in the cost-to-go as the system states move from $x(t')$ to $x(t'')$. This decrease results by the two terms on the right-hand side, namely the accumulated cost during $[t', t'']$ under the optimal control, and the cost deviation from the optimal control $u^*(x(t))$ to the executed control $u(t)$.

We now provide the following theorem which links the solution to (12.6) with the solution to (12.3), i.e., the optimal control policy and cost. A similar conclusion was established [241] for the intermediate functions obtained in reinforcement learning, instead of the optimal functions $u^*(x)$ and $V^*(x)$.

Theorem 12.1. *Let $C(x, u)$ be convex in u. Assume that all the stabilizing control policies form a convex set. If any pair of $u^*(x)$ and $V^*(x)$ satisfies* (12.6) *for all trajectories $(u(t), x(t))$, then $u^*(x)$ and $V^*(x)$ also satisfy the HJB equations* (12.3)*.*

Proof. Since the solution to (12.3) satisfies (12.6), we only need to prove that the solution to (12.6) is unique up to a constant shift in $V^*(x)$. Assume that there exist two solutions to (12.6), $(u_i^*(x), V_i^*(x))$, $i = 1, 2$. Let $W(x) := V_1^*(x) - V_2^*(x)$. Then

$$\begin{aligned}\frac{dW}{dt} =& - C(x(t), u_1^*(x(t))) + \left(\frac{\partial C}{\partial u}\right)_{(x(t),u_1^*(x(t)))} (u_1^*(x(t)) - u(t)) \\ &+ C(x(t), u_2^*(x(t))) - \left(\frac{\partial C}{\partial u}\right)_{(x(t),u_2^*(x(t)))} (u_2^*(x(t)) - u(t)).\end{aligned} \tag{12.7}$$

When $C(x, u)$ is convex in u, there exists a $\gamma(x) \in [0, 1]$, such that a control signal

$$u^*(x(t)) = \gamma(x(t))u_1^*(x(t)) + (1 - \gamma(x(t)))u_2^*(x(t)) \tag{12.8}$$

can be constructed with $dW(x(t))/dt = 0$. Specifically, if $\left(\frac{\partial C}{\partial u}\right)_{(x,u_1^*(x))} = \left(\frac{\partial C}{\partial u}\right)_{(x,u_2^*(x))}$, any $\gamma(x) \in [0, 1]$ makes $dW/dt = 0$, otherwise

$$\gamma(x) = \frac{C(x, u_2^*(x)) - C(x, u_1^*(x)) - \left(\frac{\partial C}{\partial u}\right)_{(x,u_1^*(x))} (u_2^*(x) - u_1^*(x))}{\left[\left(\frac{\partial C}{\partial u}\right)_{(x,u_1^*(x))} - \left(\frac{\partial C}{\partial u}\right)_{(x,u_2^*(x))}\right] (u_1^*(x) - u_2^*(x))}. \tag{12.9}$$

Along the trajectories driven by $u^*(x)$, $W(x)$ is a constant. Since the stabilizing control policies form a convex set, the control policy (12.8) is stabilizing. Therefore, starting from any x, under $u^*(x)$, the trajectory approaches the origin with $dW/dt = 0$, which implies that $W(x) = W(0)$, i.e., $V_1^*(x) - V_2^*(x) \equiv W(0)$. $\square$

Different from (12.3), (12.6) requires data instead of the system model f, g. Therefore the optimal control problem is naturally converted to the problem of finding $(u^*(x), V^*(x))$ which best satisfies (12.6) based on the collected data. For simplicity we assume $C(x,u) = Q(x) + u^\top Ru$, where $Q(x)$ is a positive definite function, and R is a positive definite matrix. Then (12.6) turns into

$$\int_{t'}^{t''} (u^*(x(t)))^\top R(u^*(x(t)) - 2u(t))dt + V^*(x(t')) - V^*(x(t'')) - \int_{t'}^{t''} Q(x(t))dt = 0. \tag{12.10}$$

Now we approximate the functions $V^*(x)$ and $u_l^*(x), l = 1, 2, \ldots, m$ through a finite set of basis functions, $\psi_k(x)$, $k = 1, 2, \ldots, K_0$ and $\phi_{lk}(x)$, $k = 1, 2, \ldots, K_l$, $l = 1, 2, \ldots, m$:

$$V^*(x) \approx \sum_{k=1}^{K_0} \eta_k \psi_k(x), \quad u_l^*(x) \approx \sum_{k=1}^{K_l} \theta_{lk}\phi_{lk}(x). \tag{12.11}$$

where η_k is the coefficient of the basis function $\psi_k(x)$ in the approximate expression of $V^*(x)$ and θ_{lk} is the coefficient of the basis function $\phi_{lk}(x)$ in the approximate expression of $u_l^*(x)$, $k = 1, 2, \ldots, K_l$, $l = 1, 2, \ldots, m$. The basis functions are usually chosen as polynomials, given that the $u^*(x)$ and $V^*(x)$ are linear and quadratic functions, respectively, for linear systems (which approximate nonlinear systems near the origin), and that $u^*(x)$ and $V^*(x)$, as continuous functions, can be uniformly approximated by polynomials. With a proper choice of basis functions, the approximation error can become sufficiently small.

Substituting (12.11) into (12.10), one should have the coefficients η_k and θ_{lk}, $k = 1, 2, \ldots, K_l$, $l = 1, 2, \ldots, m$ satisfy

$$\begin{aligned} 0 \approx &\sum_{l=1}^{m}\sum_{l'=1}^{m}\sum_{k=1}^{K_l}\sum_{k'=1}^{K_{l'}} r_{ll'}\theta_{lk}\theta_{l'k'} \int_{t'}^{t''} \phi_{lk}(x(t))\phi_{l'k'}(x(t))dt - 2\sum_{l=1}^{m}\sum_{l'=1}^{m}\sum_{k=1}^{K_l} \\ &r_{ll'}\theta_{lk}\int_{t'}^{t''} \phi_{lk}(x(t))u_{l'}(t)dt + \sum_{k=1}^{K_0}\eta_k\left[\psi_k(x(t')) - \psi_k(x(t''))\right] - \int_{t'}^{t''} Q(x(t))dt. \end{aligned} \tag{12.12}$$

This can be written in a compact vector-matrix form as $\alpha^\top\eta + \theta^\top\Xi\theta - 2\beta^\top\theta - 1 \approx 0$, where $\alpha \in \mathbb{R}^{K_0}$, $\beta \in \mathbb{R}^K$ and $\Xi \in \mathbb{R}^{K\times K}$ ($K = \sum_{l=1}^m K_l$) with components

$$\begin{aligned} \alpha_k &= \frac{\psi_k(x(t')) - \psi_k(x(t''))}{\int_{t'}^{t''} Q(x(t))dt}, \quad \beta_{lk} = \sum_{l'=1}^{m} r_{ll'} \frac{\int_{t'}^{t''} \phi_{lk}(x(t))u_{l'}(t)dt}{\int_{t'}^{t''} Q(x(t))dt}, \\ \xi_{lk,l'k'} &= r_{ll'} \frac{\int_{t'}^{t''} \phi_{lk}(x(t))\phi_{l'k'}(x(t))dt}{\int_{t'}^{t''} Q(x(t))dt}. \end{aligned} \tag{12.13}$$

The values of α, β and Ξ are specified by the trajectory sample $(x(\cdot), u(\cdot))$. For different samples $i = 1, \ldots, N$, the corresponding parameters α_i, β_i, Ξ_i can be calculated, and the optimal values of θ and η should make all squared regression residuals

$$F_i(\theta, \eta) = (\alpha_i^\top \eta + \theta^\top \Xi_i \theta - 2\beta_i^\top \theta - 1)^2, \ \ i = 1, 2, \ldots, N \tag{12.14}$$

close to 0. Therefore, given data samples $i = 1, \ldots, N$, we solve the following regression problem for the parameters θ, η:

$$\min_{\theta,\eta} \ \frac{1}{N} \sum_{i=1}^{N} F_i(\theta, \eta). \tag{12.15}$$

Note that the objective function in the optimization problem (12.15) is a summation of data-based terms. Motivated by this, we proceed to cast the problem as a consensus optimization problem to be solved with distributed optimization algorithms.

12.3 Distributed Optimization Formulation

12.3.1 Multi-agent optimization

Now we consider solving the optimization problem (12.15) through distributed algorithms. Denoting $z = [\theta^\top \ \ \eta^\top]^\top$, and

$$A_i = \begin{bmatrix} 2\Xi_i & 0 \\ 0 & 0 \end{bmatrix}, \ \ b_i = \begin{bmatrix} -2\beta_i \\ \alpha_i \end{bmatrix},$$

the problem (12.15) is expressed as

$$\min_z \ F(z) = \frac{1}{N} \sum_{i=1}^{N} F_i(z) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2} z^\top A_i z + b_i^\top z - 1 \right)^2. \tag{12.16}$$

For the solution of (12.16), we adopt the alternating direction method of multipliers (ADMM) [44], which is a type of primal-dual algorithms for multi-agent optimization. Specifically, here we consider a transformation of the problem (12.16) to:

$$\min_{z_0, z_1, \dots, z_N} \sum_{i=1}^{N} F_i(z_i) \quad \text{s.t. } z_0 = z_i, \;\; i = 1, \dots, N. \tag{12.17}$$

The augmented Lagrangian is defined as

$$\mathcal{L} = \sum_{i=1}^{N} \left[F_i(z_i) + \lambda_i^\top (z_i - z_0) + \frac{\mu}{2} \|z_i - z_0\|^2 \right]. \tag{12.18}$$

in which λ_i is the dual variable corresponding to the consensus constraint $z_i = z_0$ and μ is the penalty parameter. The optimization is performed in iterations of agent updates (parallelizable), consensus update, and dual update. Specifically, in each iteration, the following procedures are executed:

$$\begin{aligned} z_i &:= \arg\min_{\hat{z}_i} \mathcal{L}(z_0, z_1, \dots, \hat{z}_i, \dots, z_N), \;\; i = 1, \dots, N \\ z_0 &:= \arg\min_{\hat{z}_0} \mathcal{L}(\hat{z}_0, z_1, z_2, \dots, z_N) = \frac{1}{N} \sum_{i=1}^{N} \left(z_i + \frac{\lambda_i}{\mu} \right) \\ \lambda_i &:= \lambda_i + \mu(z_i - z_0), \;\; i = 1, \dots, N. \end{aligned} \tag{12.19}$$

The consensus of z_0 with $z_1, \dots, z_N$ is guaranteed to form by the dual update and the quadratic penalty term. The ADMM algorithm is known to have a linear convergence rate under certain conditions (see, e.g., [148, 229]). When the functions F_i are non-convex, [162] showed that the ADMM linearly converges to a stationary point under certain conditions. We propose to apply the ADMM method to (12.17) for the solution of (12.15). We call this method of solving the approximate optimal control problem as "*distributed adaptive dynamic programming*" (dADP), in the sense that the parameters for the basis functions, θ, η, in (12.11) are updated adaptively in a distributed optimization algorithm based on data.

Remark 12.1. *For the problems in the form of* (12.16)*, a pristine "push-sum" algorithm was initially proposed in [287], where each agent i seeks to optimize its local objective $F_i(z)$, and the obtained updates are communicated with its neighbors on a communication graph. The underlying mechanism is that all the agents, with the same* *step size for the gradient descent line search, update their optimized variables in such a coordinated way that their average is also updated in its gradient descent. The "dual averaging" algorithm, which has a similar form but performs communication and averaging operations on dual variables, was also proposed (e.g., [472]). These algorithms are known to be computationally inefficient (e.g., [174]), and hence not used in the present work. The critical reason for such slow convergence is that all the agents are required to move in the same pace while their local objectives can be highly different. Different from the push-sum algorithm, in ADMM, the agents can pursue the minima of local objectives autonomously without a pre-conditioned uniform step size.*

In the following example, we show that through the ADMM iterations, the parameters in the approximate expressions of $u^*(x)$ and $V^*(x)$ are updated with decreasing regression residual.

Example 12.1. *We consider the following system:*

$$\dot{x}_1 = -x_1 + x_2, \quad \dot{x}_2 = -\frac{x_1 + x_2}{2} + \frac{x_2}{2}(2 + \cos 2x_1)^2 + (2 + \cos 2x_1)u. \tag{12.20}$$

The system is open-loop unstable. 100 samples of trajectory data are generated starting from random initial conditions in $[-0.5, 0.5] \times [-0.5, 0.5]$ under a stabilizing control $u = x_1 - 5x_2$ for the linearized model. The optimal control cost $V^*(x)$ is approximated using basis functions x_1^2, x_2^2, x_1x_2, and the optimal control $u^*(x)$ is approximated with the basis functions x_1^2, x_2^2, x_1x_2, x_1, x_2. Let $Q(x) = x_1^2 + x_2^2$ and $R = 1$.

For this problem, the push-sum algorithm fails to converge to a stabilizing control. Therefore, we use ADMM (12.19) for the solution of problem (12.17), where the minimization operation is executed by the `fminunc` command in MATLAB. Fig. 12.1 shows the evolution of the average residual $\sqrt{\frac{1}{N}\sum_{i=1}^{N} F_i(z)}$ with increasing iterations. Within 10^5 iterations, the average residual decreases to 3.71% from 426%. The obtained approximate optimal control is

$$u^*(x) = -0.0194x_1^2 + 1.0281x_2^2 - 0.1171x_1x_2 + 0.2796x_1 - 1.7510x_2. \tag{12.21}$$

Starting from multiple initial points, the control given above is applied. The closed-loop system is stable (see Fig. 12.1).

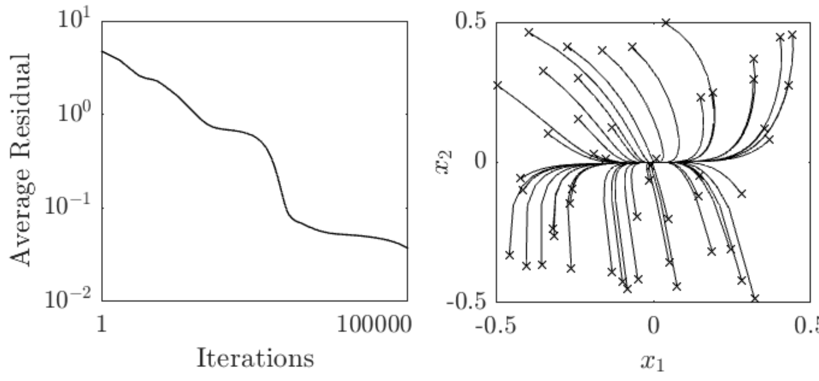


Figure 12.1: Evolution of the average residual during iterations (left) and simulation of the closed-loop system under different initial points (right).

12.3.2 Acceleration of ADMM algorithm

A significant weakness of the above traditional ADMM algorithm (also called *vanilla* ADMM [185]) is its high computational cost. This originates from two reasons – the involvement of minimization procedures and the large number of iterations needed to converge. Hence we adopt the following techniques to improve the computational performance.

Instead of directly minimizing the Lagrangian with respect to z_i, since z_i is optimized near z_0 when the quadratic regularization term $\frac{\mu}{2}\|z_i - z_0\|^2$ exists, we linearize $F_i(z_i)$ near z_0, i.e.,

$$\begin{aligned} F_i(z_i) &\approx F_i(z_0) + \nabla F_i(z_0)^\top (z_i - z_0) \\ &= F_i(z_0) + 2\left(\frac{1}{2} z_0^\top A_i z_0 + b_i^\top z_0 - 1\right)(A_i z_0 + b_i)^\top (z_i - z_0). \end{aligned} \tag{12.22}$$

With this linearization, the minimizer of z_i becomes explicit:

$$\begin{aligned} z_i &\approx \arg\min_{\hat{z}_i} \left[2\left(\frac{1}{2} z_0^\top A_i z_0 + b_i^\top z_0 - 1\right)(A_i z_0 + b_i)^\top (\hat{z}_i - z_0) + \frac{\mu}{2}\|\hat{z}_i - z_0 + \frac{\lambda_i}{\mu}\|^2\right] \\ &= z_0 - \frac{1}{\mu}\left[2\left(\frac{1}{2} z_0^\top A_i z_0 + b_i^\top z_0 - 1\right)(A_i z_0 + b_i) + \lambda_i\right]. \end{aligned} \tag{12.23}$$

This modification is called *proximal* ADMM [310, 82].

We also adopt an acceleration scheme which resembles that of [185] for classical two-block ADMM. This scheme, using Nesterov's technique [290], accelerates the ADMM to a quadratic convergence rate. In the j-th iteration ($j = 1, 2, \dots$), the minimization of z_i and z_0 is not directly based on the primal and dual values from the previous iteration, but on intermediate variables $\hat{z}_0$ and $\hat{\lambda}$ obtained from the primal and dual values from the previous two iterations:

$$\begin{aligned}
z_{i,j} &= \hat{z}_{0,j} - \frac{2}{\mu}\left(\frac{1}{2}\hat{z}_{0,j}^\top A_i \hat{z}_{0,j} + b_i^\top \hat{z}_{0,j} - 1\right)(A_i \hat{z}_{0,j} + b_i) - \frac{\hat{\lambda}_{i,j}}{\mu} \\
z_{0,j} &= \frac{1}{N}\sum_{i=1}^{N}\left(z_{i,j} + \frac{\hat{\lambda}_{i,j}}{\mu}\right) \\
\lambda_{i,j} &= \hat{\lambda}_{i,j} + \mu(z_{i,j} - z_{0,j}) \\
\hat{z}_{0,j+1} &= z_{0,j} + \frac{\nu_j - 1}{\nu_{j+1}}(z_{0,j} - z_{0,j-1}) \\
\hat{\lambda}_{i,j+1} &= \lambda_{i,j} + \frac{\nu_j - 1}{\nu_{j+1}}(\lambda_{i,j} - \lambda_{i,j-1})
\end{aligned} \tag{12.24}$$

where the series $\{\nu_j | j \in \mathbb{N}\}$ is specified by

$$\nu_0 = 1, \ \ \nu_{j+1} = \frac{1}{2}[1 + (1 + 4\nu_j^2)^{1/2}], \ \ j = 0, 1, \dots. \tag{12.25}$$

In the next example, we compare the non-accelerated and accelerated proximal ADMM algorithms to examine the effect of the acceleration method introduced.

Example 12.2. *Consider a second-order system [241]:*

$$\dot{x}_1 = -x_1 + x_2, \ \ \dot{x}_2 = -0.5x_1 - 0.5x_2 + 0.5x_1^2 x_2 + x_1 u. \tag{12.26}$$

Let $Q(x) = x_1^2 + x_2^2$ *and* $R = 1$. *For this system, one can obtain the optimal control by analytically solving the HJB equations:* $u^*(x) = -x_1 x_2$ *and* $V^*(x) = 0.5x_1^2 + x_2^2$. *We use* x_1^2, x_2^2, $x_1 x_2$, x_1, x_2 *as basis functions for* $u^*(x)$, *and* x_1^2, x_2^2, $x_1 x_2$ *as basis functions for* $V^*(x)$. *The samples are obtained by randomly choosing the initial conditions in* $[-1, 1] \times [-1, 1]$ *under the stabilizing control* $u = -0.5x_1 x_2$. *We compare the proximal ADMM, where the minimization of* z_i *is approximated by a linearization* (12.23) *and its accelerated version* (12.24), *starting from the initial guess of* $u^*(x)$ *as* $-2x_1 x_2$ *and of* $V^*(x)$ *as* $2x_1^2 + 2x_2^2$. *Fig. 12.2 shows the evolution of the parameters and the growth of*

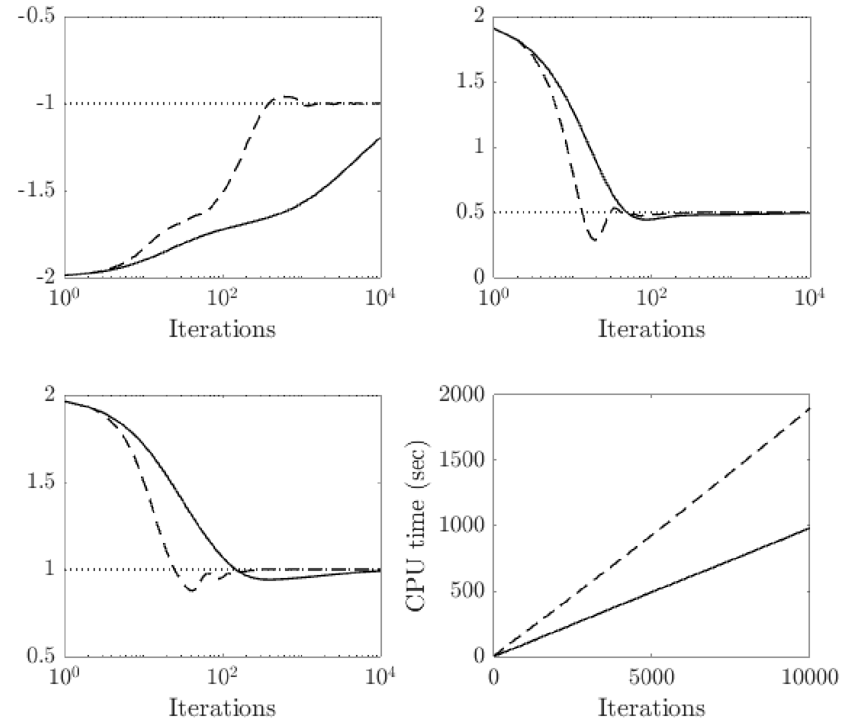


Figure 12.2: The effect of acceleration on the performance of ADMM. Subplot (1,1): Coefficient of x_1x_2 in $u^*(x)$. Subplot (1,2): Coefficient of x_1^2 in $V^*(x)$. Subplot (2,1): Coefficient of x_2^2 in $V^*(x)$. Subplot (2,2): Computational time. The non-accelerated and accelerated ADMM correspond to the solid and dashed line, respectively. The dotted horizontal lines represent the optimum.

computational time during 10000 iterations. It can be observed that although for each iteration, the accelerated algorithm requires about twice computational time of that of the non-accelerated one due to the combination steps (see subplot (2,2) of Fig. 12.2), the coefficients approach the optimal values much faster – the required number of iterations decreases by orders of magnitude. In the remaining text, we will use the accelerated proximal ADMM as the distributed optimization algorithm of choice.

Example 12.3 (Continuously stirred tank reactor, CSTR)**.** *We apply our proposed method to a chemical reactor in which an exothermic first-order reaction $A \to B$ takes place [48]. The dynamics of the system is given as follows, where x_1 and x_2 represent*

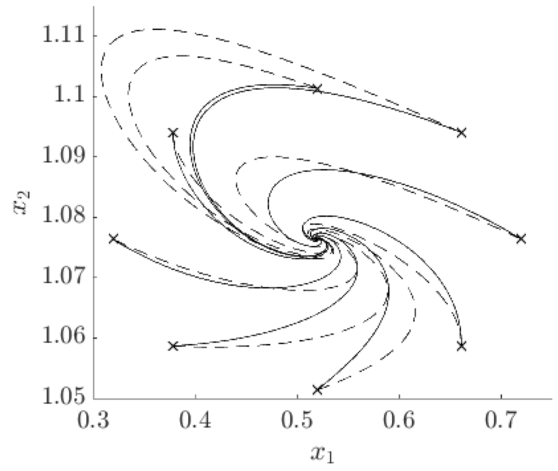


Figure 12.3: Trajectories under the approximate optimal control and suboptimal control. Solid lines and dash lines correspond to $u^*(x)$ and $u^0(x)$, respectively.

the dimensionless reactant concentration and the temperature, respectively.

$$\begin{aligned} \dot{x}_1 &= 1 - x_1 - Dx_1 \exp\left[\zeta\left(1 - 1/x_2\right)\right], \\ \dot{x}_2 &= 1 - x_2 + \varepsilon D x_1 \exp\left[\zeta\left(1 - 1/x_2\right)\right] + \nu(u - x_2). \end{aligned} \tag{12.27}$$

The parameters are $D = 0.0637$, $\zeta = 37.70$, $\nu = 0.1356$, *and* $\varepsilon = 0.1670$. *Given a steady state input* $u^{ss} = 1.0480$, *the system has an open-loop unstable steady state* $x_1^{ss} = 0.5196$, $x_2^{ss} = 1.0764$. *We translate the input and states by* $u := u - u^{ss}$, $x_1 := x_1 - x_1^{ss}$ *and* $x_2 := x_2 - x_2^{ss}$, *and aim to regulate the states towards the origin (the unstable steady state). Let* $Q(x) = x_1^2 + 50x_2^2$ *and* $R = 2.5$. *100 samples for learning are generated with an initial concentration deviation in* ± 0.20 *and a temperature deviation in* ± 0.025 *under a stabilizing control* $u = -8x_2$. *The accelerated proximal ADMM algorithm is applied to optimize the parameters* z. *Starting from an initial guess of* $u^*(x)$ *as* $-10x_2$ *and* $V^*(x)$ *as* $1.25x_1^2 + 176x_2^2 + 32x_1x_2$, *through 10000 iterations, the average residual decreases from* 103.2% *to* 5.07×10^{-5}. *The obtained approximate optimal solutions are*

$$\begin{aligned} u^*(x) &= -0.0010x_1^2 - 0.1117x_2^2 - 0.0231x_1x_2 + 0.3447x_1 - 15.8547x_2 \\ V^*(x) &= 0.3502x_1^2 + 12.4834x_2^2 + 4.1763x_1x_2 \end{aligned} \tag{12.28}$$

The closed-loop system is found to be stable. We perform simulations using several *initial conditions, and compare the trajectories under the approximate optimal control $u^*(x)$ to the suboptimal initial guess $u^0(x) = -x_2$, as shown in Fig. 12.3. It is observed that the trajectories under $u^*(x)$ lead to smaller deviations from the steady state.*

From the result of $u^*(x)$ we see that it suffices to use a linear feedback control as the coefficients of the quadratic terms are very small in magnitude. In other words, more basis functions are used for regression than needed, which leads to unnecessary regression effort, i.e., an overfitting in $u^*(x)$. We also note that the corresponding control cost $V^*(x)$ is not in accordance with the simulated result with $u^*(x)$. For example, starting from $[0.1; 0.01]^\top$, the simulation gives a control cost of 8.61×10^{-2} while $V^*(0.1, 0.01) = 8.93 \times 10^{-3}$. This large discrepancy is due to the underfitting of $V^*(x)$, for which fewer basis functions are used than needed. Specifically, since the system is severely nonlinear away from the steady state, the optimal control cost $V^*(x)$ can not be well approximated by a quadratic function with basis functions x_1^2, x_2^2, and x_1x_2. If an accurate expression of $V^*(x)$ is needed in addition to the optimal control policy, polynomials of higher orders should be added to the basis functions for $V^*(x)$. The above observation suggests that the performance of our proposed method, like that of machine learning algorithms in general, is critically dependent on the choice of basis functions for $u^*(x)$ and $V^*(x)$.

12.4 Input-Constrained Control

In the previous text we assumed the absence of constraints on the control signal, based on which equation (12.4) is derived from Bellman's optimality conditions. Now we consider the case when the inputs are subject to the time-invariant constraints $\rho(u) \leq 0$, where ρ is a smooth vector-valued function. Then

$$u^*(x) = \arg\min_u \ \Big[(\nabla V^*(x))^\top (f(x) + g(x)u) + C(x, u)\Big] \quad \text{s.t. } \rho(u) \leq 0 \tag{12.29}$$

which leads to the Karush-Kuhn-Tucker (KKT) conditions

$$\begin{aligned} &\nabla V^*(x)^\top g(x) u^*(x) + \left(\frac{\partial C}{\partial u}\right)_{(x, u^*(x))} + \lambda^*(x)^\top \left(\frac{d\rho}{du}\right)_{u^*(x)} = 0 \\ &\lambda^*(x) \geq 0, \ \ \rho(u^*(x)) \leq 0, \ \ \lambda_j^*(x) = 0 \text{ if } \rho_j(u^*(x)) < 0 \end{aligned} \tag{12.30}$$

by introducing the optimal dual variables $\lambda^*(x)$ dependent on the state variables x corresponding to the input constraints $\rho(u) \leq 0$. Combining with the second equation in (12.3), this gives, along any trajectory $(x(t), u(t))$:

$$\frac{dV^*}{dt} = \left[\left(\frac{\partial C}{\partial u}\right)_{(x(t),u^*(x(t)))} + \lambda^*(x(t))^\top \left(\frac{d\rho}{du}\right)_{u^*(x(t))} \right] (u^*(x(t)) - u(t)) - C(x(t), u^*(x(t))). \tag{12.31}$$

Consider the most common case when the constraints on the inputs are the upper and lower bounds. Without loss of generality, let the bounds be ± 1. Also assume for the moment that R is a diagonal matrix $R = \text{diag}(r_{11}, \ldots, r_{mm})$. Denote by $\overline{\lambda}^*(x)$ and $\underline{\lambda}^*(x)$ the optimal dual variables corresponding to the upper bound constraints and lower bound constraints on u, and let $\lambda^*(x) = \overline{\lambda}^*(x) - \underline{\lambda}^*(x)$. Then we have

$$\frac{dV^*}{dt} = \left[\left(\frac{\partial C}{\partial u}\right)_{(x(t),u^*(x(t)))} + \lambda^*(x(t))^\top \right] (u^*(x(t)) - u(t)) - C(x(t), u^*(x(t))), \tag{12.32}$$

where any l-th component $\lambda_l^*(x)$, as a function of x, should satisfy the complementary slackness condition:

$$\lambda_l^*(x) \begin{cases} \geq 0, & u_l^*(x) = 1 \\ \leq 0, & u_l^*(x) = -1 \\ = 0, & |u_l^*(x)| < 1 \end{cases}. \tag{12.33}$$

We also assume for simplicity that $C(x, u) = Q(x) + u^\top Ru$ for a positive definite function Q and $R \succeq 0$. Then

$$\frac{dV^*}{dt} = u^*(x(t))^\top R(u^*(x(t)) - 2u(t)) - Q(x(t)) + \lambda^*(x)^\top (u^*(x(t)) - u(t)). \tag{12.34}$$

i.e., for any trajectory $(x(t), u(t))$, it holds that

$$\begin{aligned} &\int_{t'}^{t''} u^*(x(t))^\top R(u^*(x(t)) - 2u(t))dt + V^*(x(t')) - V^*(x(t'')) \\ &- \int_{t'}^{t''} Q(x(t))dt + \int_{t'}^{t''} \lambda^*(x)^\top (u^*(x(t)) - u(t))dt = 0. \end{aligned} \tag{12.35}$$

where $\lambda^*(x)$ is uniquely determined by $u^*(x)$ through the complementary slackness condition (12.33). Comparing (12.35) to its input-unconstrained counterpart (12.10), the difference lies in that for the input-constrained problem, there exists a duality term related to the input constraints.

For the constrained control, it is infeasible to approximate $V^*(x)$, $u^*(x)$ and $\lambda^*(x)$

independently and perform an analogous regression to optimize the coefficients, due to the existence of the complementary slackness condition (12.33), which, for any component index $l = 1, \dots, m$, allows only one of $u_l^*(x)$ and $\lambda_l^*(x)$ to be free. In view of this, we define a *combined optimal primal-dual function* dependent on x:

$$\mu_l^*(x) = u_l^*(x) + \frac{1}{2r_{ll}} \lambda_l^*(x), \tag{12.36}$$

from which the optimal control (primal) and optimal dual variables can be recovered:

$$\begin{cases} u_l^*(x) = 1, \;\; \lambda_l^*(x) = 2r_{ll}(\mu_l^*(x) - 1), & \mu_l^*(x) \geq 1 \\ u_l^*(x) = \mu_l^*(x), \;\; \lambda_l^*(x) = 0, & |\mu_l^*(x)| < 1 \\ u_l^*(x) = -1, \;\; \lambda_l^*(x) = 2r_{ll}(\mu_l^*(x) + 1), & \mu_l^*(x) \leq -1 \end{cases} \tag{12.37}$$

It follows that

$$r_{ll} u_l^*(x)^2 + \lambda_l^*(x) u_l^*(x) = r_{ll} \mathrm{wind}(\mu_l^*(x)) \tag{12.38}$$

where the function $\mathrm{wind}(\cdot)$ is defined on $\mathbb{R}$ as

$$\mathrm{wind}(y) = \begin{cases} 2|y| - 1, & |y| \geq 1 \\ y^2, & |y| < 1 \end{cases} \tag{12.39}$$

which is twice differentiable:

$$\frac{d}{dy}\mathrm{wind}(y) = \mathrm{sat}(y) = \begin{cases} \mathrm{sign}(y), & |y| \geq 1 \\ y, & |y| < 1 \end{cases}, \quad \frac{d}{dy}\mathrm{sat}(y) = \mathbf{1}_{[-1,1]}(y). \tag{12.40}$$

The notations sat and wind stand for "saturation" and "windup", respectively. The function $\mathbf{1}_{[-1,1]}(y)$ is the characteristic function of the interval $[-1, 1]$, valued 1 if $y \in [-1, 1]$ and 0 otherwise. Substituting (12.38) into (12.35), we obtain the data-driven HJB equations in terms of the combined optimal primal-dual function $\mu^*(x)$ and the optimal control cost $V^*(x)$:

$$\begin{aligned} &\sum_{l=1}^{m} \int_{t'}^{t''} \left[r_{ll} \mathrm{wind}(\mu_l^*(x(t))) - 2r_{ll} \mu_l^*(x(t)) u_l(t) \right] dt \\ &+ V^*(x(t')) - V^*(x(t'')) - \int_{t'}^{t''} Q(x(t)) dt = 0, \end{aligned} \tag{12.41}$$

and more generally for a nondiagonal matrix R:

$$\begin{aligned}&\sum_{l=1}^{m}\int_{t'}^{t''} r_{ll}\left[\mathrm{wind}(\mu_l^*(x(t))) - 2\mu_l^*(x(t))u_l(t)\right]dt\\&+\sum_{l\neq l'} 2r_{ll'}\mathrm{sat}(\mu_l^*(x(t)))\mathrm{sat}(\mu_{l'}^*(x(t)))dt\\&+V^*(x(t')) - V^*(x(t'')) - \int_{t'}^{t''} Q(x(t))dt = 0.\end{aligned} \tag{12.42}$$

For the data-driven solution of the optimal control problem, we approximate the $\mu_l^*(x)$ and $V^*(x)$ (instead of $u_l^*(x)$ and $V^*(x)$) with basis functions:

$$V^*(x) \approx \sum_{k=1}^{K_0} \eta_k \psi_k(x), \;\; \mu_l^*(x) \approx \sum_{k=1}^{K_l} \theta_{lk}\phi_{lk}(x). \tag{12.43}$$

The parameters θ, η are obtained through a regression such that for any trajectory $(x(t), u(t))$,

$$\alpha^\top \eta + \Xi(\theta) - 2\beta^\top \theta - 1 \approx 0, \tag{12.44}$$

where the vectors α and β are information extracted from the trajectory samples and defined in the same form as in (12.13). The quadratic form $\theta^\top \Xi \theta$ in (12.13) is now replaced by a more complex form involving saturation and windup functions:

$$\Xi(\theta) = \frac{\Xi^\circ(\theta)}{\int_{t'}^{t''} Q(x(t))dt} \tag{12.45}$$

in which

$$\begin{aligned}\Xi^\circ(\theta) =& \sum_{l=1}^{m} r_{ll}\int_{t'}^{t''} \mathrm{wind}\left(\sum_{k=1}^{K_l}\theta_{lk}\phi_{lk}(x(t))\right)dt\\&+2\sum_{l\neq l'} r_{ll'}\int_{t'}^{t''}\mathrm{sat}\left(\sum_{k=1}^{K_l}\theta_{lk}\phi_{lk}(x(t))\right)\mathrm{sat}\left(\sum_{k=1}^{K_{l'}}\theta_{l'k}\phi_{l'k}(x(t))\right)dt.\end{aligned} \tag{12.46}$$

The regression problem can be solved in a distributed optimization formulation with an accelerated proximal ADMM algorithm, in which the gradient of $\Xi^\circ(\theta)$ is computed using the corresponding data according to

$$
\begin{aligned}
\frac{\partial \Xi^\circ(\theta)}{\partial \theta_{lk}} &= 2r_{ll}\int_{t'}^{t''} \operatorname{sat}\left(\sum_{k=1}^{K_l}\theta_{lk}\phi_{lk}(x(t))\right)\phi_{lk}(x(t))dt \\
&+ 2\sum_{l'\neq l} r_{ll'}\int_{t'}^{t''} \mathbf{1}_{[-1,1]}\left(\sum_{k=1}^{K_l}\theta_{lk}\phi_{lk}(x(t))\right)\operatorname{sat}\left(\sum_{k=1}^{K_{l'}}\theta_{l'k}\phi_{l'k}(x(t))\right)\phi_{lk}(x(t))dt.
\end{aligned}
\tag{12.47}
$$

The optimal control is thereafter obtained from $\mu^*(x)$ according to (12.37), namely $u^*(x) = \operatorname{sat}(\mu^*(x))$.

Remark 12.2. *In previous works (see, e.g., [462]), the data-driven constrained optimal control was considered by replacing the $u^\top Ru$ term in the control objective function with a transformation $\varphi^{-1}(u)^\top R\varphi^{-1}(u)$ where φ is a finite-valued function so that the control cost approaches infinity as any input approaches its bound, e.g., $u = \varphi(v) = \tanh v$, $v \to \pm\infty$ as $u \to \pm 1$. However, due to its distortion of the input domain by the artificial mapping φ, this technique does not give a genuine optimal control with respect to its original control cost $C(x, u)$.*

The above method is illustrated with the following example.

Example 12.4. *Consider the following system [462]:*

$$\dot{x}_1 = -0.4\sin x_1 + 0.6x_2,\ \ \dot{x}_2 = -\sin x_1\cos x_2 + 0.3u \tag{12.48}$$

where the input signal is constrained ($-1 \leq u \leq 1$). Let $Q(x) = x_1^2 + x_2^2$ and $R = 1$. We choose polynomial terms $x_1^a x_2^b$ as the basis functions of $u^*(x)$ for integers $a, b \geq 0, a + b = 1, 3, 5$, and of $V^*(x)$ for integers $a, b \geq 0, a + b = 2, 4, 6, 8$ (36 parameters in total). 300 samples for regression are generated by randomly assigning the initial condition $x_1, x_2 \in [-1.5, 1.5]$ subject to a stabilizing control $u = -\operatorname{sat}(x_2)$. The initial guess for $\mu^*(x)$ and $V^*(x)$ is $-1.5x_2$ and $2x_1^2 + 2x_2^2$, respectively.

After 10000 iterations, the average residual converges from 24.2% to 0.152%, with an approximate optimal control $u^*(x) = \operatorname{sat}(\mu^*(x))$, where

$$
\begin{aligned}
\mu^*(x) = &- 0.0215x_1^5 - 0.0679x_1^4x_2 - 0.0569x_1^3x_2^2 - 0.0299x_1^2x_2^3 - 0.3113x_1x_2^4 + 0.0601x_2^5 \\
&- 0.2384x_1^3 + 0.2061x_1^2x_2 - 0.3299x_1x_2^2 + 0.3438x_2^3 + 0.4147x_1 - 1.4807x_2.
\end{aligned}
\tag{12.49}
$$

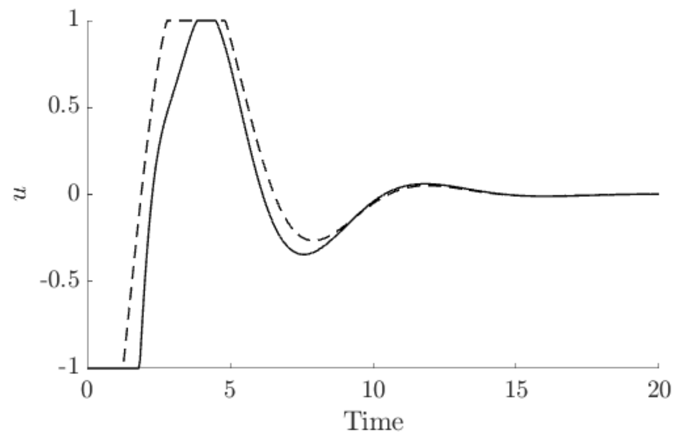


Figure 12.4: Input trajectories under $u^*(x)$ (solid) and its initial guess (dashed).

Fig. 12.4 shows the comparison of input trajectories under the obtained optimal control and the initial guess, starting from the initial point $[1.5; 1.5]$*. It is seen that under* $u^*(x)$*, the input stays on its bounds for a shorter time, and hence leads to a lower control cost.*

12.5 Conclusion

In this work, we have applied distributed optimization algorithms to solve the data-driven optimal control problem of nonlinear affine systems. We first developed an approximation and regression formulation of the optimal control policy $u^*(x)$ and control cost $V^*(x)$. Distributed algorithms, including ADMM and its proximal and a quadratically convergent accelerated variant, were then adopted to optimize the regression coefficients in a data-distributed manner. The input-constrained control problem was also considered, where the optimal control policy $u^*(x)$ and the optimal dual $\lambda^*(x)$ were combined into a single state-dependent function, and regressed together with the optimal control cost $V^*(x)$. Several examples, including a chemical reactor one, were used to illustrate the potential of the proposed method.

Chapter 13
Dissipativity Learning Control (DLC): A Framework of Input–Output Data-Driven Control

13.1 Introduction

Big data analytics is playing an increasing role in the operations and optimization of chemical process systems [338, 432]. Data-driven control, which aims to design controllers based on historical and/or online operational data provides an alternative to model-based control with the potential of circumventing the difficulties of deriving, identifying, updating, and modifying dynamic models [163]. Data-driven modeling approaches, ranging from traditional transfer function identification [211], temporal linearization [62], regression [403, 282], adaptive estimation [151] and Koopman operators [449, 331, 202, 283] to artificial neural networks [1, 279] and machine learning algorithms [261], can in principle be incorporated into model-based control methods to resolve the complexity involved in first-principles modeling. Although intrinsically dependent on data, these approaches are not truly model-free and their efficacy is strongly affected by the complexity and accuracy of the learned surrogate models.

Model-free data-driven control approaches have also been developed, mostly based on approximate dynamic programming (ADP). In these approaches, one focuses on the optimal control policy and/or control cost (or Q-function) as state-dependent functions determined by the Hamilton-Jacobi-Bellman (HJB) optimality principle, and obtains their approximations [216, 215] through either offline regression [241, 407] or online iterative schemes under the name of reinforcement learning (RL) [223, 392]. Instead of obtaining a full dynamic model, in these model-free approaches, one seeks otitessential control-relevant information (e.g., Q-function), thus largely reducing the complexity of designing well-performing controllers. With the development of machine learning, especially deep learning methods [376], ADP and RL approaches are expected to find wider applications.

However, the application of ADP in process control is still limited to small-scale systems with relatively simple dynamics. This is due to the dependence of ADP formulations on the state-space information of the system, which can be limited for chemical processes. For example, for systems with unobservable states, one does not have access to full state information, and the construction of a model-free state estimator is nontrivial (see, e.g., [126]). Further, in the presence of high-dimensional nonlinear dynamics of the states, it is difficult to choose the approximators of the state-dependent optimal control policy and cost functions. It is clear that so far, a data-driven model-free control framework that applies to process systems with possibly unobservable, high-dimensional, and nonlinearly-related states, remains an open problem.

A promising approach to such a framework is to adopt an *input–output* perspective of process systems towards a data-driven control strategy depending only on input and output data without involving any state-space description. To this end, we note that the concept of *dissipativity* [448, 154, 278, 155], as a characterization of input–output behavior, has been widely exploited for output-feedback control [323, 239]. In the context of model-based control, through a thermodynamic analysis on the dynamic model under certain (rather restrictive) assumptions, the dissipative properties of process systems involving a storage function and a supply rate function can be determined by choosing inputs and outputs consistent with irreversible thermodynamics [4, 464, 357, 157] or adopting a Hamiltonian modeling approach [145, 339, 123].

Dissipativity-based control can be naturally extended into an input–output data-driven control strategy, which we call *dissipativity learning control* (DLC). Key to this data-driven framework is the use of machine learning techniques to obtain the dissipative property from data rather than a first-principles model. The learned dissipativity is then combined with a dissipativity-based controller synthesis formulation to obtain a desirable control law. Recently, data-based dissipativity learning approaches have been introduced [439, 253, 353]; however, these works are restricted to specific simple forms of dissipativity properties or linear dynamics, and are not followed by controller design. In [414], we first proposed a dissipativity learning control framework, where a one-class support vector machine for learning the dissipativity property is combined with the controller synthesis. This results in an integrated quadratic and semidefinite programming problem that can be solved via an iterative algorithm, throughout which the dissipativity property is updated until the optimal estimation is approached. However, this procedure is computationally expensive due to the repeated learning, and its performance is dependent on the convergence of the iterative algorithm.

In this work, we propose a more effective approach, where the dissipativity learning and the controller design are carried out successively, thus avoiding the complexity of the iterative algorithm. The learning procedure estimates a *range* of the supply rate, and involves the following three steps: (1) the data samples under the system dynamics are treated with independent component analysis for dimensionality reduction, (2) bi-exponential distribution inference is performed to obtain a polyhedral confidence region of trajectory data, and (3) a dual polyhedral cone is constructed as the approximate range of the parametric representation of the supply rate function. Based on the estimated range of the supply rate, a controller is designed to minimize an upper bound of the L_2-gain by solving a semidefinite programming problem. Such a novel approach can be applied to both regulating and tracking tasks, for the latter of which deviation variables from the time varying input and output trajectories are used instead of deviations from static setpoints.

The remainder of this chapter is organized as follows. We first introduce preliminaries of dissipativity and dissipativity-based control in Section 13.2. The DLC framework is proposed in Section 13.3. We examine the proposed method with case studies on regulating control and tracking control of two different chemical reactors in Section 13.4 and Section 13.5, respectively. Conclusions are given in Section 13.6.

13.2 Preliminaries

13.2.1 Dissipativity

Dissipativity is an important characterization of the input–output property of dynamic systems that describes how the states of the system move across the contours of a nonnegative function under the effect of the input and output variables. Dissipativity, as defined by [448], states that the change of a state-dependent storage function $V(x)$ can not exceed the accumulation of an input and output-dependent supply rate $s(u;y)$.

Definition 13.1. *A dynamic system in the general nonlinear form*

$$\dot{x} = f(x,u), \;\; y = h(x,u). \tag{13.1}$$

is said to be dissipative in the (nonnegative) storage function $V(x)$ with respect to the supply rate $s(u;y)$ if under the system dynamics (13.1)*, the dissipative inequality holds* *for any input trajectory $u(t)$ on any time interval $[t_1, t_2]$:*

$$V(x(t_2)) - V(x(t_1)) \leq \int_{t_1}^{t_2} s(u(t); y(t)) dt. \tag{13.2}$$

Generally, the inputs of the plant include not only control inputs u but also disturbances d, whose components are assumed to be equal to 0 in the nominal plant. Here we consider plants governed by the following (unknown) input-affine dynamics, where x, u and d are vectors:

$$\dot{x} = f(x) + g(x)u + l(x)d. \tag{13.3}$$

For reference tracking problems, the setpoint trajectories of inputs, $\bar{u}$, and outputs, $\bar{y}$, which satisfy the undisturbed dynamics

$$\dot{\bar{x}} = f(\bar{x}) + g(\bar{x})\bar{u}, \;\; \bar{y} = h(\bar{x}), \tag{13.4}$$

may vary with time. For regulating control, the setpoints are fixed at $\bar{u} = 0$, $\bar{y} = 0$. The setpoint signals are given a priori by a dynamic or static simulator of the plant. Define the deviations of the control inputs and the outputs from the corresponding setpoints as $\tilde{u} := u - \bar{u}$ and $\tilde{y} := y - \bar{y}$, respectively. Thus, the plant is viewed as a mapping $(\tilde{u}, d, \bar{u}) \to \tilde{y}$ with states $\tilde{x}$ and $\bar{x}$:

$$\begin{aligned} \dot{\tilde{x}} &= f(\bar{x} + \tilde{x}) + g(\bar{x} + \tilde{x})(\bar{u} + \tilde{u}) + l(\bar{x} + \tilde{x})d - f(\bar{x}, \bar{u}) - g(\bar{x})\bar{u}, \\ \dot{\bar{x}} &= f(\bar{x}) + g(\bar{x})\bar{u}, \;\; \tilde{y} = h(\bar{x} + \tilde{x}) - h(\bar{x}). \end{aligned} \tag{13.5}$$

We consider the controller as a mapping from the output deviations (errors) and the input setpoints to the input deviations $\kappa : (\tilde{y}, \bar{u}) \to \tilde{u}$ to be designed ($\bar{y}$ is not included since it is determined by $\bar{u}$), i.e., we seek

$$\tilde{u} = \kappa(\tilde{y}, \bar{u}). \tag{13.6}$$

Hence the closed-loop system is a map from $(d, \bar{u})$ to the tracking errors in both the inputs and the outputs $(\tilde{y}, \tilde{u})$. The architecture of such a control system is illustrated in Fig. 13.1. Under such an architecture, suppose that the dissipative inequality of the

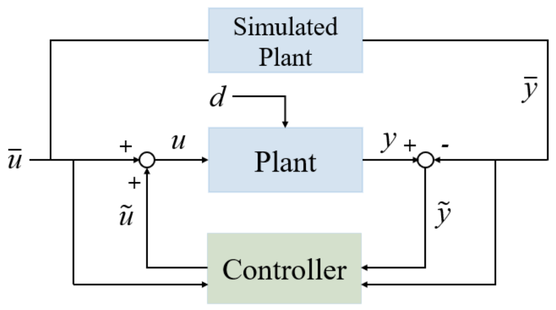


Figure 13.1: System architecture

open-loop system (13.5) is written as

$$V(\tilde{x}(t_2), \bar{x}(t_2)) - V(\tilde{x}(t_1), \bar{x}(t_1)) \leq \int_{t_1}^{t_2} s(\tilde{u}(t), d(t), \bar{u}(t); \tilde{y}(t))dt. \tag{13.7}$$

Substituting the feedback control law (13.6) into the above formula, we see that the closed-loop system, with $(\tilde{x}, \bar{x})$ as its states, is dissipative with respect to a new supply rate $\check{s}$:

$$s(\kappa(\tilde{y}, \bar{u}), d, \bar{u}; \tilde{y}) =: \check{s}(d, \bar{u}; \tilde{y}). \tag{13.8}$$

The closed-loop dissipative inequality is naturally connected to Lyapunov stability. Consider the undisturbed case when $d = 0$. If $\check{s}(0, \bar{u}; \tilde{y}) \leq 0$, then according to the Krasovskii-LaSalle's principle of invariance [193], the system states will converge to an invariant set in which the supply rate remains 0 and the storage reaches its minimum. If $\check{s}(0, \bar{u}; \tilde{y}) \leq 0$ and the equality holds only when $\tilde{y} = 0$, then the afore-mentioned invariant set is such that the tracking errors become zero, i.e., the output tracking control is realized. If we further assume that (13.5) is partially observable in $\tilde{x}$, then we realize state tracking. For regulating control, it suffices to have the inequality hold only for $\bar{u} = 0$. When there exist disturbances, the control performance is characterized by the effect of the disturbances on $(\tilde{y}, \tilde{u})$ in the sense of an L_2-gain. To this end, we have the following assertion.

Theorem 13.1. *If the closed-loop supply rate $\check{s}(d, \bar{u}; \tilde{y})$ satisfies the bounded nonconcavity condition:*

$$\check{s}(d, \bar{u}; \tilde{y}) \leq \beta \|d\|^2 - \|\kappa(\tilde{y}, \bar{u})\|^2 - \|\tilde{y}\|^2, \tag{13.9}$$

for a positive real number β, then the closed-loop system is L_2-stable, with an L_2-gain *(from the disturbances to the input and output tracking errors) no larger than* $\beta^{1/2}$.

Proof. With (13.8) as a supply rate function and the definition of the controller (13.6), on any time interval $[0,T]$ we have

$$V|_{t=T} - V|_{t=0} \leq \int_0^T \check{s} dt \leq \int_0^T (\beta\|d\|^2 - \|\tilde{u}\|^2 - \|\tilde{y}\|^2) dt. \tag{13.10}$$

Rearrange and relax the non-positive $-V|_{t=T}$ term on the right-hand side to obtain

$$\|\tilde{u}\|^2_{L_2[0,T]} + \|\tilde{y}\|^2_{L_2[0,T]} \leq \beta\|d\|^2_{L_2[0,T]} + V|_{t=0}, \tag{13.11}$$

which conforms to the definition of L_2-stability with the L_2-gain bounded by $\beta^{1/2}$. $\square$

This connection between the dissipativity of the closed-loop system and L_2-stability is the theoretical basis of dissipativity-based control to be discussed next.

13.2.2 Dissipativity-based control

The determination of a supply rate function of the plant satisfying the condition (13.9), namely the dissipativity learning procedure, will be addressed in the next section. Now we assume that such a function s is known, and consider the problem of synthesizing a feedback controller in the form of (13.6) with desired closed-loop performance. For simplicity, we assume that the supply rate s is quadratic in $\tilde{u}, d, \tilde{y}$, the feedback law κ is linear in $\tilde{y}$, and hence $\check{s}$ is also quadratic in $d, \tilde{u}, \tilde{y}$, namely

$$\begin{aligned} s(\tilde{u}, d, \bar{u}; \tilde{y}) &= [d^\top \;\; \tilde{u}^\top \;\; \tilde{y}^\top]\Pi(\bar{u})[\cdot], \quad \kappa(\tilde{y}, \bar{u}) = K(\bar{u})\tilde{y}, \\ \check{s}(d, \bar{u}, \tilde{y}) &= [d^\top \;\; (K(\bar{u})\tilde{y})^\top \;\; \tilde{y}^\top]\Pi(\bar{u})[\cdot], \end{aligned} \tag{13.12}$$

where $\Pi(\bar{u})$ is a symmetric matrix called the dissipativity matrix and $K(\bar{u})$ is the feedback gain matrix that may depend on $\bar{u}$. The upper bound of the squared closed-loop L_2-gain is estimated by (13.9), i.e., the smallest positive β such that

$$[d^\top \;\; (K(\bar{u})\tilde{y})^\top \;\; \tilde{y}^\top]\Pi(\bar{u})[\cdot] \leq \beta\|d\|^2 - \|K(\bar{u})\tilde{y}\|^2 - \|\tilde{y}\|^2 \tag{13.13}$$

holds for any d, $\tilde{y}$ and $\bar{u}$, i.e., such that

$$\begin{bmatrix} I & 0 & 0 \\ 0 & K(\bar{u})^\top & I \end{bmatrix} \left(\Pi(\bar{u}) + \begin{bmatrix} -\beta I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ 0 & K(\bar{u}) \\ 0 & I \end{bmatrix} \preceq 0. \tag{13.14}$$

For simplicity, we need to further assume that Π and hence K are independent of $\bar{u}$, so that the above semidefinite inequality does not need to be repeated for all possible (or multiple) values of $\bar{u}$. This is equivalent to choosing an overestimate for the supply rate function s within the possible range of $\bar{u}$. This conservatism aims at designing a control law that is dependent only on deviations and invariant to the reference trajectory, and is usually acceptable as long as such a control law gives satisfactory performance. Thus, we consider the L_2-optimal dissipativity-based controller design as the problem of finding the controller gain K such that the upper bound of the squared L_2-gain β is minimized:

$$\begin{aligned} &\min_K \ \beta \\ &\text{s.t.} \ \begin{bmatrix} I & 0 & 0 \\ 0 & K^\top & I \end{bmatrix} \left(\Pi + \begin{bmatrix} -\beta I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ 0 & K \\ 0 & I \end{bmatrix} \preceq 0. \end{aligned} \tag{13.15}$$

We note that proportional (P), proportional-integral (PI) and proportional-integral-differential (PID) control laws are the three most classical forms in process control. For implementing PID controllers, we need to augment the plant outputs with their integrals and derivatives, i.e., $(y_{\mathrm{P}}, y_{\mathrm{I}}, y_{\mathrm{D}}) = (y, \int_0^t y(\tau)d\tau, dy/dt)$, and the reference outputs into $(\bar{y}_{\mathrm{P}}, \bar{y}_{\mathrm{I}}, \bar{y}_{\mathrm{D}}) = (\bar{y}, \int_0^t \bar{y}(\tau)d\tau, d\bar{y}/dt)$, so that the feedback signals to the controller include the integral and derivative of $\tilde{y}$ (assuming that the function h is differentiable, so that the time derivatives of y and $\bar{y}$ exist). The augmented outputs can be regarded as the output variables of the corresponding augmented plant dynamics (with auxiliary state variables v):

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} f(x,u) \\ h(x) \end{bmatrix}, \quad \begin{bmatrix} y_{\mathrm{P}} \\ y_{\mathrm{I}} \\ y_{\mathrm{D}} \end{bmatrix} = \begin{bmatrix} h(x) \\ v \\ \dfrac{dh(x)}{dx} f(x,u) \end{bmatrix}. \tag{13.16}$$

Given the dissipativity property of the above augmented dynamics, if we find the optimal controller gain matrix K with augmented outputs, then the matrix can be partitioned into $K = [K_{\mathrm{P}}, K_{\mathrm{I}}, K_{\mathrm{D}}]$ so that the feedback control law is expressed as

$$\tilde{u}(t) = K_{\mathrm{P}}\tilde{y}(t) + K_{\mathrm{I}}\int_0^t \tilde{y}(\tau)d\tau + K_{\mathrm{D}}\frac{d\tilde{y}(t)}{dt}. \tag{13.17}$$

In this case, the L_2-optimal PID controller design results from the following problem modified from (13.15):

$$\begin{aligned} &\min_{K_{\mathrm{P}},K_{\mathrm{I}},K_{\mathrm{D}}} \quad \beta \\ &\text{s.t.} \quad [\cdot]^\top \left(\Pi + \begin{bmatrix} -\beta I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \right) \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & K_{\mathrm{P}} & K_{\mathrm{I}} & K_{\mathrm{D}} \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \preceq 0, \end{aligned} \tag{13.18}$$

where Π is now a matrix with 5×5 blocks corresponding to d, $\tilde{u}$, $\tilde{y}_{\mathrm{P}} = \tilde{y}$, $\tilde{y}_{\mathrm{I}} = \int_0^t \tilde{y}(\tau)d\tau$ and $\tilde{y}_{\mathrm{D}} = d\tilde{y}/dt$, respectively.

Remark 13.1. *Although PID is the most widely used type of controllers in practice, its design or tuning usually requires a transfer function representation of the process. In passivity-based control [20], a PID controller is proved to be strictly input-passive (i.e., dissipative with respect to a supply rate of $y^\top u - \epsilon\|u\|^2$ for some $\epsilon > 0$) and results in closed-loop stability if the plant is passive (i.e., dissipative with respect to $y^\top u$). However, this has not been discussed in a dissipativity-based control setting, where the plant may have more general forms of supply rate functions. Moreover, an optimal way of designing the PID controller gain matrices is lacking. These issues are addressed by the proposed formulation* (13.18) *for the L_2-optimal dissipativity-based PID design.*

13.3 Dissipativity Learning Control

13.3.1 Dissipativity set and dual dissipativity set

Now we deal with the problem of determining the dissipativity property of the system. We note that to obtain a dissipative inequality, a storage function $V(\tilde{x}, \bar{x})$ depending on the states of the system (13.5) and an input and output dependent supply rate function $s(\tilde{u}(t), d(t), \bar{u}(t); \tilde{y}(t))$ are needed. The involvement of the state-dependent storage function is undesirable since only input and output data are available. This can be avoided by using a theorem similar to the one proved in [154] under the following assumption.

Assumption 13.1. *For the system* (13.5)*, any state* $(\tilde{x}, \bar{x})$ *is reachable in finite time from a state with zero state tracking error, i.e., there exists a finite time* $T > 0$ *and a trajectory of inputs* $(\tilde{u}(t), d(t), \bar{u}(t))$ *on* $t \in [0, T]$*, such that the state at* $t = 0$ *is* $(0, \bar{x}_0)$ *for some* $\bar{x}_0$*, and the state at* $t = T$ *is* $(\tilde{x}, \bar{x})$*.*

This assumption is not restrictive, since the reachability from zero state tracking error is naturally satisfied as long as the system is controllable, which can be usually guaranteed by appropriate control variable selection.

Theorem 13.2. *Suppose that Assumption 13.1 holds. Then the system* (13.5) *is dissipative with respect to* $s(\tilde{u}, d, \bar{u}; \tilde{y})$ *in a nonnegative storage function* $V(\tilde{x}, \bar{x})$ *satisfying* $V(0, \bar{x}) = 0$ *for any* $\bar{x}$*, if and only if for any trajectory starting from any states* $(\tilde{x}, \bar{x})$ *with* $\tilde{x} = 0$*, the following inequality holds*

$$\int_{t_1}^{t_2} s(\tilde{u}(t), d(t), \bar{u}(t); \tilde{y}(t))dt \geq 0. \tag{13.19}$$

Proof. The necessity is evident by using Definition 13.1. We only prove the sufficiency here. Consider the following function

$$\underline{V}(\tilde{x}, \bar{x}) = \inf_{\substack{(\tilde{u}(t), d(t), \bar{u}(t)),\ t \in [0,T] \\ \tilde{x}(0)=0,\ \tilde{x}(T)=\tilde{x},\ \bar{x}(T)=\bar{x}}} \int_0^T s(\tilde{u}(t), d(t), \bar{u}(t); \tilde{y}(t))dt. \tag{13.20}$$

According to the reachability assumption 13.1, the above function is well-defined and finite. If (13.19) holds, the value of $\underline{V}$ is always nonnegative. Consider any trajectory $(\tilde{u}(t), d(t), \bar{u}(t))$ on any time interval $[t_1, t_2]$ and denote the initial and final states as $(\tilde{x}_1, \bar{x}_1)$ and $(\tilde{x}_2, \bar{x}_2)$, respectively. We then have

$$\underline{V}(\tilde{x}_2, \bar{x}_2) - \underline{V}(\tilde{x}_1, \bar{x}_1) = \inf_{\substack{(\tilde{u}(t), d(t), \bar{u}(t)), t \in [0,T_2] \\ \tilde{x}(0)=0, \tilde{x}(T_2)=\tilde{x}_2, \bar{x}(T_2)=\bar{x}_2}} \int_0^{T_2} sdt - \inf_{\substack{(\tilde{u}(t), d(t), \bar{u}(t)), t \in [0,T_1] \\ \tilde{x}(0)=0, \tilde{x}(T_1)=\tilde{x}_1, \bar{x}(T_1)=\bar{x}_1}} \int_0^{T_1} sdt \tag{13.21}$$

where the first infimum can be relaxed with any trajectory starting from a point with zero tracking error, passing $(\tilde{x}_1, \bar{x}_1)$, and extended by the given trajectory from $(\tilde{x}_1, \bar{x}_1)$

to $(\tilde{x}_2, \bar{x}_2)$. Hence

$$\underline{V}(\tilde{x}_2, \bar{x}_2) - \underline{V}(\tilde{x}_1, \bar{x}_1) \leq \int_{t_1}^{t_2} s(\tilde{u}(t), d(t), \bar{u}(t); \tilde{y}(t))dt. \tag{13.22}$$

According to Definition 13.1, the system (13.5) is then dissipative in the storage function $\underline{V}$ with respect to s. $\square$

Under the quadratic form of supply rate

$$s(\tilde{u}, d, \bar{u}; \tilde{y}) = [d^\top \;\; \tilde{u}^\top \;\; \tilde{y}^\top]\Pi[\cdot], \tag{13.23}$$

the inequality condition (13.19) becomes

$$\int_{t_1}^{t_2} [d^\top(t) \;\; \tilde{u}^\top(t) \;\; \tilde{y}^\top(t)]\Pi[\cdot]dt = \langle \Pi, \int_{t_1}^{t_2} [\cdot][d^\top \;\; \tilde{u}^\top \;\; \tilde{y}^\top]dt \rangle \geq 0, \tag{13.24}$$

where the inner product between any two symmetric matrices $\langle \cdot, \cdot \rangle$ is specified as the trace of their product. Practically, we limit (13.24) to trajectories on which d and $\tilde{u}$ belong to the L_{2e} class (i.e., are finite-time square integrable), so that the integral term in the inner product is finite.

Now we give the key definitions that will be used for dissipativity learning.

Definition 13.2. *The dual dissipativity parameter of each trajectory* $(\tilde{u}(t), d(t), \bar{u}(t), \tilde{y}(t))$, $t \in [t_1, t_2]$ *is defined as*

$$\Gamma = \int_{t_1}^{t_2} [\cdot][d(t)^\top \;\; \tilde{u}(t)^\top \;\; \tilde{y}(t)^\top]dt. \tag{13.25}$$

The collection of dual dissipativity parameters of all the trajectories that start from any $(\tilde{x}, \bar{x})$ *with* $\tilde{x} = 0$ *and are* L_{2e} *(quadratically integrable on any finite time interval) in the inputs* $(d, \tilde{u})$ *is called the dual dissipativity set, denoted as* $\mathcal{S}$*. The dual cone of the dual dissipativity set* $\mathcal{S}$*,*

$$\mathcal{S}^* = \{\Pi | \langle \Pi, \Gamma \rangle \geq 0, \forall \Gamma \in \mathcal{S}\}, \tag{13.26}$$

is called the dissipativity set. We also define the dissipativity parameter of the system as the matrix Π *in the supply rate* (13.23).

Then it directly follows from Theorem 13.2 that the dissipativity set $\mathcal{S}^*$ defined above is the range of the dissipativity parameters. This is stated as the following corollary.

Corollary 13.1. *Suppose that Assumption 13.1 holds. If $\Pi \in \mathcal{S}^*$, then the system* (13.5) *is dissipative with respect to $s(\tilde{u}, d, \bar{u}; \tilde{y}) = [d^\top \;\; \tilde{u}^\top \;\; \tilde{y}^\top]\Pi[\cdot]$.*

Hence, the problem of dissipativity learning refers to the determination of the dissipativity set $\mathcal{S}^*$, which requires only to determine the dual dissipativity set $\mathcal{S}$ – the collection of all possible dual dissipativity parameters Γ. In a model-free setting, $\mathcal{S}$ is constructed by inference from data. Specifically, we collect P independent samples of trajectories $(\tilde{u}^p(t), d^p(t), \bar{u}^p(t); \tilde{y}^p(t))$, $t \in [t_1^p, t_2^p]$, $p = 1, 2, \dots, P$ by randomly generating L_{2e}-class inputs $(d, \tilde{u}, \bar{u})$ and simulating the system dynamics (13.5). Then we calculate for each trajectory sample the corresponding dual dissipativity parameters

$$\Gamma^p = \int_{t_1^p}^{t_2^p} [\cdot][d^p(t)^\top \;\; \tilde{u}(t)^\top \;\; \tilde{y}^p(t)^\top] dt \in \mathcal{S}. \tag{13.27}$$

Then Γ^p, $p = 1, \dots, P$ are samples of a random distribution whose support set (the set on which the probability density is nonzero) is $\mathcal{S}$, as long as the input trajectories are sampled from the L_{2e} class, i.e., any L_{2e} signal has a chance of being chosen. This can in principle be realized, for example as in the present paper, using independent Wiener processes of random magnitudes or Orstein-Uhlenbeck processes, although the optimal or near-optimal sampling methods of input trajectories remain an important open problem. Now the dissipativity learning is formally expressed as:

Problem 13.1. *Given samples Γ^p, $p = 1, 2, \dots, P$, infer the support set $\mathcal{S}$ of the underlying distribution of the samples, and explicitly characterize its dual cone $\mathcal{S}^*$.*

13.3.2 Dissipativity learning approach

To estimate $\mathcal{S}$, one may directly apply a probability density estimation scheme (see, e.g., [314]) or kernel [365] or deep [355] one-class support vector machine algorithms. However, the shape of such an estimated $\mathcal{S}$ can be too complex to explicitly characterize its dual cone $\mathcal{S}^*$ and use it for a subsequent dissipativity-based control mainly due to its non-convexity. In fact, it suffices to obtain a convex hull of $\mathcal{S}$:

$$\mathrm{conv}(\mathcal{S}) = \left\{ \sum_{i=1}^{N} \alpha_i \Gamma_i \middle| \sum_{i=1}^{N} \alpha_i = 1, \;\; \alpha_i \geq 0, \;\; \Gamma_i \in \mathcal{S}, \;\; i = 1, 2, \dots, N, \;\; N \in \mathbb{N} \right\} \supseteq \mathcal{S} \tag{13.28}$$

since the dissipativity set that we aim to find, $\mathcal{S}^*$, is also the dual cone of $\mathrm{conv}(\mathcal{S})$ ($\mathcal{S}^* = \mathrm{conv}(\mathcal{S})^*$). Therefore in this work, we will estimate $\mathcal{S}$ as a *polyhedron*, which can be viewed as the simplest form of convex sets, from trajectory samples.

The key idea underlying the polyhedral estimation is to assume that *the components of* Γ *form a random vector subject to a linear mixture of independent bi-exponential distributions, so that its confidence regions yield polyhedral approximations of its support set*, and that such a mixture of bi-exponential distributions can be inferred through *independent component analysis* (ICA) and *parametric statistical inference* of the component distributions. Here we first represent the matrix Γ and Π isomorphically as vectors γ and π, respectively, by choosing an orthonormal basis in the corresponding matrix space $\{E_k\}$, so that $\langle \Pi, \Gamma \rangle = \pi^\top \gamma$, i.e.,

$$\Gamma = \sum_k \gamma_k E_k, \;\; \Pi = \sum_k \pi_k E_k; \gamma_k = \langle \Gamma, E_k \rangle, \;\; \pi_k = \langle \Pi, E_k \rangle. \tag{13.29}$$

By vectorizing all the samples to γ^p, $p = 1, \dots, P$, Problem 13.1 is restated as

Problem 13.2. *Given samples* γ^p*,* $p = 1, \dots, P$*, find the underlying independent components and infer their bi-exponential distributions, thus explicitly characterizing any confidence set as a polyhedron and its dual cone.*

ICA aims to determine a linear transformation of the translated data samples:

$$\gamma^p = \bar{\gamma} + M\eta^p, \;\; p = 1, 2, \dots, P \tag{13.30}$$

such that η^p, $p = 1, \dots, P$ can be viewed as samples of a random vector η whose components $\eta_j, j = 1, \dots, J$ are independent with zero means and unit variances. $\bar{\gamma} = \frac{1}{P} \sum_{p=1}^P \gamma^p$ is the sample average. The classical algorithm based on kurtosis maximization of η was introduced in [171], to which the readers are referred for details. The dimension of the independent components, denoted by J, is a tunable hyperparameter.

After the ICA processing, we estimate the bi-exponential distribution of each independent component η_j, whose samples are η_j^p, $p = 1, \dots, P$. Specifically, we suppose that the probability density function of η_j as a random variable is a linear combination of two exponential distributions that have a contacting endpoint, opposite directions, and weights summing up to 1:

$$q_j(\eta_j) = \begin{cases} w_j a_j \exp[-a_j(\eta_j - c_j)], \;\; \eta_j \geq c_j \\ (1 - w_j) b_j \exp[-b_j(c_j - \eta_j)], \;\; \eta_j < c_j \end{cases} \tag{13.31}$$

where the 4 involved parameters $a_j > 0$, $b_j > 0$, c_j, $0 < w_j < 1$ are constrained by the following 3 equalities:

$$\begin{aligned} \lim_{\eta_j \to c_j^-} q_j(\eta_j) = \lim_{\eta_j \to c_j^+} q_j(\eta_j) &\quad \text{(continuity);} \\ \int_{-\infty}^{+\infty} \eta_j q_j(\eta_j) d\eta_j = 0 &\quad \text{(zero mean);} \\ \int_{-\infty}^{+\infty} \eta_j^2 q_j(\eta_j) d\eta_j = 1 &\quad \text{(unit variance).} \end{aligned} \tag{13.32}$$

One can verify that the only one remaining degree of freedom can be represented by a parameter $\theta_j \in [-\pi/4, \pi/4]$, with the following expressions relating a_j, b_j, c_j, w_j to θ_j:

$$a_j = 1/\sin(\pi/4 - \theta_j),\ \ b_j = 1/\sin(\pi/4 + \theta_j),\ \ c_j = \sqrt{2}\sin\theta_j,\ \ w_j = (1 - \tan\theta_j)/2. \tag{13.33}$$

The bi-exponential distribution (13.31) is hence

$$q_j(\eta_j) = \frac{1}{\sqrt{2}\cos\theta_j} \exp\left[-\frac{(\eta_j - \sqrt{2}\sin\theta_j)_+}{\sin(\pi/4 - \theta_j)} - \frac{(\eta_j - \sqrt{2}\sin\theta_j)_-}{\sin(\pi/4 + \theta_j)} \right], \tag{13.34}$$

where the subscripts $+$ and $-$ for any real number stand for its positive and negative parts, respectively, namely $p_+ = \max(0, p)$, $p_- = -\min(0, p)$, $p \in \mathbb{R}$. We use the maximum (logarithmic) likelihood estimation to optimize the value of θ, i.e.,

$$\theta_j = \arg\min\ \ \ln(\sqrt{2}\cos\theta_j) + \frac{1}{P}\sum_{p=1}^{P} \frac{(\eta_j^p - \sqrt{2}\sin\theta_j)_+}{\sin(\pi/4 - \theta_j)} + \frac{1}{P}\sum_{p=1}^{P} \frac{(\eta_j^p - \sqrt{2}\sin\theta_j)_-}{\sin(\pi/4 + \theta_j)}. \tag{13.35}$$

With the distributions pf the independent components determined, we see that

$$\zeta = \frac{1}{J}\sum_{j=1}^{J} \left[\frac{(\eta_j - \sqrt{2}\sin\theta_j)_+}{\sin(\pi/4 - \theta_j)} + \frac{(\eta_j - \sqrt{2}\sin\theta_j)_-}{\sin(\pi/4 + \theta_j)} \right] \tag{13.36}$$

as an average of J independent variables, each subject to an exponential distribution of parameter 1 (see (13.34)), is subject to the Erlang distribution whose probability density and cumulative density functions are

$$q(\zeta) = \frac{J^J \zeta^{J-1} \exp(-J\zeta)}{(J-1)!}, \quad Q(\zeta) = 1 - \sum_{k=0}^{J-1} \frac{1}{J!}(J\zeta)^k \exp(-J\zeta) \tag{13.37}$$

respectively, with a mean of 1 and a variance of $1/J$. When J is large, the central limit theorem dictates that the distribution Q is well approximated by a normal distribution $\mathcal{N}(1, J^{-1/2})$. By letting $Q(\zeta) \leq 1 - \epsilon$ for a small positive number ϵ, we obtain a confidence set

$$\sum_{k=0}^{J-1} \frac{1}{J!}(J\zeta)^k \exp(-J\zeta) \geq \epsilon \xrightarrow{\text{denoted as}} \zeta \leq 1 + J^{-1/2}\Delta, \tag{13.38}$$

in which $1-\epsilon$ or Δ characterizes the confidence level. In reality, due to the discrepancy between the empirical distribution of ζ obtained from data samples and the assumed Erlang distribution, such a confidence level Δ needs to be chosen according to a specific portion (e.g., 90% or 95%) of the samples.

By combining the ICA transformation (13.30) and the construction of the Erlang-distributed random variable (13.36), we have a polyhedral approximation of the dual dissipativity set:

$$\mathcal{S}_\Delta = \{\gamma | \gamma = \bar{\gamma} + M\eta, \frac{1}{J}\sum_{j=1}^{J} \left[\frac{(\eta_j - \sqrt{2}\sin\theta_j)_+}{\sin(\pi/4 - \theta_j)} + \frac{(\eta_j - \sqrt{2}\sin\theta_j)_-}{\sin(\pi/4 + \theta_j)} \right] \leq 1 + J^{-1/2}\Delta\}. \tag{13.39}$$

Denote by c the J-dimensional vector whose j-th component is $\sqrt{2}\sin\theta_j$, D^+ and D^- the diagonal matrix of bi-exponential scales whose j-th diagonal entry is $\sin(\pi/4 - \theta_j)$ and $\sin(\pi/4 + \theta_j)$, respectively. Denote by $\mathbf{1}$ and $\mathbf{0}$ the vector with all components equal to 1 and 0, respectively. Then by using the variables σ^+ and σ^- representing the scales of deviations from the contacting endpoint c of the bi-exponential distributions, we have

$$\mathcal{S}_\Delta = \{\gamma | \gamma = \bar{\gamma} + M(c + D^+\sigma^+ - D^-\sigma^-), \mathbf{1}^\top\sigma^+ + \mathbf{1}^\top\sigma^- \leq J + J^{1/2}\Delta, \sigma^+ \geq \mathbf{0}, \sigma^- \geq \mathbf{0}\}. \tag{13.40}$$

Finally, the dissipativity set $\mathcal{S}^*$ is estimated by the dual cone of $\mathcal{S}_\Delta$, calculated using

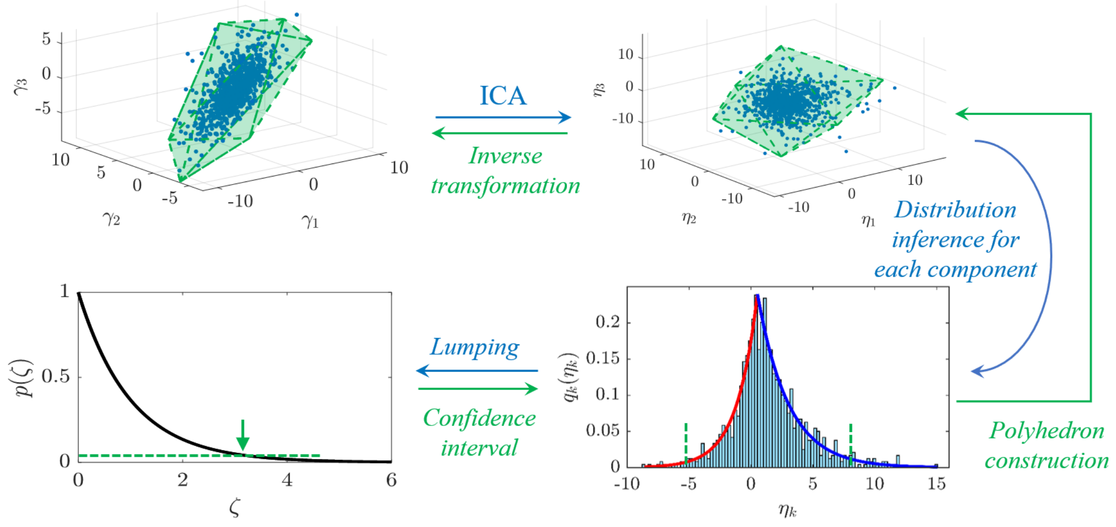


Figure 13.2: Illustration of the proposed dissipativity learning method.

linear duality theory,

$$\mathcal{S}_\Delta^* = \{\pi | \lambda \geq 0, \pi^\top(\bar{\gamma} + Mc) \geq \lambda(J + J^{1/2}\Delta), (MD^+)^\top \pi \geq -\lambda \mathbf{1}, (MD^-)^\top \pi \leq \lambda \mathbf{1}\}. \tag{13.41}$$

Our proposed dissipativity learning method is illustrated in Fig. 13.2. The algorithmic steps are represented by the blue arrows, namely the ICA, the inference of bi-exponential distributions for independent components, and lumping of the independent components into a one-dimensional random variable ζ. By picking a confidence interval of ζ, a polyhedral estimation of $\mathcal{S}$ is acquired by the inverse reasoning steps represented by the green arrows. The learning procedure depends on only two hyperparameters – the number of independent components J and the confidence level Δ.

Remark 13.2. *As will be shown in later case studies, the number of independent components J is determined through a trial-and-error approach to eliminate too large and too small choices that give either overly conservative or loose estimations of the dissipativity property. We note that under different choices of orthonormal basis $\{E_k\}$ to vectorize Π and Γ, the linear transformation linking the groups of basis will be compensated in the mixing matrix M in the ICA step. Hence $\mathcal{S}_\Delta$ and $\mathcal{S}_\Delta^*$, if expressed in terms of the original Γ and Π matrices rather than their vectorization, will be invariant to the choice of basis. In other words, the choice of orthonormal basis $\{E_k\}$ can be arbitrary.*

Remark 13.3. *Since* $\mathcal{S}^* = \operatorname{coni}(\mathcal{S})^*$, *where* $\operatorname{coni}(\mathcal{S})$ *is the conic hull of* $\mathcal{S}$*:*

$$\operatorname{coni}(\mathcal{S}) = \{p\Gamma | p \geq 0, \Gamma \in \operatorname{conv}(\mathcal{S})\} \supseteq \operatorname{conv}(\mathcal{S}) \supseteq \mathcal{S}, \tag{13.42}$$

one may seek to estimate a conic estimation of the dual dissipativity set instead of a bounded polyhedral one as we have done in the main text. This can be done by first scaling the samples onto an affine subspace with a magnitude restriction (e.g., the set of matrices with trace 1), which specifies a section of the conic estimation, and then inferring a polyhedron on this affine subspace. Compared to a polyhedral estimation that is compact in space, the conic counterpart better reflects the intrinsic unboundedness of the dual dissipativity set, as we may have trajectories with sufficiently large input and output signals. However, these trajectories with too large inputs and outputs may not be of interest to characterize system behavior for control purposes, and we tend to avoid them in the data generation procedure due to numerical issues or simulation validity.

Remark 13.4. *The ICA and distribution inference approach proposed here for polyhedral estimation of the dual dissipativity set is motivated by [485] to approximate the feasible region of optimization problems for establishing surrogate models, where convex hulls of data points sampled from the region are found and refined, and the work of [299] to construct the polyhedral uncertainty set for robust optimization, which involves a PCA and a nonparametric distribution inference by kernel smoothing.*

Remark 13.5. *The assumption of bi-exponential distributions of independent components of* η *can be replaced by the following bi-normal distributions, which gives confidence regions that are linearly transformed ellipsoids following our procedures:*

$$q_j(\eta_j) = \begin{cases} \frac{w_j}{\sqrt{2\pi}a_j} \exp\left[-\dfrac{(\eta_j - c_j)^2}{2a_j^2}\right], & \eta_j \geq c_j \\ \frac{1-w_j}{\sqrt{2\pi}b_j} \exp\left[-\dfrac{(c_j - \eta_j)^2}{2b_j^2}\right], & \eta_j < c_j \end{cases} \tag{13.43}$$

13.3.3 Dissipativity learning control

The dissipativity learning controller design problem can be derived by incorporating the dissipativity set estimation (13.41) into the L_2-optimal control formulation (13.15). However, the trajectory samples are generated on a *finite* time interval, and, although starting with zero tracking error, they end up *finitely distant* from the target trajectory.

As a result, the dissipativity parameters learned based on trajectories away from the target may fail to characterize the system behavior near the target and guide us to a stabilizing controller. Therefore, in addition to the previously constructed estimation of the dissipativity set, we need an additional constraint that requires the dissipativity parameters to satisfy the condition (13.19) for infinitesimal trajectories starting from zero tracking error. Apparently, it suffices that the submatrix of Π corresponding to inputs $(d, \tilde{u})$ be positive semidefinite. This means that when the tracking error is zero, the storage function has reached its minimum and any nonzero control inputs or disturbances will increase the storage function. Thus we have reached the following formulation of dissipativity learning control:

$$\begin{aligned} \min_{\Pi,K,\beta} \ \beta \quad \text{s.t.} \ & \begin{bmatrix} I & 0 & 0 \\ 0 & K^\top & I \end{bmatrix} \left(\Pi + \begin{bmatrix} -\beta I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ 0 & K \\ 0 & I \end{bmatrix} \preceq 0 \\ & \Pi_{d\tilde{u},d\tilde{u}} \succeq 0, \ \ \Pi \in \mathcal{S}^*. \end{aligned} \tag{13.44}$$

Substituting the true dissipativity set $\mathcal{S}$ with the learned polyhedral conic approximation $\mathcal{S}^*_\Delta$ (13.41), and expanding the descriptive definition of $\mathcal{S}^*_\Delta$ combined with the basis expansion (13.29), the formulation becomes

$$\begin{aligned} \min_{\Pi,\pi,K,\lambda,\beta} \ \beta \quad \text{s.t.} \ & \begin{bmatrix} I & 0 & 0 \\ 0 & K^\top & I \end{bmatrix} \left(\Pi + \begin{bmatrix} -\beta I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ 0 & K \\ 0 & I \end{bmatrix} \preceq 0 \\ & \Pi_{d\tilde{u},d\tilde{u}} \succeq 0, \ \ \Pi = \sum_k \pi_k E_k \\ & \lambda \geq 0, \ \ \pi^\top(\bar{\gamma} + Mc) \geq \lambda(J + J^{1/2}\Delta) \\ & (MD^+)^\top \pi \geq -\lambda \mathbf{1}, \ \ (MD^-)^\top \pi \leq \lambda \mathbf{1}. \end{aligned} \tag{13.45}$$

Remark 13.6. *It is not hard to see that if the Assumption 13.1 on the reachability of any state from a state of zero tracking error does not hold, the above-mentioned formulation can still be utilized to obtain an L_2-optimal controller that works as long as the initial state of the plant is reachable from a hypothetical state of zero tracking error under some input trajectories.*

The above formulation is a non-convex semidefinite programming problem, which can not be simplistically tackled by the available convex optimization solvers such as `cvx` [133]. The non-convexity arises from the trilinear semidefinite inequality involving the controller gains K twice together with the dissipativity parameters Π and upper bound of squared L_2-gain β. However, the problem is multi-convex – once K is fixed, the rest of the problem on $(\Pi, \pi, \lambda, \beta)$ is convex; once $(\Pi, \pi, \lambda, \beta)$ is fixed, as long as $\Pi_{d\tilde{u},d\tilde{u}} \succeq 0$ is satisfied, we have $\Pi_{\tilde{u},\tilde{u}} \succeq 0$ and hence the remaining problem on K is a convex (quadratic) feasibility problem. We therefore adopt an iterative algorithm to solve (13.45), where each iteration involves the following two steps:

(a) fix K to solve $(\Pi, \pi, \lambda, \beta)$ to the optimum;

(b) seek a different K satisfying the first constraint of (13.45), so that after changing K the solution obtained in step (a) is still feasible.

When we execute step (a) in the next iteration, the β is updated from the previous feasible solution to the optimum under the new K. Therefore, the iterations of steps (a) and (b) lead to a sequence of feasible solutions with non-increasing values of β. Specifically, since the left-hand side of the first constraint of (13.45) is dependent on K through its bottom right principal minor, $K^\top(\Pi_{\tilde{u},\tilde{u}}+I)K+K^\top\Pi_{\tilde{u},\tilde{y}}+\Pi_{\tilde{u},\tilde{y}}K+(\Pi_{\tilde{y},\tilde{y}}+I)$, which is a quadratic form of K, the different K in step (b) can always be chosen as the one such that the quadratic form is the most negative definite. In other words, step (b) updates K according to

$$K = -(\Pi_{\tilde{u},\tilde{u}} + I)^{-1}\Pi_{\tilde{u},\tilde{y}}. \tag{13.46}$$

If the quantities M, D^+, D^-, c are obtained from the ICA using the $\tilde{u}$ and $(\tilde{y}_\mathrm{P}, \tilde{y}_\mathrm{I}, \tilde{y}_\mathrm{D})$ as inputs and outputs, the formulation (13.45) can also be extended to an L_2-optimal dissipativity learning PID controller:

$$\begin{aligned}
&\min\ \beta \\
&\text{s.t.}\ [\cdot]^\top \left(\Pi + \begin{bmatrix} -\beta I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\right) \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & K_\mathrm{P} & K_\mathrm{I} & K_\mathrm{D} \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \preceq 0 \\
&\qquad \Pi_{d\tilde{u},d\tilde{u}} \succeq 0,\ \ \Pi = \sum_k \pi_k E_k,\ \ \pi^\top(\bar{\gamma} + Mc) \geq \lambda(J + \Delta J^{1/2}) \\
&\qquad \lambda \geq 0,\ \ (MD^+)^\top \pi \geq -\lambda \mathbf{1},\ \ (MD^-)^\top \pi \leq \lambda \mathbf{1}.
\end{aligned} \tag{13.47}$$

The solution algorithm has no formal difference expect that the update of $K = [K_{\mathrm{P}}, K_{\mathrm{I}}, K_{\mathrm{D}}]$ is expressed as $K = -(\Pi_{\tilde{u},\tilde{u}} + I)^{-1}\Pi_{\tilde{u},\tilde{y}_{\mathrm{P}}\tilde{y}_{\mathrm{I}}\tilde{y}_{\mathrm{D}}}$.

Remark 13.7. *Although the properties of multiconvex optimization algorithms have been discussed in some recent works (see, e.g., [375]), it appears that the theoretic convergence of the above algorithm using simple iterations is an open problem. However, likely due to the exploitation of the quadratic constraint on K, our algorithm achieves very fast practical convergence (within 20 iterations for the case studies in the following sections), and is therefore suitable for use.*

We summarize the entire procedure of dissipativity learning control as follows.

1. Preliminaries.
   (a) Generate data samples.
   (b) Calculate dissipativity parameter samples Γ^p, $p = 1, 2, \dots, P$ according to (13.27).
   (c) Choose orthonormal matrix bases $\{E_k\}$ and vectorize the Γ^p into γ^p according to (13.29).
   (d) Set the number of independent components J and confidence level Δ.
   (e) Initialize controller gains K.
2. Dissipativity learning.
   (a) Perform ICA and return average $\bar{\gamma}$, mixing matrix M and transformed samples η^p in (13.30).
   (b) For each component j, optimize the maximum likelihood estimate of the parameter θ_j through (13.35).
   (c) Obtain vector c with components $c_j = [\sqrt{2}\sin\theta_j]$, and matrices $D^+ = \mathrm{diag}(1/\sin(\pi/4 - \theta_j))$, and $D^- = \mathrm{diag}(1/\sin(\pi/4 + \theta_j))$.
3. Controller design.
   (a) With fixed K, solve (13.45) for P control or (13.47) for PID control and update $(\Pi, \pi, \lambda, \beta)$.
   (b) With fixed $(\Pi, \pi, \lambda, \beta)$, update K by $K = -(\Pi_{\tilde{u},\tilde{u}} + I)^{-1}\Pi_{\tilde{u},\tilde{y}}$ for P control and $K = -(\Pi_{\tilde{u},\tilde{u}} + I)^{-1}\Pi_{\tilde{u},\tilde{y}_{\mathrm{P}}\tilde{y}_{\mathrm{I}}\tilde{y}_{\mathrm{D}}}$ for PID control.
   (c) If the updated K does not have a sufficiently small deviation from the previous K, return to (a) to iterate. Otherwise terminate.

13.4 Case study: Dissipativity Learning Regulating Control of a Polymerization Reactor

In this section we perform a case study of our proposed dissipativity learning control method on a continuously stirred tank reactor (CSTR) with exothermic polymerization reactions of methyl methacrylate taking place, which was used as a benchmark for nonlinear geometric control [75, 388] due to its highly nonlinear dynamics. Here we consider the regulating control of such a reactor ($\bar{u} = 0$, $\bar{y} = 0$) since the polymerization extent needs to be held constant for the polymer product.

13.4.1 System description

The system model involves 6 states, 2 inputs, 2 outputs and 2 exogenous disturbances. The 6 states represent the monomer and initiator concentrations, reactor and jacket temperatures, amount of substance and mass of the product. The initiator feed and the cold water flow rates are used as control inputs ($u_1 = F_\mathrm{i}$, $u_2 = F_\mathrm{w}$). The average molecular weight and temperature are outputs ($y_1 = D_1/D_0$, $y_2 = T$, where D_0 and D_1 are the molar and mass concentration of the polymer products, respectively). There are two disturbances – monomer concentration and temperature of the feed stream ($d_1 = C_\mathrm{m,in}$, $d_2 = T_\mathrm{in}$).

The governing equations are given as follows:

$$\begin{aligned}
\dot{C}_\mathrm{m} &= -(k_\mathrm{p} + k_\mathrm{m})C_\mathrm{m}\phi + \frac{F}{V}(C_\mathrm{m,in} - C_\mathrm{m}) \\
\dot{C}_\mathrm{i} &= -k_\mathrm{i}C_\mathrm{i} + \frac{F_\mathrm{i}C_\mathrm{i,in} - FC_\mathrm{i}}{V} \\
\dot{T} &= k_\mathrm{p}C_\mathrm{m}\phi\frac{-\Delta H_\mathrm{p}}{\rho c_p} - \frac{UA}{\rho c_p V}(T - T_\mathrm{j}) + \frac{F}{V}(T_\mathrm{in} - T) \\
\dot{T}_\mathrm{j} &= \frac{F_\mathrm{w}}{V_\mathrm{w}}(T_\mathrm{w} - T_\mathrm{j}) + \frac{UA}{\rho_\mathrm{w}c_\mathrm{w}V_\mathrm{w}}(T - T_\mathrm{j}) \\
\dot{D}_0 &= \left(\frac{1}{2}k_\mathrm{c} + k_\mathrm{d}\right)\phi^2 + k_\mathrm{m}C_\mathrm{m}\phi - \frac{FD_0}{V} \\
\dot{D}_1 &= (k_\mathrm{p} + k_\mathrm{m})C_\mathrm{m}\phi M - \frac{FD_1}{V}
\end{aligned} \tag{13.48}$$

The reaction rate constants $k_\star$ for termination by coupling (c), disproportionation (d), initiation (i), propagation (p) and chain transfer to monomer (m) are expressed in the

Table 13.1: Parameters and nominal input and state values for the polymerization reactor system

Par.	Value	Par.	Value
A_c	3.8223×10^{10} kmol/(m^3·h)	f^*	0.58
A_d	3.1457×10^{11} kmol/(m^3·h)	F	1.00 m^3/h
A_i	3.7920×10^{18} h^{-1}	ρ	866 kg/m^3
A_p	1.7700×10^{9} kmol/(m^3·h)	$C_\mathrm{i,in}$	6.0 kmol/m^3
A_m	1.0067×10^{15} kmol/(m^3·h)	R	8.314 J/(mol·K)
E_c	2944.2 kJ/kmol	ΔH_p	-57.8 kJ/mol
E_d	2944.2 kJ/kmol	V	0.1 m^3
E_i	128770 kJ/kmol	M	100.12 kg/kmol
E_p	18283 kJ/kmol	c_p	2.0 kJ/(kg·K)
E_m	74478 kJ/kmol	T_w	293.2 K
U	720 kJ/(h·K·m^2)	A	2.0 m^2
c_w	4.2 kJ/(kg·K)	ρ_w	10^3 kg/m^3
V_w	0.02 m^3		
Input	**Nom. value**	**Input**	**Nom. value**
F_i	0.01679 m^3/h	F_w	3.26363 m^3/h
State	**Nom. value**	**State**	**Nom. value**
C_m	7.7697 kg/kmol	C_i	0.1143 kg/kmol
T	329.98 K	T_j	296.67 K
D_0	3.5155×10^{-4} kmol/m^3	D_1	23.061 kg/m^3
Dist.	**Nom. value**	**Dist.**	**Nom. value**
$C_\mathrm{m,in}$	8.0 kmol/m^3	T_in	350 K

form of Arrhenius law:

$$k_\star = A_\star \exp(-E_\star/RT). \tag{13.49}$$

The molar fraction of live monomer chains ϕ is specified by the quasi-equilibrium assumption:

$$\phi = \sqrt{\frac{2f^* C_\mathrm{i} k_\mathrm{i}}{k_\mathrm{d} + k_\mathrm{c}}}. \tag{13.50}$$

The parameters and nominal states are given in Table 13.1. The inputs, outputs, disturbed variables and time are translated with the corresponding nominal values (so that the origin is the steady state to be regulated at) and scaled by 0.001 m^3/h, 1 m^3/h, 1000 kg/kmol, 1 K, 1 kmol/m^3, 1 K and 0.1 h, respectively.

13.4.2 Data generation and dissipativity learning

The trajectory samples are generated using random walks. Independent Wiener processes of random magnitudes uniformly distributed in $[0,1]$ are assigned to u_1, u_2, d_1 and d_2 in the time interval $[0,1]$ to simulate the system starting from the origin. 3000 independent trajectories are sampled, from which the dual dissipativity parameters Γ^p are calculated by (13.27). With 2 disturbances, 2 control inputs and 2 outputs, each Γ^p is a symmetric r-th order matrix ($\Gamma \in \mathbb{S}^r$) with $r = 6$. We choose the orthonormal bases ($r(r+1)/2 = 21$ in total) for $\mathbb{S}^r$ according to

$$E_{\frac{1}{2}(2r-i)(i-1)+j,kl} = \begin{cases} \delta_{kl}/\sqrt{r}, \;\; i=j=1 \\ \delta_{kl}/\sqrt{i(i-1)}, \;\; i=j\neq 1, k<i \\ -(i-1)\delta_{kl}/\sqrt{i(i-1)}, \;\; i=j\neq 1, k=i \\ 0, \;\; i=j\neq 1, k>i \\ (\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk})/\sqrt{2}, \;\; i\leq j \end{cases} \tag{13.51}$$

for $1 \leq i \leq j \leq r$ and $1 \leq k, l \leq r$ to vectorize the matrices Γ^p into $\gamma^p \in \mathbb{R}^{21}$, where the Kronecker's δ symbol δ_{kl} for any two subscripts k and l equals 1 if $k = l$ and 0 otherwise.

We first pick the number of independent components J equal to the dimension of sample vectors, i.e., $J = r(r+1)/2 = 21$ for ICA. Subplots 2–22 in Fig. 13.3 show the histograms of the obtained components for all the samples, with the probability density function of their inferred bi-exponential distributions shown as red curves, respectively. The accordingly computed samples of the lumped random variable ζ as defined in (13.36) and its inferred distribution are shown in the first subplot of Fig. 13.3. It is observed that compared to the inferred bi-exponential distribution, the empirical distribution of the samples has apparently higher densities near the centers (c_j in (13.31)) but lower densities at moderate distances from the centers. To keep the empirical and the inferred distributions having the same variance equal to 1, the empirical distributions have longer tails than the inferred distributions. The deviations in η_j result in the significant difference between the empirical and inferred distributions of ζ. In principle, if our aim is only to capture the dissipative property of the system based on the data samples, we may seek to modify the bi-exponential distribution by

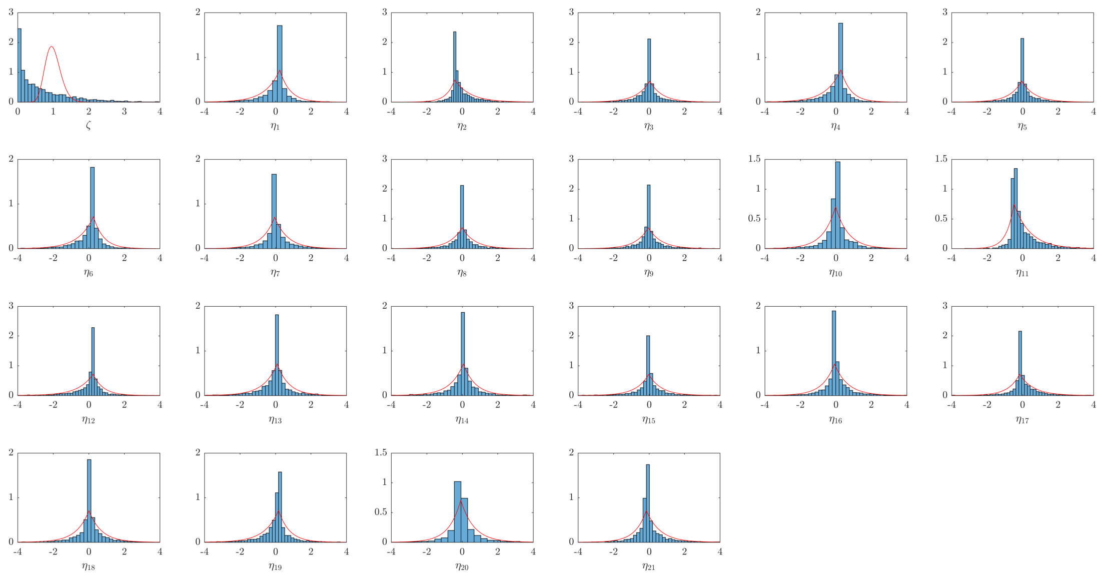


Figure 13.3: Distribution of the independent component parameters and the lumped random variable ζ when $J = 21$ for the polymerization reactor.

asymmetric generalized normal distributions:

$$q_j(\eta_j) = \begin{cases} w_j \mu_j \exp(-a_j |\eta_j - c_j|^{\alpha_j}), & \eta_j \geq c_j \\ (1 - w_j)\nu_j \exp(-b_j |\eta_j - c_j|^{\alpha_j}), & \eta_j < c_j \end{cases} \tag{13.52}$$

with smaller powers $\alpha_j < 1$ and optimized parameters a_j, b_j, c_j, w_j and normalizing constants μ_j, ν_j. However, when $\alpha_j < 1$, the confidence region obtained by the lumped random variable $\eta = \sum_{j=1}^{J} a_j |\eta_j - c_j|^{\alpha_j}$ is not a convex set, and will cause computational intractability of the dissipativity learning control problem (13.45) or (13.47). Unfortunately, this does not seem to improve by changing the number of independent components J.

After ICA, we construct the polyhedral estimations of the dual dissipativity set and subsequently the dissipativity set as in (13.41). Since the empirical distribution of ζ is different from an Erlang or normal distribution, the confidence level Δ needs to be assigned according to the empirical distribution of ζ rather than using the properties of the Erlang or normal distributions such as the 3σ rule of thumb. Instead, due to the long-tail feature, we should make Δ larger than the corresponding value to the desired confident level under the Erlang or normal distribution. For example, if we need

Table 13.2: Minimized leading eigenvalue of the output blocks of the dissipativity matrix Π under different numbers of independent components.

J	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
$\rho_{\max}(\Pi_{\tilde{y},\tilde{y}})$	$-\infty$	$-\infty$	-0.5917	-0.4034	-0.4022	-0.3923	-0.3227	-0.1476	-0.0689	-0.0684	0.3099	0.3964	0.5324	$+\infty$	$+\infty$

to cover 95% of the samples, the empirical distribution requires to choose $\Delta \geq 6.5762$, which is significantly higher than the value of 1.6449 required by the normal distribution $\mathcal{N}(1, J^{-1/2})$.

For $J = 21$ and Δ corresponding to 95% of the samples, we found that the origin is an interior point of $\mathcal{S}_\Delta$ and therefore $\mathcal{S}_\Delta^*$ is a singleton of $\Pi = 0$, which is physically meaningless since the supply rate and hence the storage function are constantly zero. This can be avoided only when Δ is as small as to cover 2% of the samples, which fails to capture the true input–output response of the system. Therefore, the independent component number J must be well-tuned so that a meaningful dissipativity set can be generated under a sufficiently high confidence level Δ. For this, we vary J and always set Δ at the value to cover 95% of the ζ samples. For each J, we find the element $\Pi \in \mathcal{S}_\Delta^*$ that minimizes the leading eigenvalue of the output blocks:

$$\begin{aligned} \min_{\Pi} \ & \rho_{\max}(\Pi_{\tilde{y},\tilde{y}}) \\ \text{s.t.} \ & \Pi \in \mathcal{S}_\Delta^*, \ \ \text{trace}(\Pi_{d\tilde{u},d\tilde{u}}) = 1 \end{aligned} \tag{13.53}$$

where the last constraint is imposed for a fair comparison since $\mathcal{S}_\Delta^*$ is a cone. The Π thus determined can be roughly considered as the least conservative estimate of the dissipativity matrix. Their traces are compared in Table 13.2.

From Table 13.2 we observe that both too large and too small numbers of independent components are undesirable. On one hand, when J is large, the dissipativity learning procedure to construct the dual dissipativity set $\mathcal{S}_\Delta$ takes into consideration insignificant dimensions of the data samples, and results in an overly conservative estimation of $\mathcal{S}_\Delta$ (i.e. the constructed polyhedron is much larger than it need be), thus

Table 13.3: Optimal control performances (β) under different number of independent components and confidence levels for the polymerization reactor.

Δ	85%	90%	95%	98%	99%
$J = 10$	0.1919	1.5152	1.5152	1.5152	1.5152
$J = 11$	0.4755	0.4844	0.5275	0.5586	0.5783
$J = 12$	0.7323	0.7589	0.7904	0.8139	0.8297
$J = 13$	5.3541	5.3541	5.3541	5.3541	5.3541

making its dual cone $\mathcal{S}^*_\Delta$ too small (which reduces to a singleton when $J = 20$ or 21). On the other hand, if J is too small, important dimensions of the data are ignored, and hence the dissipativity learning does not responsibly capture the system dynamics and gives naively radical estimates, such as in the case of $J = 7$ or 8, when $\mathcal{S}^*_\Delta$ allows a dissipativity matrix with $\Pi_{d\tilde{u},d\tilde{u}} = 0$ and $\Pi_{\tilde{y},\tilde{y}} \prec 0$, meaning that the storage can not increase anyhow. When $J = 10$, 11 and 12, the minimized $\rho_{\max}(\Pi_{\tilde{y},\tilde{y}})$ results are similar, which suggest a proper range of J. This can be further justified by the covariance matrix of the samples γ^p, $p = 1, \dots, P$:

$$\Sigma = \frac{1}{P-1}\sum_{p=1}^{P}(\gamma^p - \bar{\gamma})(\gamma^p - \bar{\gamma})^\top, \tag{13.54}$$

whose 11th eigenvalue in descending order, 0.0059, is an order-of-magnitude smaller than the 10th eigenvalue 0.0572, and the eigenvalues after the 13th are smaller than 1% of the leading eigenvalue, 0.1944, implying that there are about 10–13 orthogonal dimensions in which the data variations significantly exist.

13.4.3 Controller design and simulation

Now we use different values of J between 10 and 13 and different values of Δ to cover different percentages of the data samples, and solve the L_2-optimal P control problem (13.45) to obtain the guaranteed upper bounds of the squared L_2-gain, β. The results are shown in Table 13.3. It can be observed that J should be chosen as 11 or 12, so that the confidence level Δ can be found to have a non-trivial effect on the control performance. Since $J = 11$ gives lower upper bound on the L_2-gain than the other 3 values of J, in the sequel we set $J = 11$.

When the degree of polymerization (reflecting the rate of reactions) and the temperature deviate from their setpoints, since the reaction rates increase with temperature and the reactions are exothermic, the deviations with opposite (same) signs annihilate (exacerbate) each other to approach (escape) the setpoints, resulting in a decrease (an increase) of the storage function. To reflect this dynamic feature of the system, the output block $\Pi_{\tilde{y},\tilde{y}}$ should have one negative eigenvalue whose associated eigenvector has two components with opposite signs, and one positive eigenvalue whose associated eigenvector lies in the first quadrature. Also, the controller gain K should have 4 positive elements so that the controller signals are small when the output deviations have the same signs and large when the output deviations are opposite. We note that this is true for the optimized results obtained from the L_2-optimal controller problem (13.45), e.g., when the confidence level is 95%, we have

$$K = \begin{bmatrix} 1.5424 & 2.1511 \\ 1.0371 & 0.7978 \end{bmatrix} \tag{13.55}$$

and

$$\Pi = \begin{bmatrix} 0.3970 & 0.1460 & 0.4864 & -0.2130 & -0.3465 & -1.6909 \\ 0.1460 & 0.3642 & 0.6708 & -1.0100 & -0.1917 & 0.2739 \\ 0.4864 & 0.6708 & 1.3754 & -1.7372 & -1.8620 & -3.7239 \\ -0.2130 & -1.0100 & -1.7372 & 2.9100 & -1.3759 & 0.6174 \\ -0.3465 & -0.1917 & -1.8620 & -1.3759 & 3.0431 & 6.2436 \\ -1.6909 & 0.2739 & -3.7239 & 0.6174 & 6.2436 & 1.4358 \end{bmatrix}. \tag{13.56}$$

The same physical explanations can be made for $\Pi_{\tilde{u},\tilde{u}}$ and $\Pi_{d,d}$. The eigenvectors associated with the larger eigenvalue of $\Pi_{\tilde{u},\tilde{u}}$ and $\Pi_{d,d}$ should have opposite signs and same signs, respectively, since the two control inputs have opposite effects (feeding initiator accelerates polymerization while cooling water decelerates reactions) and the two disturbances have the same effects (increasing monomer feed and increasing feed temperature both accelerate reactions). These physical requirements are satisfied by the obtained Π with $\Pi_{\tilde{u}_1,\tilde{u}_2} < 0$ and $\Pi_{d_1,d_2} > 0$.

We simulate the dissipativity learning controllers obtained under confidence levels 85%, 90%, 95% and 99%, along with the open-loop system with feedback gain equal to the zero matrix. The disturbance signals d_1 and d_2 that we use are two independent zero-mean Orstein-Uhlenbeck processes, defined as

$$dU = -\omega U dt + v dW,\ \ U(0) = 0, \tag{13.57}$$

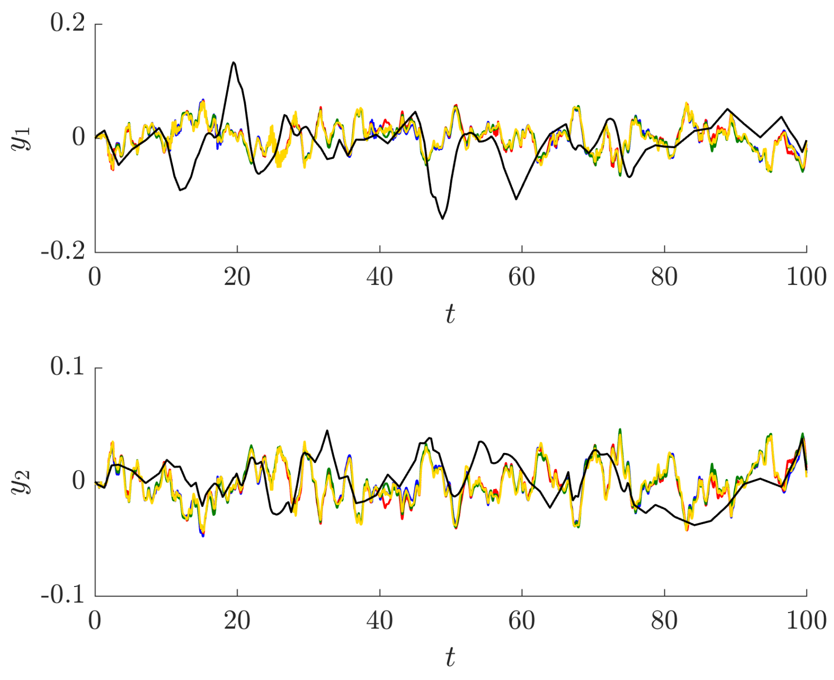


Figure 13.4: Process simulation for dissipativity learning controllers under disturbances. The blue, red, green, and yellow lines correspond to confidence levels 85%, 90%, 95% and 99%, respectively, under $J = 11$. The black lines correspond to the open-loop system ($u = 0$).

where dW denotes the Wiener process, and $\upsilon = 0.1$, $\omega = 6$ (corresponding to a time constant of 1/6 time scales or 1 min). Fig. 13.4 shows the simulated output trajectories within 100 time scales (10 hours). Apparently, compared to the open-loop system, the learned controllers have a better performance of disturbance rejection, with the outputs (degree of polymerization and reactor temperature) kept close to the setpoints and free from long-lasting significant deviations.

Of course, the use of a P controller in this case study is only for the illustration of how the dissipativity learning control method can give well-performing controllers with reasonable physical interpretations of the controller gains. P controllers may be not suitable and an offset will be expected if the setpoint of the process can vary with time. In such cases, one may consider designing a dissipativity-based PID controller, as illustrated in the next case study.

13.5 Case study: Dissipativity Learning Tracking Control of an Oscillatory Reactor

In this section we apply the proposed dissipativity learning control method to the tracking control of a CSTR [311] in which the catalyzed gas phase oxidization of ethylene takes place. There exist two side reactions to oxidize the primary product – ethylene oxide – and the reactant ethylene into carbon dioxide. It is known that appropriate periodic operation of this reactor can be economically more favorable than the optimal steady-state operation [57].

13.5.1 System description

The state-space model involves 4 states and 2 inputs:

$$\begin{aligned}
\dot{x}_1 &= (0.35 + u_1)(1 - x_1 x_4) \\
\dot{x}_2 &= (0.35 + u_1)(0.1 + u_2 - x_2 x_4) - A_1 \exp(C_1/x_4)(x_2 x_4)^{0.5} - A_2 \exp(C_2/x_4)(x_2 x_4)^{0.25} \\
\dot{x}_3 &= -(0.35 + u_1) x_3 x_4 + A_1 \exp(C_1/x_4)(x_2 x_4)^{0.5} - A_3 \exp(C_3/x_4)(x_3 x_4)^{0.5} \\
\dot{x}_4 &= x_1^{-1}[(0.35 + u_1)(1 - x_4) + B_1 \exp(C_1/x_4)(x_2 x_4)^{0.5} \\
&\quad + B_2 \exp(C_2/x_4)(x_2 x_4)^{0.25} + B_3 \exp(C_3/x_4)(x_3 x_4)^{0.5} - B_4(x_4 - 1 - d)]
\end{aligned} \tag{13.58}$$

The 4 states (x_1, x_2, x_3, x_4) are the dimensionless density, ethylene concentration, ethylene oxide concentration, and temperature, respectively. The gas phase flow rate and the ethylene concentration at the reactor inlet are used as control inputs. We assume that there is a disturbance d in the cooling water temperature. The nominal steady states under zero inputs and disturbances are given in Table 13.4. The reactor temperature is taken as the output ($y = x_4 - x_4^{\text{ss}}$). The inputs u_1 and u_2, disturbance d, and output y are then scaled by 0.35, 0.10, 0.0001 and 0.0001, respectively.

[57] identified for $d = 0$ the optimal operation under control inputs in sinusoidal and piecewise constant forms, whose concrete forms are omitted here for brevity and the resulting trajectories are illustrated in Fig. 13.5. In this section we assume that these two sets of trajectories are available without model knowledge, and examine whether the proposed dissipativity learning control method can efficiently track them. For tracking control where the desired inputs and the output oscillate over considerable ranges, it may not be sufficient to use P controllers. Hence we consider dissipativity learning

Table 13.4: Parameters and nominal steady state for the oscillatory reactor system

Par.	Value	Par.	Value
A_1	92.80	B_1	7.32
A_2	12.66	B_2	10.39
A_3	2412.71	B_3	2170.5
C_1	-8.13	B_4	7.02
C_2	-7.12	C_3	-11.07
State	Value	State	Value
x_1^{ss}	0.99889	x_2^{ss}	0.06494
x_3^{ss}	0.00949	x_4^{ss}	1.00111

with expanded outputs with integral and derivative of y, and controller design in the formulation of (13.47).

13.5.2 Data generation and dissipativity learning

Independent trajectories ($P = 2500$) are sampled in a similar way as in Section 13.4. For each sample, zero-mean stochastic processes bounded in $[-1, 1]$ with a modified Orstein-Uhlenbeck form:

$$dU = -\frac{\omega U}{1 - U^4} dU + \sigma dW \tag{13.59}$$

($\omega = 1$, $\sigma = 1$) are used to generate $\bar{u}(t)$ and $u(t)$, and a standard Wiener process is used to generate $d(t)$ within a time scale to drive the plant from the nominal steady state, yielding the trajectory of $(\bar{y}_{\mathrm{P}}, \bar{y}_{\mathrm{I}}, \bar{y}_{\mathrm{D}})$, $(y_{\mathrm{P}}, y_{\mathrm{I}}, y_{\mathrm{D}})$ and $(\tilde{y}_{\mathrm{P}}, \tilde{y}_{\mathrm{I}}, \tilde{y}_{\mathrm{D}})$ by subtraction. Since the output derivatives are used in dissipativity learning, the output signals need to be smooth enough. For this purpose, Savitzky-Golay filters [361] of order 3 with a frame length of 0.5 time scale are applied to the disturbance and inputs. 100 of these trajectories are shown in Fig. 13.6. From the trajectories, the dual dissipativity parameters Γ^p are calculated.

The distributions of independent components η_j, $j = 1, \ldots, J$ under $J = 21$ are shown in Fig. 13.7. It can be expected that compared with the polymerization reactor example, the Γ^p data samples from the oscillatory reactor can be better described by the bi-exponential distributions of its independent components. This is due to the higher extent of dispersion of independent components from their distribution centers, which may be explained by the fact that the data are sampled from a wide range of trajectories to track rather than a single steady state to regulate at. The approach to determine

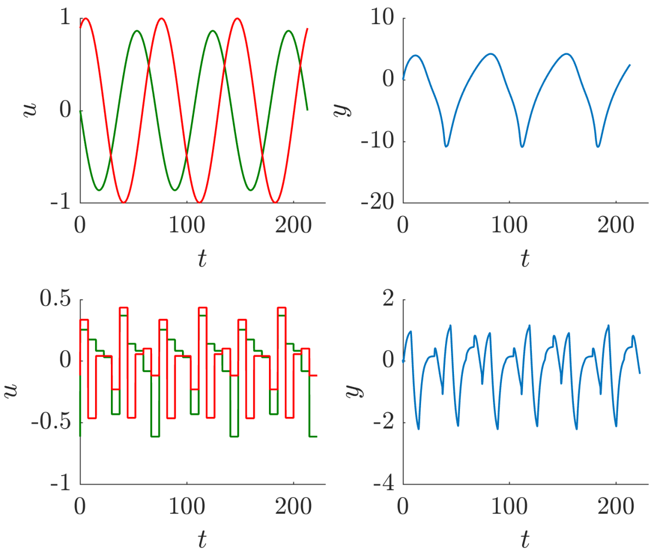


Figure 13.5: Desired input and output trajectories the oscillatory reactor operated periodically under sinusoidal (upper subplots) and piecewise constant inputs (lower subplots). Red and green curves correspond to u_1 and u_2, respectively.

the appropriate range of J is similar to the previous case study, i.e., we perform the dissipativity learning step under a series of different values of J and check whether solving (13.53) returns a negative feasible solution. We then calculate the covariance matrix (13.54) to choose the number of eigenvalues greater than 1% of the leading eigenvalue, which suggests that $J = 5$.

13.5.3 Controller design and simulation

Finally we solve the L_2-optimal dissipativity learning PID control problem (13.47) under $J = 5$ and confidence levels Δ corresponding to 85%, 90%, 95% and 99% of the samples. For example, under the confidence level of 95% of the samples, the learned PID controller is

$$\begin{aligned} \tilde{u}_1 &= 2.4407\tilde{y}_{\mathrm{P}} + 0.2008\tilde{y}_{\mathrm{I}} + 0.1410\tilde{y}_{\mathrm{D}}, \\ \tilde{u}_2 &= -4.7141\tilde{y}_{\mathrm{P}} - 1.2145\tilde{y}_{\mathrm{I}} - 0.2655\tilde{y}_{\mathrm{D}}. \end{aligned} \tag{13.60}$$

The negative gains for the second input reflect the cooling effect on the reactor temperature of the cooling water. The positive gains for the second input result from the fact that increasing (decreasing) the gas phase flow rate decreases (increases) the residence time, thus decreasing (increasing) the reactor temperature with a lower (higher) extent

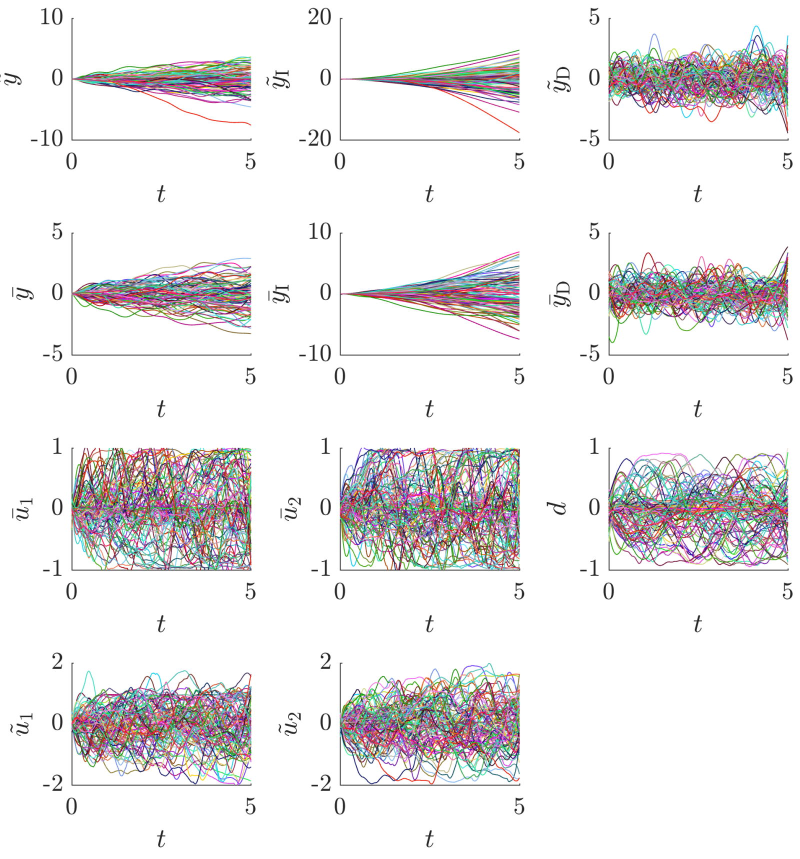


Figure 13.6: Trajectory samples of the oscillatory reactor system.

of exothermic reactions.

A stochastic Orstein-Uhlenbeck process is generated as the disturbance signal. The simulated trajectories under the obtained 4 controllers and the open-loop trajectories are shown in Fig. 13.8. It becomes apparent that under the learned controllers, the disturbance is well attenuated compared to the open-loop system, where the disturbance results in large fluctuations of the output.

13.6 Conclusions

Dissipativity, as an important characterization of the input–output response of dynamic systems, can be leveraged in the setting of input–output data-driven control. Specifically, with the trajectories sampled from the dynamics, we can learn the range of the

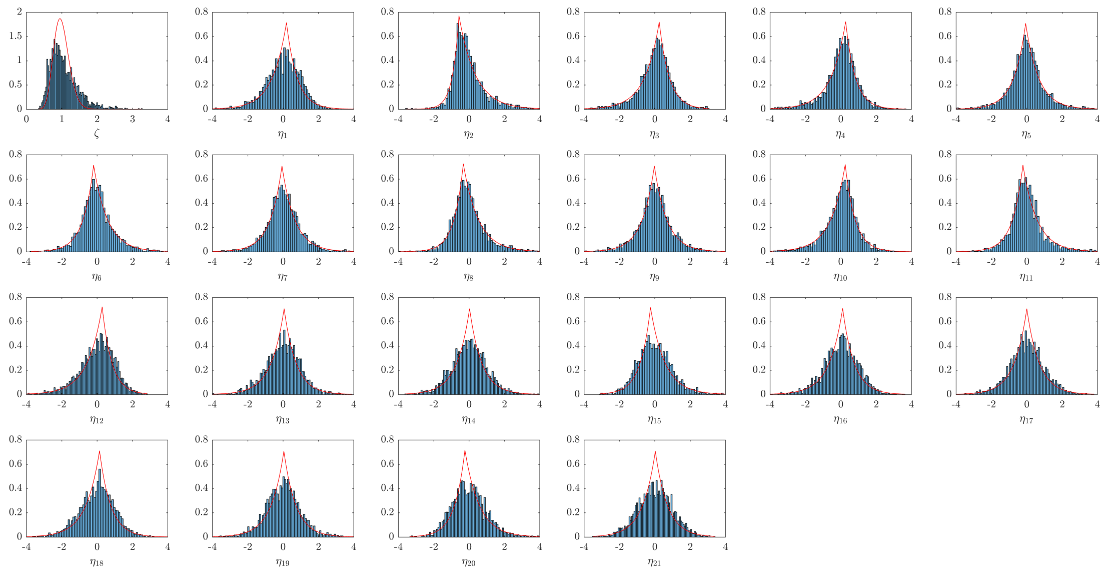


Figure 13.7: Distribution of the independent components and the lumped random variable ζ when $J = 21$ for the oscillatory reactor.

parameters in the supply rate function (dissipativity set), based on which a dissipative controller can be synthesized to shape the closed-loop stability. In this work, we first pointed that such a dissipativity learning method is applicable to both regulating and tracking control, and then proposed a method of learning dissipativity and synthesizing dissipativity-based controllers from trajectory data based on independent component analysis and distribution estimation. Two chemical reactor systems are used for detailed case studies that demonstrated the efficacy of our proposed method. The P and PID controllers designed with the proposed method achieve satisfactory performance. In addition, we have shown that the learned dissipativity parameters are of good physical interpretability, based on which the signs of the optimal controller gains are found to be in accordance with the physical relations among the process variables.

Although tested with only a few case studies so far, this framework is promising for wider application of input–output data-driven control on process systems, including those governed by partial differential and differential-algebraic equations, and large-scale processes. We note that as a machine learning-based approach, the performance of dissipativity learning control is essentially dependent on the accuracy of the dissipativity learning result. To better guarantee the learning performance, there are two open problems yet to be explored in future research, namely how to optimally generate or

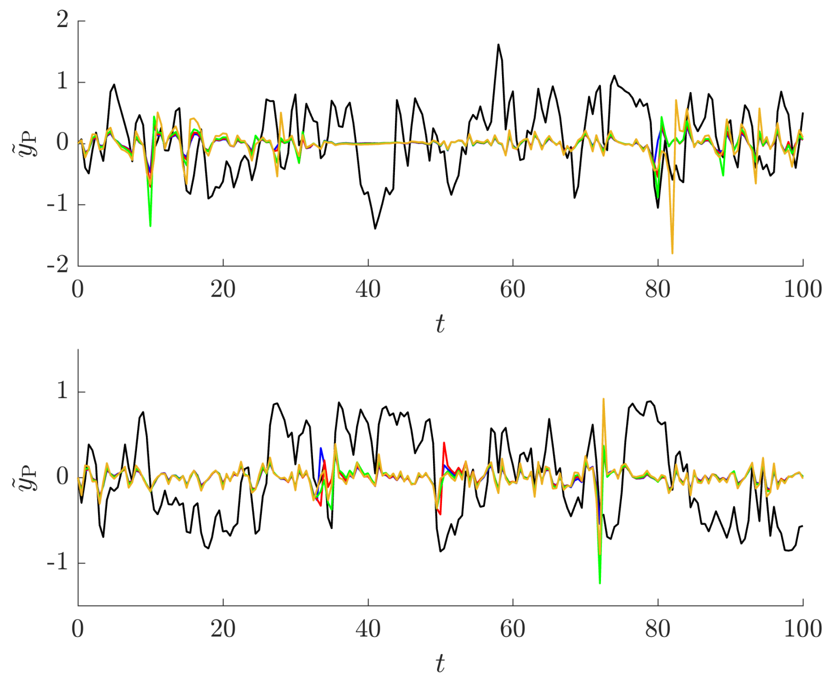


Figure 13.8: Simulated signals of the output errors of tracking the periodic trajectories. The upper and lower subplots correspond to the trajectories in Fig. 13.5 with sinusoidal and piecewise constant signals, respectively. The blue, red, green, and yellow lines correspond to dissipativity learning controllers obtained under $J = 5$ and confidence levels 85%, 90%, 95% and 99%, respectively. The black lines correspond to the open-loop system ($u = 0$).

select data samples, and how to perform learning when data are less satisfactory, e.g., when sufficient open-loop trajectories are not available.

Chapter 14
Dissipativity Learning Control (DLC): Theoretical Foundations of Input–Output Data-Driven Process Control

14.1 Introduction

The increasing role of machine learning and data-driven techniques in modeling, optimization, control and monitoring has become an important feature of process systems engineering in the recent years, based on the belief that process data are highly informative of the underlying relations or interactions in the systems of interest, thus assisting or even replacing the traditional model-based paradigms of decision making [214, 338, 432]. For control purposes, due to the potential of easing the derivation and maintenance of dynamic models, a wide range of data-driven control algorithms have been proposed [163, 301].

Based on the different roles of data in these approaches, we can classify them into two categories: data-driven *model-based* ones and data-driven *model-free* ones. In data-driven model-based control, data is mainly used for the identification of a dynamic model. This ranges from the classical approaches of transfer function or linear state-space model construction [238], parameter estimation in nonlinear models [106], neural nets [35] to more recent deep learning [240] and Koopman operator approaches [449, 283]. Apart from the model itself, data may also provide specific auxiliary information such as model-plant mismatch or uncertainties, either through passive correction in classical MPC [121] or active learning in stochastic control methods [261]. In these model-based approaches, one still encounters the difficulties of establishing models, which should be sufficiently complex to accurately describe the system dynamics while sufficiently simple to synthesize controllers.

In contrast, in data-driven model-free control, one directly seeks to learn from data some *essential control-relevant information*, which can be *much simpler than the full dynamic model* but has a *more direct relation to the resulting control performance*. This idea dates back to the traditional PID tuning approaches based on the time and frequency constants on response curves [13] for simple systems. More generic frameworks such as iterative feedback tuning (IFT) [159] and virtual reference feedback tuning (VRFT) [50] have been developed for linear systems. Preliminary explorations on the theoretical foundations and potential of data-driven model-free control approaches based on the behavior theory of linear systems have been made [251, 79, 429]. In recent years, approximate dynamic programming (ADP) and reinforcement learning (RL) have gained more popularity [412, 376, 391]. In these approaches, the optimal control policy and cost as state-dependent functions are considered as the essential control-relevant information to be learned. Rooted in the optimal control theory, ADP and RL approaches can often provide stability guarantees for nonlinear systems, and appear to be promising for model-free process control.

Despite the potential advantages of model-free control approaches, their applications in nonlinear processes are usually limited to small-scale systems with relatively simple dynamics. We note that this is mainly due to the facts that (i) chemical processes are intrinsically nonlinear, whose control performance is restrictive if linear controller synthesis approaches are used, and that (ii) the mechanisms underlying the dynamics is typically complex with unobservable or unknown states, which makes function approximations on the state space highly challenging. For these reasons, in our previous works [414, 411] we have developed an *input–output data-driven model-free control* framework, named *dissipativity learning control* (DLC).

The DLC framework is built upon the dissipative theory in classical nonlinear control [323, 239], where dissipativity is used as a characterization of the input–output behavior of systems and as a basis for controller synthesis, and also motivated by the works on dissipativity-based control of chemical processes [4, 145, 357, 157], where the dissipativity property is obtained through a rigorous derivation from the irreversible thermodynamic model. Different from these works on model-based dissipative control, in DLC, the dissipativity property is considered as the essential control-relevant information to be *learned from data* in the form of input–output trajectories under excitations. Such a dissipativity learning approach hence avoids the restrictions of thermodynamic analysis, and establishes itself as a generic data-driven model-free control framework for process systems. We note that recently in parallel to our works, the determination and learning of dissipativity (or related properties) from the trajectories data of persistently excited linear time-invariant (LTI) systems have been discussed [253, 352].

Although the dissipative approach is well established in nonlinear model-based control theory, the theoretical foundation of the learning of dissipativity from process data and its impact on the resulting control, which is necessary for formally guaranteeing the performance of the DLC framework, is still lacking. In this work, we aim at providing such a theoretical support for DLC. Specifically, the following points are made:

- If free of statistical errors, the DLC yields the optimal dissipative output-feedback control law that is in a certain nearly L^2-optimal sense defined on a certain neighborhood of the origin;
- The errors resulting from data sampling and statistical inference of the effective dual dissipativity set cause an error in the dissipativity learning result in terms of an upper bound of the L^2-gain from the exogenous disturbances to the inputs and outputs;
- Under small errors in dissipativity learning, the perturbation on the resulting upper bound of L^2-gain is also small, so that nearly L^2-optimal control performance is still achievable.

The remainder is organized as follows. We first give the control-theoretic description of the dissipativity-based control and its related concepts, providing the control-theoretic foundation in Section 14.2. The statistical aspects of dissipativity learning, including the sampling of trajectories and inference of the effective dual dissipativity set, are discussed in Section 14.3. Thus we reach at a standardized DLC algorithm with a formal guarantee of control performance, formulated in Section 14.4. We examine such a standardized DLC framework and compare it with linear model identification-based optimal control through a case study on a two-phase reactor in Section 14.5. Conclusions are given in Section 14.6.

14.2 Control-Theoretic Foundations

14.2.1 Dissipativity

We consider nonlinear systems in the form of

$$\dot{x}(t) = f(x(t), u(t), d(t)), \;\; y(t) = h(x(t)) \tag{14.1}$$

where $x(t) \in \mathbb{R}^{n_x}$ is the vector of states, $y(t) \in \mathbb{R}^{n_y}$ is the outputs, $u(t) \in \mathbb{R}^{n_u}$ is the vector of control inputs (manipulated variables), $d(t) \in \mathbb{R}^{n_d}$ is the exogenous disturbances. The vector of inputs $v(t) = (u(t), d(t)) \in \mathbb{R}^{n_v}$ is stacked from manipulated variables and disturbances. Functions f and h are assumed to be Lipschitz continuous, satisfying

$f(0,0,0) = 0$ and $h(0) = 0$, i.e., the origin is an equilibrium point of (14.1) giving zero outputs under zero inputs. An output feedback control law is considered as a Lipschitz continuous function $\kappa : \mathbb{R}^{n_y} \to \mathbb{R}^{n_u}$, leading to a closed-loop system:

$$\dot{x}(t) = f(x(t), \kappa(h(x(t))), d(t)). \tag{14.2}$$

For the design of κ, it is desirable that the closed-loop system (14.2) is asymptotically attracted to the origin from some neighborhood of the origin in the absence of disturbances, or is subject to limited impact of disturbances. The concept of dissipativity, originally introduced in [448] and later developed in [154, 278, 155], provides a global description of a fundamental constraint on the input–output behavior of dynamical systems. We note that since a globally dissipative property is difficult to obtain, here we consider dissipativity in a more restricted but practical context.

We first define the set of admissible input signals.

Definition 14.1 (admissible inputs)**.** *The set of admissible input signals on* $[0,t]$ *is the collection of real* n_v*-dimensional vector-valued continuous functions on* $[0,t]$ *that has an* L^2*-norm not exceeding* 1*, and for each of its component, if expressed as cosine series, the* L^2*-norm of all the high-frequency terms with wave number over* $N_\mathrm{f} \in \mathbb{N}$ *can not exceed a proportion of a small positive number* ϵ_f*, i.e.,*

$$\begin{aligned}\mathcal{V}_{N_\mathrm{f},\epsilon_\mathrm{f}}(t) = \Bigg\{ v : [0,t] \to \mathbb{R}^{n_v} \Bigg| v^j(\tau) = \frac{a_0^j}{\sqrt{2}} + \sum_{i=1}^{\infty} a_i^j \cos\frac{i\pi\tau}{t} \\ \sum_{i=0}^{\infty} (a_i^j)^2 \le 1, \sum_{i=N_\mathrm{f}+1}^{\infty} (a_i^j)^2 \le \epsilon_\mathrm{f} \sum_{i=1}^{N_\mathrm{f}} (a_i^j)^2, j = 1, \dots, n_v \Bigg\}\end{aligned} \tag{14.3}$$

Such defined admissible input signal set excludes very large or highly oscillatory signals. By using this definition, we are implicitly assuming that for the controller design of the system (14.1), its behavior under excessively large or oscillatory input signals does not contain any information that is of interest, nor does the controller necessarily result in such signals. We also define the domain of states that are reachable from the origin within time t.

Definition 14.2 (reachable domain)**.** *The reachable domain at time* t *(under admissible input signals) is the endpoint of all trajectories of system* (14.1) *on* $[0,t]$ *under input* *signals whose restriction on* $[0,\tau]$ *is admissible for all* $\tau \in [0,t]$*:*

$$\mathcal{D}_{N_{\mathrm{f}},\epsilon_{\mathrm{f}}}(t) = \left\{ x(t) \middle| \dot{x}(\tau) = f(x(\tau), v(\tau)), \tau \in [0,t], v|_{[0,\tau]} \in \mathcal{V}_{N_{\mathrm{f}},\epsilon_{\mathrm{f}}}(\tau), \tau \in [0,t], x(0) = 0 \right\}. \tag{14.4}$$

For brevity we will usually omit the $N_{\mathrm{f}}, \epsilon_{\mathrm{f}}$ in the subscript. Clearly, if $x \in \mathcal{D}(t_1)$ for some $t_1 \geq 0$, then there exists v on $[0,t_1]$ driving the states from the origin to x; then for any $t_2 > t_1$, the signal v with a time delay of $t_2 - t_1$ drives the states from 0 to x in $[0,t_2]$, and hence $x \in \mathcal{D}(t_2)$. This implies $\mathcal{D}(t_1) \subseteq \mathcal{D}(t_2)$ whenever $0 \leq t_1 \leq t_2$, and hence the reachable domain at time t is equivalent to the reachable domain *within* time t. With these definitions, we reconstruct the dissipative control theory through an adaptation of the approach of [154].

Definition 14.3 (Hill-Moylan inequality)**.** *The system* (14.1) *is said to satisfy Hill-Moylan inequality on* $\mathcal{D}(T)$ *for some* $T > 0$*, if for any* $t \in [0,T]$ *and* $v : [0,t] \to \mathbb{R}^{n_v}$ *such that* $v|_{[0,\tau]} \in \mathcal{V}(\tau)$ *for all* $\tau \in [0,t]$*, the resulting trajectory on* $[0,t]$ *satisfies*

$$\int_0^t s(y(\tau), v(\tau)) d\tau \geq 0 \tag{14.5}$$

for some continuous function $s : \mathbb{R}^{n_y} \times \mathbb{R}^{n_v} \to \mathbb{R}$*. Such a function* s *is called the supply rate function.*

Now we establish the dissipativity property of the system. For this we define the storage function and the concept of early reachable domain.

Definition 14.4 (storage function)**.** *If the system* (14.1) *satisfies the Hill-Moylan inequality on* $\mathcal{D}(T)$ *for some* $T > 0$*, then the following function* $V(x)$ *defined on* $\mathcal{D}(T)$*, which is positive semidefinite with* $V(0) = 0$*, is called the storage function:*

$$V(x) = \min_{\substack{v|_{[0,\tau]} \in \mathcal{V}(\tau) \quad \tau \in [0,t] \\ x(0)=0, \ x(t)=x}} \int_0^t s(y(\tau), v(\tau)) d\tau. \tag{14.6}$$

For convenience, we refer to any such minimizing input signal from the origin to x *as an effective reaching signal.*

The minimum in the above definition is well defined due to the Lipschitz continuity of the dynamics, continuity of the storage function, and completeness of the admissible input signal set. Obviously, the minimum can always be found for an input signal with $t = T$ (by using the time-delay argument). However, for studying the dissipativity property, we should consider only the states for which an effective reaching signal exists for some $t < T$, so that this signal may be extended after t.

Definition 14.5 (early reaching domain). *The early reaching domain $\mathring{\mathcal{D}}(T)$ is the subset of $\mathcal{D}(T)$ with states x such that an effective reaching signal for x defined in* (14.6) *exists on some time t strictly less than T. We call any such effective reaching signal as an early reaching signal.*

Now we may follow a similar approach to [154] to establish the dissipativity property.

Lemma 14.1 (Hill-Moylan Lemma). *Suppose that the Hill-Moylan inequality* (14.5) *holds. Then for any $x_1 \in \mathring{\mathcal{D}}(T)$, suppose that it has early reaching signal that is defined on $[0, t_1]$ for some $t_1 < T$, can be extended to some $t_2 \in [t_1, T]$ without violating the admissibility, and drives the states to x_2. Then*

$$V(x_2) - V(x_1) \leq \int_{t_1}^{t_2} s(y(t), v(t))dt. \tag{14.7}$$

We refer to such a property as the dissipativity of storage function V with respect to supply rate s on $\mathcal{D}(T)$, and (14.7) *as the dissipative inequality.*

Proof. According to the definition of the storage function (14.6),

$$V(x_2) - V(x_1) = \min_{\substack{v|_{[0,\tau]} \in \mathcal{V}(\tau) \quad \tau \in [0,t] \\ x(0)=0, \ x(t)=x_2}} \int_0^t s d\tau - \int_0^{t_1} s d\tau. \tag{14.8}$$

The lemma is proved by relaxing the minimum term to this hypothetic trajectory on $[0, t_1]$ from the origin to x_1 continued on $[t_1, t_2]$ from x_1 to x_2. $\square$

Therefore, to acquire the knowledge about the dissipativity property of the system (14.1), one needs to solve the following problem, which we will further discuss later.

Problem 14.1 (determination of the supply rate function). *Given hyperparameters N_f, ϵ_f and T, find (a range of) continuous functions s satisfying the Hill-Moylan inequality.*

14.2.2 Dissipative controller synthesis

Suppose that we now know a supply rate function s (or a range of supply rate functions) of the system. According to the Hill-Moylan Lemma, the system is dissipative with respect to supply rate s for some storage function V. We now consider how the dissipativity property leads to results in controller synthesis to guarantee closed-loop stability. Key to the desirable stability property is to shape the closed-loop supply rate function with bounded nonconcavity:

$$s(y, \kappa(y), d) \leq \beta \|d\|^2 - \|\kappa(y)\|^2 - \|y\|^2, \tag{14.9}$$

for some $\beta > 0$, so that the dissipative inequality (14.7) constrains the L^2-gain from d to the performance outputs $z = (y, u)$ not to exceed $\beta^{1/2}$.

To guarantee the dissipative inequality throughout the time, we need to prevent the concepts given in the previous subsection from becoming ill-defined.

Definition 14.6 (effective reachable domain). *The effective reachable domain $\check{\mathcal{D}}^\kappa(T) = \check{\mathcal{D}}^\kappa_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$ of the output feedback control law κ is such a subset of $\mathring{\mathcal{D}}$ satisfying the following condition. For any point $x_0 \in \mathring{\mathcal{D}}(T)$ and for any exogenous disturbance signal that is admissible on $[0, t]$ for all $t \geq 0$, the closed-loop system starting from x_0 at time 0 under control $u = \kappa(y) = \kappa(h(x))$ remains in $\check{\mathcal{D}}$, and the input signal retains the admissibility of the effective reaching signal for all $t \geq 0$.*

It appears to be difficult to characterize the effective reachable domain and prove whether it is a connected open set. For simplicity, we make the following assumption.

Assumption 14.1. *Assume that when the output-feedback law κ is chosen within a predefined range of interest $\mathcal{K}$, there is a neighborhood of the origin $\check{\mathcal{D}}$ that is in the intersection of all the effective reachable domains, i.e., $\check{\mathcal{D}}(T) = \cap_{\kappa \in \mathcal{K}} \check{\mathcal{D}}^\kappa(T)$.*

With a slight abuse of terminology, we still call this $\check{\mathcal{D}}(T)$ the effective reachable domain. Clearly, it is the domain on which the dissipative inequality (14.7) holds recursively. Moreover, since we only discuss the dissipativity on the effective reachable domain $\check{\mathcal{D}}(T)$ under controller κ, it suffices to have a relaxed definition of supply rate, called effective supply rate function. Of course, since $\check{\mathcal{D}}(T)$ is difficult to know, so does the effective supply rate.

Definition 14.7 (effective supply rate). *An effective supply rate function is a continuous function s such that Hill-Moylan inequality holds for some effective input signals on $[0, t]$ for some $t < T$ that is admissible on any $[0, \tau]$, $\tau \in [0, t]$ for some state in $\check{\mathcal{D}}(T)$. We refer to the set of supply rate functions as $\mathcal{S}$ and set of effective supply rate functions as $\check{\mathcal{S}}$. Then $\check{\mathcal{S}} \supseteq \mathcal{S}$.*

Now we establish the guarantee of L^2-stability according to the following proposition, for which the proof is simple by combining (14.7) and (14.9).

Lemma 14.2 (stability of dissipative control). *Suppose that there exist an output-feedback control law $\kappa \in \mathcal{K}$ and an effective supply rate function $s \in \check{\mathcal{S}}$ satisfying* (14.9) *for some $\beta > 0$, then starting from any $x(0) \in \check{\mathcal{D}}(T)$, for any $t \geq 0$, we have a nonnegative constant C such that*

$$\int_0^t \left(\|y\|^2 + \|u\|^2\right) d\tau \leq \beta \int_0^t \|d\|^2 d\tau + C \tag{14.10}$$

always holds, and hence the exogenous disturbance has a L^2-gain to $z = (y, u)$ not exceeding $\beta^{1/2}$. In other words, the closed-loop system is L^2-stable.

Hence the controller synthesis problem based on dissipativity is formally stated as the following problem.

Problem 14.2 (determination of the control law). *Given a (or a range of) supply rate function s with respect to which the system is dissipative on $\mathcal{D}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$, find an output feedback control law κ such that $s(y, \kappa(y), d) \leq -\|y\|^2 - \|\kappa(y)\|^2 + \beta\|d\|^2$ for some (or the smallest) $\beta > 0$.*

For the computational tractability of the controller synthesis, we shall consider supply rate functions in quadratic forms:

$$s(y, u, d) = \begin{bmatrix} y^\top & u^\top & d^\top \end{bmatrix} \Pi \begin{bmatrix} y \\ u \\ d \end{bmatrix} \tag{14.11}$$

and hence the supply rate function $s(y, u, d)$ is represented by a symmetric matrix $\Pi \in \mathbb{R}^{(n_y+n_v)\times(n_y+n_v)}$. We also consider controllers in linear form $\kappa(y) = Ky$. This ends up at a semidefinite programming problem of finding a feasible or optimal solution of (K, β) satisfying the following matrix inequality

$$\begin{bmatrix} I & K^\top & 0 \\ 0 & 0 & I \end{bmatrix} \left(\Pi + \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & -\beta I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ K & 0 \\ 0 & I \end{bmatrix} \preceq 0. \tag{14.12}$$

Remark 14.1 (PID control). *Although we have represented the controller in the form of* *proportional feedback, it is worth noting that the semidefinite programming formulation can be extended to more general linear controller forms such as PID controllers, by simply augmenting the output variables with its derivatives and the state variables with an integrator. If the set of supply rate functions is accurately known and the optimal controller gain is optimized over all supply rate functions, then we obtain theoretically the optimal dissipative P/PI/PID controller in the L^2-sense within $\mathcal{K}$.*

Remark 14.2 (tracking control)**.** *For tracking control tasks where the goal is to drive the system towards a reference trajectory rather than to the origin, it suffices to split the process variables (v, x, y) into the corresponding reference variables $(\bar{v}, \bar{x}, \bar{y})$ (whose trajectory is specified a priori) and deviation variables $(\tilde{v}, \tilde{x}, \tilde{y})$ (so that the goal is to drive the deviations to zero), and correspondingly redefine the reachable domain, storage function, and supply rate functions based on the origin of deviation variables. This was discussed in our previous work [411].*

14.2.3 Dissipativity and dual dissipativity sets

Suppose that the model of the system (14.1) is unknown. In order to find a (or a range of) supply rate function parameterized by a quadratic form (14.11) in a data-driven setting, we first note that the Hill-Moylan inequality (14.5) can be rewritten as

$$\langle \Pi, \int_0^t \begin{bmatrix} y \\ u \\ d \end{bmatrix} \begin{bmatrix} y^\top & u^\top & d^\top \end{bmatrix} d\tau \rangle \geq 0, \tag{14.13}$$

where the inner product $\langle M_1, M_2 \rangle$ for any two symmetric matrices M_1 and M_2 is defined as trace$(M_1 M_2)$, with $\langle M, M \rangle = \|M\|^2$ being the Frobenius norm. We note that Π as a representation of the supply function is a property of the system, and the integral above is a property of the excitation input signal. Hence we introduce the following definitions in dual.

Definition 14.8 (dissipativity parameters and sets)**.** *The matrix* Π *is called the (matrix of) dissipativity parameters. The set of dissipativity parameters such that the corresponding supply rate function* (14.11) *is in $\mathcal{S}$ is called the dissipativity set (still denoted as $\mathcal{S}$). The dissipativity parameter set* Π *is said to be effective if the corresponding supply rate is effective. The set of effective dissipativity parameters is called the effective dissipativity set (still denoted as $\check{\mathcal{S}}$).*

Definition 14.9 (dual dissipativity parameters and sets). *The dual dissipativity parameter of an excitation input signal on* $[0, t]$ *that is admissible on all* $[0, \tau]$ *for* $\tau \in [0, t]$ *is defined based on the resulting excited trajectory as*

$$\Gamma = \int_0^t \begin{bmatrix} y \\ u \\ d \end{bmatrix} \begin{bmatrix} y^\top & u^\top & d^\top \end{bmatrix} d\tau. \tag{14.14}$$

The set of dual dissipativity parameters is called the dual dissipativity set (denoted as $\mathcal{G}$*), and the set of dual dissipativity parameters of effective input signals is called the effective dual dissipativity set (denoted as* $\check{\mathcal{G}}$*).*

It directly follows from these definitions that $\mathcal{S} \subseteq \check{\mathcal{S}}$, $\mathcal{G} \supseteq \check{\mathcal{G}}$, and $\mathcal{S}$ and $\check{\mathcal{S}}$ are the dual cones of $\mathcal{G}$ and $\check{\mathcal{G}}$, respectively, i.e.,

$$\begin{aligned} \mathcal{S} = \mathcal{G}^* = \{\Pi | \langle \Pi, \Gamma \rangle \geq 0, \forall \Gamma \in \mathcal{G}\} \\ \check{\mathcal{S}} = \check{\mathcal{G}}^* = \{\Pi | \langle \Pi, \Gamma \rangle \geq 0, \forall \Gamma \in \check{\mathcal{G}}\}. \end{aligned} \tag{14.15}$$

The dissipativity sets and dual dissipativity sets defined above are also hyper-parametrized by N_f, ϵ_f and T. These symbols are omitted for brevity. For quadratic supply rate function and linear feedback laws, we can restate Lemma 14.2 as

Lemma 14.3 (stability of dissipative control). *Suppose that there exist* $\Pi \in \check{\mathcal{G}}^*$, $K \in \mathcal{K}$, *and* $\beta > 0$ *satisfying* (14.12). *Then the closed-loop system is* L^2*-stable in* $\check{\mathcal{D}}$.

Therefore, the task of obtaining a dissipative controller for system (14.1), involving the determination of supply rate function (Problem 14.1) and the output-feedback control (Problem 14.2), is stated as follows.

Problem 14.3 (dissipativity-based control). *Given hyperparameters* N_f, ϵ_f *and* T, *obtain* $\check{\mathcal{G}}$ *and its dual cone* $\check{\mathcal{S}} = \check{\mathcal{G}}^*$, *and solve for a control law* K *satisfying* (14.12) *with some* $\Pi \in \check{\mathcal{S}}$ *and* $\beta > 0$.

14.3 Statistical Foundations

In a data-driven setting, the determination of the effective dual dissipativity set $\check{\mathcal{G}}$ and its dual cone for the system (14.1) is done on the basis of statistical inference. That is, an estimation of $\check{\mathcal{G}}$, denoted as $\hat{\mathcal{G}}$ and its dual cone $\hat{\mathcal{S}}$, as an estimation of $\check{\mathcal{S}}$, should be obtained from some data samples of the system. In this section, we discuss the effect of sampling and inference steps, and provide guidelines on how to collect samples and conduct inference.

14.3.1 Sampling of input excitations

In order to obtain estimations of the dissipativity and dual dissipativity sets, one needs to create samples of admissible input signals to excite the system (14.1) from the origin, collect the resulting output trajectories, and calculate the corresponding dual dissipativity parameters Γ. We note that each admissible input signal in $\mathcal{V}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$ is identical to an infinite-dimensional point (series) a in $\mathcal{A}^{n_v}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$, where

$$\mathcal{A}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T) = \left\{ (a_0, a_1, \dots) \middle| \sum_{i=1}^{\infty} a_i^2 \leq 1, \sum_{i=N_\mathrm{f}+1}^{\infty} a_i^2 \leq \epsilon_\mathrm{f} \sum_{i=1}^{N_\mathrm{f}} a_i^2 \right\}. \tag{14.16}$$

Sampling from $\mathcal{A}^{n_v}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$ as an infinite-dimensional set is apparently not tractable. Hence we consider sampling on its approximation as a finite-dimensional unit ball centered at the origin:

$$\hat{\mathcal{A}}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T) = \left\{ (a_0, a_1, \dots, a_{N_\mathrm{f}}) \middle| \sum_{i=1}^{N_\mathrm{f}} a_i^2 \leq 1 \right\} = \mathbb{B}_{N_\mathrm{f}}(0,1). \tag{14.17}$$

Every element of $\hat{\mathcal{A}}(T)$ is also in $\mathcal{A}(T)$ if suffixed with infinite number of zeros, and in this sense $\hat{\mathcal{A}}(T)$ is a subset of $\mathcal{A}(T)$. For each $a \in \mathcal{A}(T)$, one can obviously find a corresponding $\hat{a} \in \hat{\mathcal{A}}(T)$ such that $\|a - \hat{a}\|_2 \leq \epsilon_\mathrm{f}\|\hat{a}\|$ by grounding the numbers after the N_f-th to zero.

Therefore, if one samples in $\hat{\mathcal{A}}(T)$ in a sufficiently "proportionally dense" manner, such that for any $\hat{a} \in \hat{\mathcal{A}}(T)$, there exist a sample a' that is close enough to $\hat{a}$ with $\|a' - \hat{a}\| \leq \epsilon_{\hat{a}}\|a'\|$ for some small $\epsilon_{\hat{a}} > 0$, then for any $a \in \mathcal{A}(T)$, there exists a sample a' with $\|a' - a\| \leq \epsilon_a\|a'\|$ for some $\epsilon_a > 0$. We note that the ℓ^2-distance on the space of Fourier coefficients equals the L^2-distance of the input signals. Hence a proportionally dense sampling on $\hat{\mathcal{A}}(T)$ implies a proportionally dense sampling of admissible input signals, which also implies a proportionally dense sampling on the dual dissipativity set $\mathcal{G}$.

Lemma 14.4 (Denseness of sampling on the dual dissipativity set)**.** *If for any input* *signal $v \in \mathcal{A}(T)$, there exists a sample input signal $v^{(p)} \in \hat{\mathcal{A}}(T)$ (where p is the sample index) such that $\|v - v^{(p)}\|_{L^2([0,T])} \leq \epsilon_v \|v^{(p)}\|_{L^2([0,T])}$ for some small $\epsilon_v > 0$ (assuming $\epsilon_v < 1$), then there exist constants $C_0, C_1, C_2 > 0$ such that the corresponding dual dissipativity parameters Γ and $\Gamma^{(p)}$ satisfy*

$$\|\Gamma - \Gamma^{(p)}\| \leq C_0(1 + C_1 e^{C_2 T})\epsilon_v \mathrm{trace}(\Gamma^{(p)}). \tag{14.18}$$

Proof. Consider the incremental dynamics

$$\Delta\dot{x} = f(x + \Delta x, v + \Delta v) - f(x, v), \;\; \Delta y = h(x + \Delta x) - h(x). \tag{14.19}$$

Let the f and h have Lipschitz constants $L_{f,x}$, $L_{f,v}$ and $L_{h,x} > 0$, respectively. Then we obtain

$$\|\Delta\dot{x}\| \leq L_{f,x}\|\Delta x\| + L_{f,v}\|\Delta v\|, \;\; \|\Delta y\| \leq L_{h,x}\|\Delta x\|. \tag{14.20}$$

The Grönwall inequality implies that

$$\|\Delta y(t)\| \leq L_{h,x} L_{f,v} \int_0^t e^{L_{f,x}(t-\tau)} \|\Delta v(\tau)\| d\tau. \tag{14.21}$$

Applying Cauchy-Schwartz inequality, one easily obtains

$$\|\Delta y(t)\| \leq \frac{L_{h,x} L_{f,v}}{(2L_{f,x})^{1/2}} e^{L_{f,x} t} \left(\int_0^t \|\Delta v(\tau)\|^2 d\tau \right)^{1/2}. \tag{14.22}$$

Relaxing the right-hand side integral onto $[0, T]$ and calculating the integral of squares, we have

$$\|\Delta y\|_{L^2([0,T])} \leq \frac{L_{h,x} L_{f,v}}{2L_{f,x}} e^{L_{f,x} T} \|\Delta v\|_{L^2([0,T])}. \tag{14.23}$$

The resulting difference of the dual dissipativity parameters can be shown to be upper boundeded by

$$\|\Delta\Gamma\|^2 \leq \frac{1}{3}\left(\frac{L_{h,x}^2 L_{f,v}^2}{4L_{f,x}^2} e^{2L_{f,x}T} + 1\right)^2 (2\|v\|_{L^2}^2 + \|\Delta v\|_{L^2}^2)\|\Delta v\|_{L^2}^2, \tag{14.24}$$

and hence

$$\|\Delta\Gamma\| \leq \frac{2}{\sqrt{3}}\left(1 + \frac{L_{h,x}^2 L_{f,v}^2}{4L_{f,x}^2} e^{2L_{f,x}T}\right)\epsilon_v \|v\|_{L^2}. \tag{14.25}$$

This leads to the conclusion of the lemma with $C_0 = 2/\sqrt{3}$, $C_1 = L_{h,x}^2 L_{f,v}^2 / 4L_{f,x}^2$ and $C_2 = 2L_{f,x}$, since $\text{trace}(\Gamma^{(p)}) = \|\Delta v\|_{L^2}^2 + \|\Delta y\|_{L^2}^2$. □

If the sampling is dense on $\mathcal{G}$, then the remaining task is to select the samples that are useful to estimate the effective subset $\check{\mathcal{G}}$, and perform the inference of $\check{\mathcal{G}}$. However, due to the very reason that $\check{\mathcal{G}}$ is unknown a priori, we have to assume an oracle of sample selection.

Assumption 14.2 (Effective sample selection or generation oracle)**.** *Assume that there is an approach for the selection of data samples such that for any dual dissipativity parameter* Γ *in the effective dual dissipativity set* $\check{\mathcal{G}}$*, there is a selected sample* $\Gamma^{(p)}$ *such that*

$$\|\Gamma - \Gamma^{(p)}\| \leq \epsilon_\Gamma \text{trace}(\Gamma^{(p)}) \tag{14.26}$$

for some $\epsilon_\Gamma > 0$*. Alternatively we may assume that the samples are directly generated so that the above inequality is satisfied.*

Since the above oracle is only assumed for theoretical analysis and unavailable in reality, here we propose a simple heuristic rule of selecting effective samples. The underlying idea is that the effective reaching signal should tend to result in a smaller extent of oscillations in the inputs and outputs. If expressed as Fourier series, these input signals should have larger (in absolute values) low-frequency coefficients than the high-frequency coefficients. Hence we sort the components of the sample points in $\hat{\mathcal{A}}_{N_\mathrm{f},\epsilon_\mathrm{f}}(T)$ in the descending order of absolute values and use the sorted $a \in \hat{\mathcal{A}}(T)$ to replace the original one. We shall refer to this rule as the *sorting heuristics* later.

Remark 14.3 (negligibly small excitations)**.** *The condition of proportionally dense sampling is in fact not realizable with a finite sample set, since the excitation input signal may be arbitrarily small. We need to compromise such excessively small signals, which drive the states from the origin to a small neighborhood. As a result, the guaranteed* L^2*-stability degrades into a practical stability on the effective reaching domain excluding such a small neighborhood.*

14.3.2 Inference of effective dissipativity set

Now suppose that a dense sampling is performed on $\mathcal{G}$ and effective samples $\Gamma^{(p)}$, $p = 1, \ldots, P$ are selected such that the condition in Assumption 14.2 is satisfied. A statistical inference procedure is subsequently used to give an approximate description of the effective dual dissipativity set $\check{\mathcal{G}}$. It is easily verifiable that, if such a set $\hat{\mathcal{G}}$ estimated from samples may leave a sample out for a small enough distance, i.e., for any $\Gamma^{(p)}$ there exists a $\hat{\Gamma} \in \hat{\mathcal{G}}$ such that for a small $\epsilon_{\hat{\Gamma}} > 0$, $\|\Gamma^{(p)} - \hat{\Gamma}\| \leq \epsilon_{\hat{\Gamma}}\|\hat{\Gamma}\|$, then there exists a $0 < \epsilon_{\Gamma}^{+} \leq \epsilon_{\Gamma} + \epsilon_{\hat{\Gamma}} + \epsilon_{\Gamma}\epsilon_{\hat{\Gamma}}\sqrt{n_y + n_v} > 0$ such that $\|\Gamma^{(p)} - \hat{\Gamma}\| \leq \epsilon_{\Gamma}^{+}\|\hat{\Gamma}\|$. Hence for any $\Gamma \in \check{\mathcal{G}}$ there exists $\hat{\Gamma} \in \hat{\mathcal{G}}$ such that $\|\Gamma - \hat{\Gamma}\| \leq \epsilon\|\hat{\Gamma}\|$ for some $0 < \epsilon < \epsilon_{\Gamma}^{+} + \epsilon_{\hat{\Gamma}}$. Simply speaking, if the sampling is effective enough and the inference of $\check{\mathcal{G}}$ is close to the samples, then the estimated $\hat{\mathcal{G}}$ is close to the actual $\check{\mathcal{G}}$.

The following proposition establishes the impact of the accuracy of $\hat{\mathcal{G}}$ on the inference of its dual cone $\hat{\mathcal{S}}$.

Lemma 14.5 (Accuracy of effective dual dissipativity set)**.** *Suppose that for any* $\Gamma \in \check{\mathcal{G}}$ *there exists a* $\hat{\Gamma} \in \hat{\mathcal{G}}$ *such that* $\|\Gamma - \hat{\Gamma}\| \leq \epsilon\|\hat{\Gamma}\|$*, then*

$$\check{\mathcal{S}} \supseteq \bigcap_{\hat{\Gamma} \in \hat{\mathcal{G}}} \left\{ \hat{\Pi} \mid \langle \hat{\Pi}, \hat{\Gamma} \rangle \geq \epsilon \|\hat{\Pi}\| \cdot \|\hat{\Gamma}\| \right\}. \tag{14.27}$$

The right-hand side is a modified dual cone of $\hat{\mathrm{G}}$ *with the* 90° *angle specification strengthened to* $\arccos\epsilon$*. Therefore for any* $\hat{\Pi} \in \hat{\mathcal{S}}$*, there exists a* $\Pi \in \check{\mathcal{S}}$ *with* $\|\Pi - \hat{\Pi}\| \leq \delta\|\hat{\Pi}\|$ *with* $\delta \in [0, 2\sin\arcsin(\epsilon/2)]$.

Proof. According to the condition of the lemma, we have

$$\check{\mathcal{G}} \subseteq \{\hat{\Gamma} + \Delta\|\hat{\Gamma}\| \mid \hat{\Gamma} \in \hat{\mathcal{G}}, \|\Delta\| \leq \epsilon\}. \tag{14.28}$$

Hence $\check{\mathcal{S}}$ is a superset of the dual cone of the right-hand side of the "$\subseteq$" symbol. Since for each $\hat{\Gamma}$,

$$\{\hat{\Gamma} + \Delta\|\hat{\Gamma}\| \mid , \|\Delta\| \leq \epsilon\}^* = \{\hat{\Pi} | \langle \hat{\Pi}, \hat{\Gamma} \rangle + \|\hat{\Gamma}\| \min_{\|\Delta\| \leq \epsilon} \langle \hat{\Pi}, \Delta \rangle \geq 0\}, \tag{14.29}$$

where the minimization leads to a $-\epsilon\|\hat{\Pi}\| \cdot \|\hat{\Gamma}\|$ term, by taking the intersection over all $\hat{\Gamma} \in \hat{\mathcal{G}}$ we see the conclusion. □

For the estimation of the efficient reaching domain $\hat{\mathcal{G}}$, any anomaly detection, one-class classification, probability density estimation, or hull algorithm is in principle suitable. However, for computational tractability of controller synthesis, we need the estimate $\hat{\mathcal{G}}$ to have a simple convex form such as polyhedron, polyhedral cone, ellipsoid, or second-order cone. In our previous works [414, 411] we have used support vector machine and probability estimation with independent bi-exponential distributions for polyhedral sets. Here we provide the procedure for inferring $\hat{\mathcal{G}}$ as a second-order cone.

Note that any $\Gamma^{(p)}$ must be a semidefinite matrix, whose trace is positive except for the singular case of the zero-input trajectory, which we can assume does not exist in the sample. We first normalize all the samples of dual dissipativity parameters to trace 1:

$$\Gamma_+^{(p)} = \Gamma^{(p)}/\|\Gamma^{(p)}\|, \;\; p = 1, \dots, P. \tag{14.30}$$

Then a principal component analysis (PCA) algorithm is applied to the trace-1 samples. For this purpose, a basis $\{E_k\}_{k=1}^K$ for (n_y+n_v)-order symmetric matrices is chosen, which vectorizes the samples into

$$\gamma^{(p)} = [\langle E_k, \Gamma_+^{(p)} \rangle]_{k=1}^K. \tag{14.31}$$

By finding the sample average $\bar{\gamma}$ and the diagonal matrix of component-wise standard deviations $D = \mathrm{diag}(D_k, k = 1, \dots, K)$, the vectorized samples are whitened by translation and scaling:

$$\bar{\gamma} = \frac{1}{P}\sum_{p=1}^{P} \gamma^{(p)}, \;\; D_k = \frac{1}{P-1}\sum_{p=1}^{P}(\gamma_k^{(p)} - \bar{\gamma}_k)^2, \;\; \gamma_+^{(p)} = D^{-1}(\gamma^{(p)} - \bar{\gamma}). \tag{14.32}$$

Then the ellipsoidal range of $\gamma_+^{(p)}$, $p = 1, \dots, P$ is found by a singular value decomposition of the horizontally stacked matrix of samples,

$$\frac{1}{\sqrt{P-1}} \begin{bmatrix} \gamma_+^{(1)} & \dots & \gamma_+^{(P)} \end{bmatrix} = USV^\top, \tag{14.33}$$

where U and V are orthogonal matrices, and $S \in \mathbb{R}^{K \times P}$ is a diagonal matrix of singular values $s_i \geq 0$ (in descending order), i.e., if subscribing the column vectors of U and V and the singular values, then the right-hand side above is $\sum_{i=1}^{\min(K,P)} s_i u_i v_i^\top$. Typically sufficiently small singular values are neglected by choosing the number of principal values J according to a certain rule, then

$$\frac{1}{\sqrt{P-1}} \begin{bmatrix} \gamma_+^{(1)} & \dots & \gamma_+^{(P)} \end{bmatrix} \approx U_{1:J} S_{1:J} V_{1:J}^\top, \tag{14.34}$$

which implies that the sample covariance matrix is approximately $U_{1:J} S_{1:J}^2 U_{1:J}^\top$ (the subscript $1:J$ stands for columns 1 to J for U and V, and also rows 1 to J for

S). Suppose that the samples are normally distributed. Then the vectors of principle components

$$\eta^{(p)} = S_{1:J}^{-1} U_{1:J}^{\top} \gamma_{+}^{(p)} \in \mathbb{R}^J, \;\; p = 1, \ldots, P \tag{14.35}$$

should be samples from a standard normal distribution, for which the ℓ^2-norm is the Hotelling statistics, and a threshold hyperparameter $\Theta > 0$ can be chosen.

Through the PCA, an estimation $\hat{\mathcal{G}}$ is obtained as a confidence region in the form of

$$\begin{aligned} \hat{\mathcal{G}} &= \{\eta | \|\eta\| \leq \Theta\} = \{\gamma_+ | \gamma_+ = U_{1:J} S_{1:J} \eta, \|\eta\| \leq \Theta\} \\ &= \{\gamma = \bar{\gamma} + D U_{1:J} S_{1:J} \eta, \|\eta\| \leq \Theta\} \end{aligned} \tag{14.36}$$

Call the following matrix as the mixing matrix (of vectorized normalized dual dissipativity matrices from principal components):

$$G = D U_{1:J} S_{1:J}. \tag{14.37}$$

Then

$$\hat{\mathcal{G}} = \left\{ \Gamma \middle| \Gamma = r \sum_{k=1}^{K} \gamma_k E_k, \gamma = \bar{\gamma} + G\eta, r \geq 0, \|\eta\| \leq \Theta \right\}, \tag{14.38}$$

whose dual cone is expressed as

$$\hat{\mathcal{S}} = \left\{ \Pi \middle| \Pi = \sum_{k=1}^{K} \pi_k E_k, \langle \pi, \bar{\gamma} \rangle - \Theta \|G^{\top} \pi\| \geq 0 \right\}. \tag{14.39}$$

14.3.3 The effect of learning on control

Finally, we consider the impact of the statistical errors on the resulting control performance in terms of the upper bound on the L^2-gain. Call the third matrix on the left-hand side of the matrix inequality (14.12) as the loop-closing matrix, and denote it as H. The perturbation on the resulting L^2-gain is then considered by the modification of the matrix inequality.

Lemma 14.6 (Resulting control performance)**.** *Let*

$$H = \begin{bmatrix} I & 0 \\ K & 0 \\ 0 & I \end{bmatrix}. \tag{14.40}$$

Suppose that the matrix inequality

$$H^\top \left(\hat{\Pi} + \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & -\beta I \end{bmatrix} \right) H \preceq 0, \tag{14.41}$$

holds for some $\hat{\Pi} \in \hat{\mathcal{S}}$, and that there exists a $\Pi \in \check{\mathcal{S}}$ such that $\|\Pi - \hat{\Pi}\| \leq \delta \|\hat{\Pi}\|$. Then the square L^2-gain from exogenous d to $z = (y, u)$ is guaranteed to be upper bounded by $\beta + \delta_\beta$ on $\check{\mathcal{D}}$ with $\delta_\beta = (1+\beta)\delta\|\hat{\Pi}\|/(1 - \delta\|\hat{\Pi}\|)$, if $\beta\|\hat{\Pi}\| \leq 1$.

Proof. Since for any symmetric matrix the spectral radius can not exceed its Frobenius norm, $\|\Pi - \hat{\Pi}\| \leq \delta\|\hat{\Pi}\|$ implies that

$$\Pi - \hat{\Pi} - \begin{bmatrix} \delta\|\hat{\Pi}\| I & 0 & 0 \\ 0 & \delta\|\hat{\Pi}\| I & 0 \\ 0 & 0 & \delta\|\hat{\Pi}\| I \end{bmatrix} \preceq 0. \tag{14.42}$$

Hence

$$H^\top \left(\Pi - \hat{\Pi} - \begin{bmatrix} \delta\|\hat{\Pi}\| I & 0 & 0 \\ 0 & \delta\|\hat{\Pi}\| I & 0 \\ 0 & 0 & \delta\|\hat{\Pi}\| I \end{bmatrix} \right) H \preceq 0, \tag{14.43}$$

which implies the following inequality by adding to (14.41) and multiplying by $1/(1 - \delta\|\hat{\Pi}\|)$:

$$H^\top \left(\frac{1}{1 - \delta\|\hat{\Pi}\|} \Pi + \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & (\beta + \delta_\beta) I \end{bmatrix} \right) H \preceq 0. \tag{14.44}$$

Since $\check{\mathcal{S}}$ is a cone, when $\Pi \in \check{\mathcal{S}}$, $\frac{1}{1-\delta\|\hat{\Pi}\|}\Pi \in \check{\mathcal{S}}$. According to Lemma 14.2, the conclusion is proved. □

Due to the possibility that the adopted schemes of sample generation, selection and statistical inference may not be effective enough to cover a significant part of $\check{\mathcal{G}}$ (satisfying the condition of Lemma 14.5), the above-mentioned perturbed control performance is guaranteed only on a subset of $\check{\mathcal{D}}$ that is positive-invariant under control laws in $\mathcal{K}$ without violating admissibility of input signals. Finally we assume that this true domain of attraction Ω is still a neighborhood of the origin with a satisfactory size.

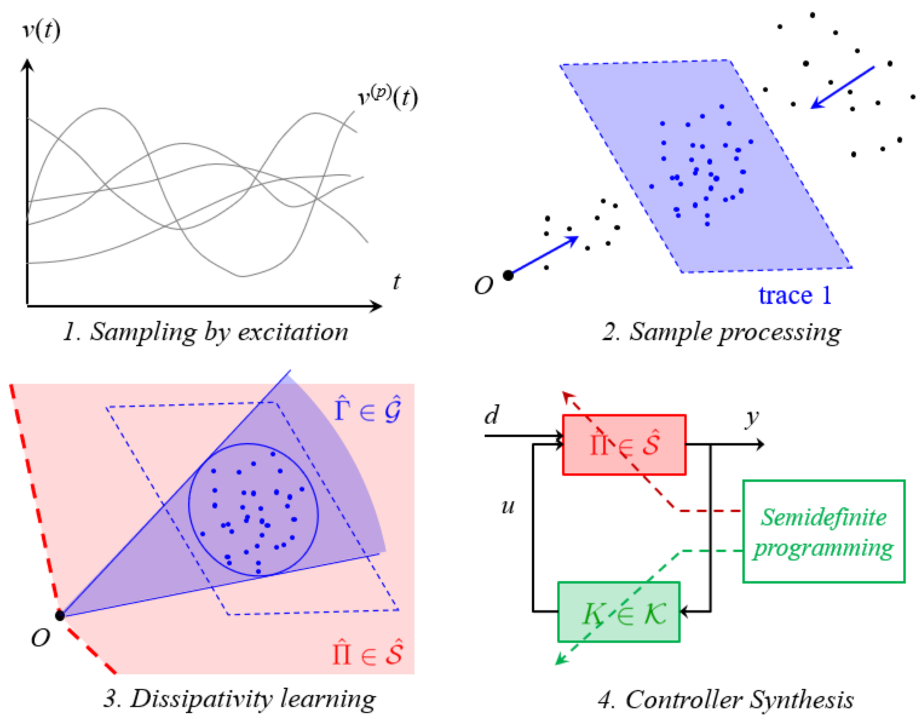

$v(t)$
$v^{(p)}(t)$
t
1. Sampling by excitation
O
trace 1
2. Sample processing
$\hat{\Gamma} \in \hat{\mathcal{G}}$
O
$\hat{\Pi} \in \hat{\mathcal{S}}$
3. Dissipativity learning
d
y
u
$\hat{\Pi} \in \hat{\mathcal{S}}$
Semidefinite programming
$K \in \mathcal{K}$
4. Controller Synthesis

Figure 14.1: Procedures involved in the DLC framework.

14.4 The DLC Framework

14.4.1 General DLC algorithm and its performance

At this point we summarize the results in the previous two sections. Generally, the DLC approach to input–output data-driven model-free control contains the following procedures, for which an illustration is given in Figure 14.1.

1. *Sampling by excitation*: Densely choose admissible input signals by the vector of Fourier coefficients, and heuristically select effective samples (e.g., using the sorting heuristics). Alternatively, samples may be generated directly by a heuristic rule.

2. *Sample processing*: Calculate the dual dissipativity parameters of the sample input–output trajectories. Then choose matrix basis, and vectorize the dual dissipativity parameters and normalize to trace 1.

3. *Dissipativity learning*: Perform a PCA procedure on the vector samples to find the mixing matrix. Characterize the dissipativity set by (14.39).

4. *Controller synthesis*: Find a solution (or the optimal solution) of K satisfying the matrix inequality (14.41) with some $\hat{\Pi} \in \hat{\mathcal{S}}$ and some (or the smallest) $\beta \geq 0$.

The performance guarantee of DLC is formalized as the following theorem, which follows from the lemmas in the previous two sections.

Theorem 14.1 (Performance of DLC)**.** *Suppose that*

- *Assumption 14.1 (on the existence of an effective reaching domain in the vicinity of the origin) and Assumption 14.2 (on the selection of dense samples around the effective reaching domain) hold;*
- *For any $\Gamma \in \hat{\mathcal{G}}$, there exists $\hat{\Gamma} \in \hat{\mathcal{G}}$ such that $\|\Gamma - \hat{\Gamma}\| \leq \epsilon \|\hat{\Gamma}\|$ for some $\epsilon > 0$;*
- *There exists $K \in \mathcal{K}$, $\hat{\Pi} \in \hat{\mathcal{S}}$, and $\beta \geq 0$ satisfying* (14.41)*, with H specified in* (14.40)*;*
- $2 \sin \arcsin(\epsilon/2) \cdot \|\hat{\Pi}\| < 1$.

Then on $\check{\mathcal{D}}$ the closed-loop system under output-feedback control law $u = Ky$ is L^2-stable, with the L^2-gain from d to $z = (y, u)$ upper bounded by $\beta + \delta_\beta$, where $\delta_\beta = (1+\beta)\delta\|\hat{\Pi}\|/(1 - \delta\|\hat{\Pi}\|)$.

14.4.2 Solution of the DLC algorithm

For controller synthesis where we consider to seek the optimal solution $(K, \hat{\Pi}, \beta) \in \mathcal{K} \times \hat{\mathcal{S}} \times [0, +\infty)$, it often helps to accelerate the semidefinite programming problem solution by tightening the feasible region. We first impose constraints $\hat{\Pi}_{vv} \succeq 0$, which indicates that an excitation at the origin can not lead to any further decrease of the storage function. The two-letter subscript stands for the matrix block whose rows and columns correspond to the indicated process variables, i.e.,

$$\Pi = \begin{bmatrix} \Pi_{yy} & \Pi_{yu} & \Pi_{yd} \\ \Pi_{uy} & \Pi_{uu} & \Pi_{ud} \\ \Pi_{dy} & \Pi_{du} & \Pi_{dd} \end{bmatrix} = \begin{bmatrix} \Pi_{yy} & \Pi_{yv} \\ \Pi_{vy} & \Pi_{vv} \end{bmatrix}. \tag{14.45}$$

We then note that the matrix inequality (14.41) is not convex, but is multi-convex in $(\hat{\Pi}, \beta)$ and K as two groups of variables. An iterative algorithm can therefore be adopted [411]. That is, in each iteration, the controller K is first fixed to solve the remaining variables, and then the remaining variables are fixed to update K.

To update K, we let $\Pi' = \hat{\Pi} + \mathrm{diag}(I, I, -\beta I)$ and rewrite (14.41) as

$$\begin{bmatrix} \Pi'_{yy} + K^\top \Pi'^\top_{yu} + \Pi'_{yu} K + K^\top \Pi'_{uu} K & \Pi'_{yd} + K^\top \Pi'_{ud} \\ \Pi'^\top_{yd} + \Pi'^\top_{ud} K & \Pi'_{dd} \end{bmatrix} \preceq 0. \tag{14.46}$$

When (14.41) is satisfied by some Π and β, the bottom-right block Π'_{dd} is negative semidefinite. The Schur complement of the above matrix over the Π'_{dd} block is therefore

$$\begin{aligned} &K^\top \left(\Pi'_{uu} - \Pi'_{ud}\Pi'^{-1}_{dd}\Pi'^\top_{ud}\right) K + K^\top \left(\Pi'^\top_{yu} - \Pi'_{ud}\Pi'^{-1}_{dd}\Pi'^\top_{yd}\right) \\ &+ \left(\Pi'_{yu} - \Pi'_{yd}\Pi'^{-1}_{dd}\Pi'^\top_{ud}\right) K + \left(\Pi'_{yy} - \Pi'_{yd}\Pi'^{-1}_{dd}\Pi'^\top_{yd}\right). \end{aligned} \tag{14.47}$$

Since $\Pi'_{uu} \succeq 0$, $\Pi'_{dd} \preceq 0$, and hence $\Pi'_{uu} - \Pi'_{ud}\Pi'^{-1}_{dd}\Pi'^\top_{ud} \succeq 0$, the negative semidefiniteness of the Schur complement is best achieved by a controller gain matrix of

$$K = \mathrm{proj}_{\mathcal{K}}\left(-\left(\Pi'_{uu} - \Pi'_{ud}\Pi'^{-1}_{dd}\Pi'^\top_{ud}\right)^{-1}\left(\Pi'^\top_{yu} - \Pi'_{ud}\Pi'^{-1}_{dd}\Pi'^\top_{yd}\right)\right) \tag{14.48}$$

where the projection operator $\mathrm{proj}_{\mathcal{K}}(\cdot)$ means the element of $\mathcal{K}$ with the smallest distance to the object in the parentheses, which is computationally tractable if $\mathcal{K}$ is convex.

Hence each iteration executes the following three steps.

1. Obtain (Π, β) from

$$\begin{aligned} \min_{\hat{\Pi}, \beta}\ & \beta \\ \text{s.t. } & H^\top \left(\hat{\Pi} + \mathrm{diag}(I, I, -\beta I)\right) H \preceq 0 \\ & \hat{\Pi} \in \hat{\mathcal{S}},\ \ \hat{\Pi}_{vv} \succeq 0,\ \ \beta \geq 0; \end{aligned} \tag{14.49}$$

2. Let $\Pi' = \hat{\Pi} + \mathrm{diag}(I, I, -\beta I)$ and update K by (14.48);

3. Reset the loop-closing matrix H according to (14.40).

The solution of semidefinite programming problem (14.49) can be executed with the `cvx` software installed into Matlab or the `cvxopt` package for Python. For example, when `cvxopt` is used and the estimated dissipativity set is given by (14.39), one can verify that (14.49) can be converted into the following standard form of conic programming [305] with variables $\chi = (\pi, \pi', \beta, \xi_0, \xi_1, \pi'', \pi_v)$:

$$\begin{aligned} \min\ & \beta \\ \text{s.t.}\ & M\chi = c \\ & \beta \geq 0,\ \ (\xi_0, \xi_1) \in \mathbb{L},\ \ \pi'' \in \mathbb{S},\ \ \pi_v \in \mathbb{S}. \end{aligned} \tag{14.50}$$

where

$$M = \begin{bmatrix} I & -I & -\mathrm{vec}(\mathrm{diag}(0,0,I)) & 0 & 0 & 0 & 0 \\ 0 & \Psi & 0 & 0 & 0 & I & 0 \\ -\Phi_{vv} & 0 & 0 & 0 & 0 & 0 & I \\ \bar{\gamma}^\top & 0 & 0 & -1 & 0 & 0 & 0 \\ \Theta G^\top & 0 & 0 & 0 & -I & 0 & 0 \end{bmatrix}$$

$$c = (-\mathrm{vec}(\mathrm{diag}(I, I, 0)), 0, 0, 0, 0).$$

Here the matrix Ψ is a transformation of π' equivalent to the loop closing operator $H^\top \Pi' H$ imposed on the original matrix π', Π'. That is, if $\mathrm{vec}(\dot{})$ is such that $\Pi'_{ij} = \pi'_{k(i,j)}$, then $\Psi_{k(i,j)k(i',j')} = H_{i'i}H_{j'j}$. The matrix Φ_{vv} selects components of π corresponding to the (v, v)-block of Π. The function vec converts matrices into vectors. $\mathbb{L}$ stands for the second-order cone (also known as Lorentz cone) $\{(\xi_0, \xi_1) | \xi_0 \geq \|\xi_1\|\}$, and $\mathbb{S}$ is the cone of positive semidefinite matrices of appropriate order.

14.4.3 Tuning of the DLC algorithm

The DLC algorithm contains multiple hyperparameters in the sampling and dissipativity learning steps, which may affect the performance of the final controller. Specifically, in the sampling stage, the hyperparameters for the set of admissible input signals, namely N_f (number of significant terms in Fourier series) and ϵ_f (allowable truncation error), affect the range of excitation signals generated, and T (time span of excitations) affects the reachable domain. Generally, if N_f is too high or ϵ_f is too low, there may be an abuse of signals with high-frequency oscillations. On the contrary, the allowed bandwidth of the signals may be too shallow. Both extremes can restrict the effectiveness of the excitation signals. T should also be well chosen so that the reachable domain is of a proper size, in which the control performance is of interest.

For the statistical inference of effective reaching domain, two hyperparameters are

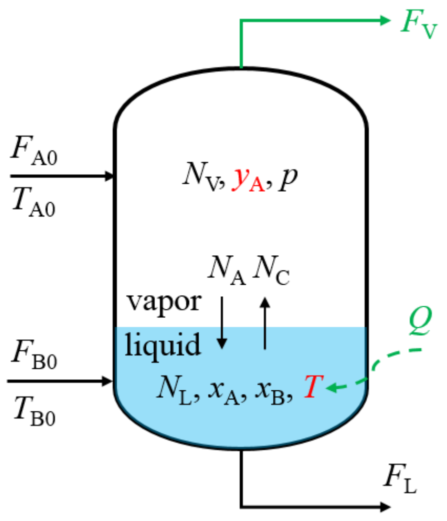


Figure 14.2: Two phase reactor.

involved if using PCA for a second-order cone estimation, namely the number of principal components J and the threshold of the Hotelling statistics Θ. These two hyperparameters represent the reduced dimension and the confidence level, respectively. Two analogous hyperparameters appeared in a polyhedral cone estimation scheme used in our previous work [411]. If J and Θ are large, the estimated $\hat{\mathcal{G}}$ is large and hence the estimated dissipativity set $\hat{\mathcal{S}}$ is small, which may result in the conservativeness (namely sub-optimality) of controller performance. On the other hand, if J and Θ are too small, $\hat{\mathcal{S}}$ is large, which may bring the risk of obtaining unrealistically optimistic L^2-performance. Therefore, J and Θ needs to be tuned to a suitable range. Since J affects the dimension of the feasible region of the semidefinite programming problem, we expect that the impact of J on the control performance is more significant than that of Θ, and needs to be tuned in priority.

14.5 Case Study: Two-Phase Reactor

In this section we examine the efficacy of DLC in data-driven model-free control with a two-phase reactor system [209], shown in Fig. 14.2. The deterministic first-principles model in the aforementioned paper is considered as the true dynamics, which we assume to be unknown for the controller synthesis and only used for generating simulated trajectories. For simplicity we consider regulating control at a steady state. DLC is compared to LQG optimal control based on a linear state-space model identified using the Kalman-Ho subspace method [80].

14.5.1 System description

Consider a two-phase reactor where the vapor holdup is not negligible compared to the liquid. The reactants A and B are fed into the vapor and liquid phase, respectively. The reaction $\mathrm{A} + \mathrm{B} \longrightarrow \mathrm{C}$ takes place in the liquid phase at a total rate of

$$R_{\mathrm{tot}} = k_0 \exp(-E_{\mathrm{a}}/RT) M_{\mathrm{L}} \rho x_{\mathrm{A}} x_{\mathrm{B}}. \tag{14.51}$$

B is assumed to be a non-volatile component, and components A and C in the two phases are in equilibrium, whose saturated pressures are dependent on temperature by the Antoine equations (the unit for pressures is Pa and temperature is K):

$$\ln P_{\mathrm{A}}^{\circ} = 30.5 - \frac{3919.7}{T - 34.1}, \ \ \ln P_{\mathrm{C}}^{\circ} = 30.0 - \frac{5000.0}{T + 70.0}. \tag{14.52}$$

The dynamic model is expressed as a set of differential algebraic equations (DAE) of index 2:

$$\begin{aligned}
\dot{M}_{\mathrm{V}} &= F_{\mathrm{A0}} - N_{\mathrm{A}} + N_{\mathrm{C}} - F_{\mathrm{V}} \\
\dot{y}_{\mathrm{A}} &= \frac{1}{M_{\mathrm{V}}}[(F_{\mathrm{A0}} - N_{\mathrm{A}})(1 - y_{\mathrm{A}}) - N_{\mathrm{C}} y_{\mathrm{A}}] \\
\dot{M}_{\mathrm{L}} &= F_{\mathrm{B0}} - F_{\mathrm{L}} - r_{\mathrm{tot}} + N_{\mathrm{A}} - N_{\mathrm{C}} \\
\dot{x}_{\mathrm{A}} &= \frac{1}{M_{\mathrm{L}}}[(N_{\mathrm{A}} - r_{\mathrm{tot}})(1 - x_{\mathrm{A}}) + (N_{\mathrm{C}} - F_{\mathrm{B0}}) x_{\mathrm{A}}] \\
\dot{x}_{\mathrm{B}} &= \frac{1}{M_{\mathrm{L}}}[(F_{\mathrm{B0}} - r_{\mathrm{tot}})(1 - x_{\mathrm{B}}) + (N_{\mathrm{C}} - N_{\mathrm{A}}) x_{\mathrm{B}}] \\
\dot{T} &= \frac{1}{(M_{\mathrm{L}} + M_{\mathrm{V}}) c_p}[F_{\mathrm{A0}} c_p (T_{\mathrm{A0}} - T) + F_{\mathrm{B0}} c_p (T_{\mathrm{B0}} - T) \\
&\quad + (N_{\mathrm{A}} - N_{\mathrm{C}}) \Delta H_{\mathrm{vap}} - r_{\mathrm{tot}} (\Delta H_{\mathrm{R}}^{\circ} - c_p (T - T^{\circ})) + Q] \\
0 &= P_{\mathrm{A}}^{\circ}(T) x_{\mathrm{A}} - P y_{\mathrm{A}} \\
0 &= P_{\mathrm{C}}^{\circ}(T)(1 - x_{\mathrm{A}} - x_{\mathrm{B}}) - P(1 - y_{\mathrm{A}}) \\
0 &= M_{\mathrm{V}} R T - P(V_{\mathrm{tot}} - M_{\mathrm{L}}/\rho).
\end{aligned} \tag{14.53}$$

The parameters and nominal values of variables are given in Table 14.1. For simulation, we convert the DAE model into a set of ordinary differential equations (ODEs) by solving for the pressure P from the last algebraic equation and differentiating the two

Table 14.1: Parameters and nominal values of variables for the two-phase reactor

c_p	80 J/(mol $\cdot$ K)	ρ	15 kmol/m^3
k_0	10^{11} m^3/(mol $\cdot$ s)	E_a	110 kJ/mol
F_{A0}	171.25 mol/s	F_{B0}	300 mol/s
T_{A0}	310 K	T_{B0}	298 K
V_{tot}	3.0 m^3	F_L	375 mol/s
ΔH_R	−50 kJ/mol	ΔH_{vap}	20 kJ/mol
R	8.314 J/(mol $\cdot$ K)	T°	298 K
F_V	50 mol/s	Q	−3422.6 kW
M_L	12812.88 mol	M_V	12834.25 mol
x_A	0.23789	x_B	0.67667
y_A	0.71579	T	341.51 K

phase equilibrium equations to solve for N_A and N_C.

For the system, we consider the outlet flow rates F_V and F_V simultaneously and the heating rate Q as manipulated inputs, and the vapor phase composition of A y_A and temperature T as controlled outputs. Disturbances in F_{B0} and T_{A0} are considered. The control inputs, disturbances and outputs are scaled by:

$$\begin{aligned} &F_\mathrm{V} = F_{\mathrm{V,nom}} + u_1,\;\; F_\mathrm{L} = F_{\mathrm{L,nom}} - u_1,\;\; Q = Q_{\mathrm{nom}} + 25u_2,\;\; F_{\mathrm{B0}} = F_{\mathrm{B0,nom}} + d_1 \\ &T_{\mathrm{B0}} = T_{\mathrm{B0,nom}} + 2.5d_2,\;\; y_\mathrm{A} = y_{\mathrm{A,nom}} + 0.001y_1,\;\; T = T_{\mathrm{nom}} + 0.5y_2. \end{aligned} \tag{14.54}$$

14.5.2 Dissipativity learning control

For dissipativity learning we generate 1000 trajectory samples, where the excitation signals are randomly generated using 5 terms in the Fourier series during $T = 120$ seconds. 100 of the sample trajectories are shown in Fig. 14.3, where the output endpoints cover approximately output ranges of $[-5, 5]$ for y_1 and $[-2, 2]$ for y_2. We consider this as an appropriate domain from which the system is expected to be controlled towards the origin. We consider designing a PID controller and therefore the outputs are augmented with their integrals and derivatives. The dual dissipativity parameters for each trajectory sample comprise a symmetric matrix of order 10 which is vectorized into a 100-dimensional vector by column stacking.

After whitening the dual dissipativity parameter samples, PCA is performed. The

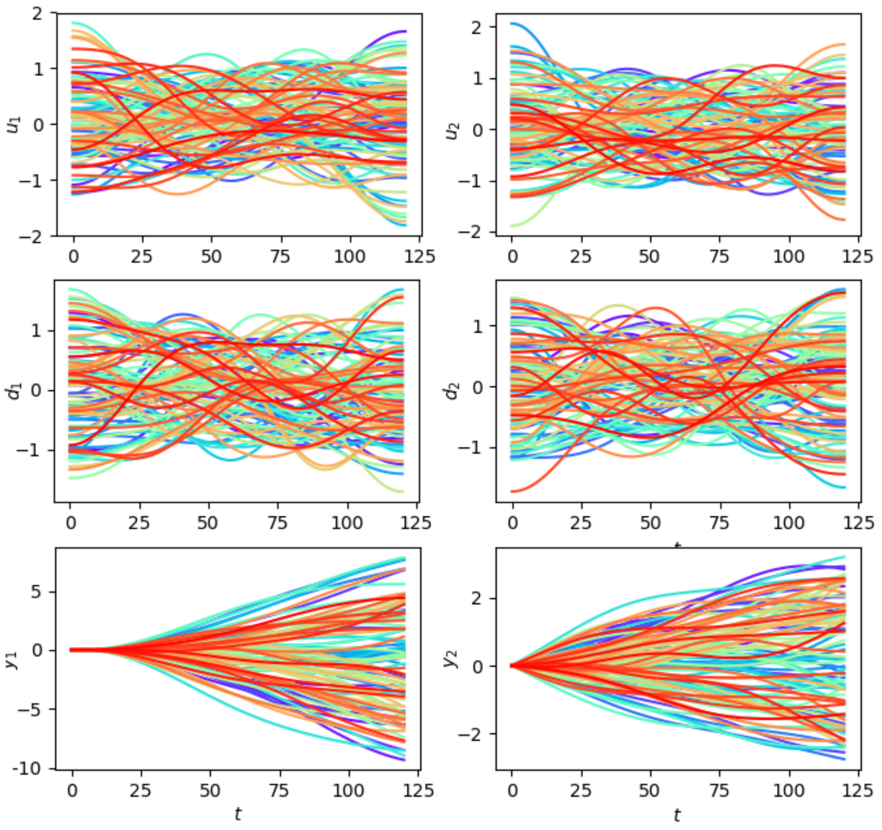
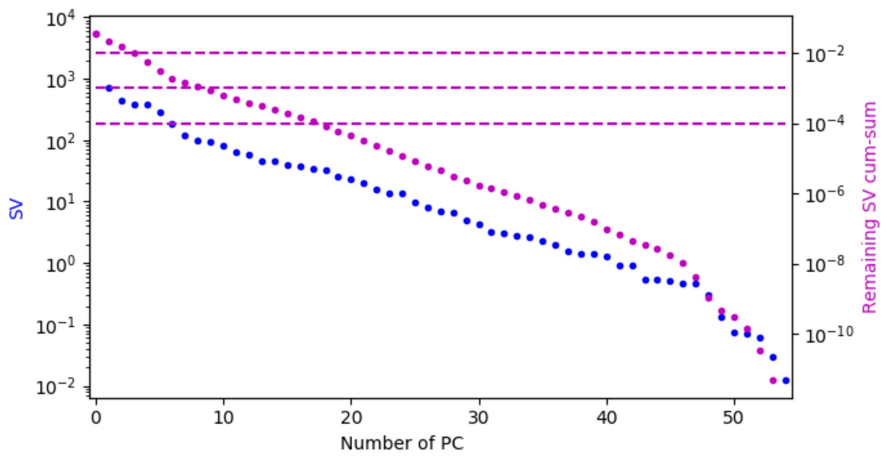


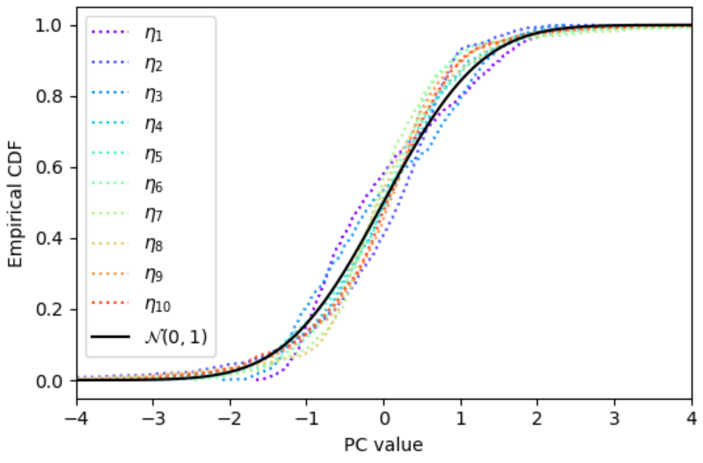
Figure 14.3: Trajectory samples for DLC of the two-phase reactor.

resulting singular values s_p and the remaining fraction of the singular value cumulative sum $\frac{\sum_{q=p+1}^{K} s_q^2}{\sum_{q=1}^{K} s_q^2}$ are shown in Fig. 14.4(a), from which it can be shown that for retaining 99%, 99.9%, and 99.99% of the data variations, only the 4, 10, and 19 leading singular values are needed, respectively. The empirical distributions of the 10 leading principal components of the samples' dual dissipativity parameters are shown in Fig. 14.4(b) in the form of empirical cumulative density functions (CDFs), which are observed to be close to that of the standard normal distribution $\mathcal{N}(0, 1)$. It is hence appropriate to use a PCA-based estimation of the dissipativity set (14.39) for the subsequent controller synthesis.

Setting the number of principal components as 4 and the confidence level as 90% (i.e., the $\hat{\mathcal{G}}$ covers 90% of the dual dissipativity parameter samples) and following the iterative procedure in Subsection 14.4.2, the following PID controller is obtained from



(a) Singular values



(b) Distribution of the principal components

Figure 14.4: PCA results of the dissipativity parameter samples.

DLC synthesis:

$$\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = K_\mathrm{P} \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} + K_\mathrm{I} \int_0^t \begin{bmatrix} y_1(\tau) \\ y_2(\tau) \end{bmatrix} d\tau + K_\mathrm{D} \begin{bmatrix} dy_1(t)/dt \\ dy_2(t)/dt \end{bmatrix}, \tag{14.55}$$

where

$$K_\mathrm{P} = \begin{bmatrix} -1.1632 & 0.4930 \\ 1.7109 & -2.9244 \end{bmatrix}, \; K_\mathrm{I} = \begin{bmatrix} 1.4628 & 7.3509 \\ -0.6166 & 0 \end{bmatrix} (\mathrm{h}^{-1}),$$

$$K_{\mathrm{D}} = \begin{bmatrix} -0.8131 & 5.0374 \\ 11.1784 & -18.9960 \end{bmatrix} (\mathrm{s}).$$

The above procedures can be used for synthesizing PI and P controllers by removing the rows and columns of the dual dissipativity parameter samples corresponding to the output derivatives and integrals. The DLC-PI controller is given by

$$K_{\mathrm{P}} = \begin{bmatrix} -0.4035 & 0.6516 \\ 0.2325 & -3.7077 \end{bmatrix}, \quad K_{\mathrm{I}} = \begin{bmatrix} 1.7731 & 2.8824 \\ -0.9406 & 0.0040 \end{bmatrix} (\mathrm{h}^{-1}), \tag{14.56}$$

and the DLC-P controller is

$$K_{\mathrm{P}} = \begin{bmatrix} -0.3033 & 0.1631 \\ 0.1479 & -2.3213 \end{bmatrix}. \tag{14.57}$$

To test the performance of the DLC controllers, we generate the disturbances as piecewise constant signals, where each component in each time interval of 10 seconds is held at a value randomly generated through a standard normal distribution, and obtain the closed-loop input and output trajectories in 3600 seconds. The results of such simulations under the DLC-PID, DLC-PI and DLC-P controllers are shown in Fig. 14.5). The control performance can be measured by the integrated squares of errors of the outputs (ISE) $\frac{1}{3600}\int_0^{3600} \|y(t)\|^2 dt$ and of control inputs (ISC) $\frac{1}{3600}\int_0^{3600} \|u(t)\|^2 dt$. The ISE+ISC index is 2.5846, 2.4316, and 2.5345 for the PID, PI and P controllers, respectively, among which the PI controller slightly outperforms the other two, indicating that it suffices to use a PI or even a P controller for disturbance rejection around the steady state. The performance of the DLC controllers is also compared to the open-loop trajectories (with fixed $u = 0$) which result in significantly larger deviations in the two outputs, with $\mathrm{ISE} + \mathrm{ISC} = 35.0907$.

14.5.3 Comparison with identification-based LQG control

The Kalman-Ho algorithm is a classical algorithm for finding the minimal realization of a linear time-invariant (LTI) system from the impulse response, which can be obtained through the differences of step responses. Given a desirable reduced order of states, the algorithm can be used for identification with an approximate singular value decomposition. For the two-phase reactor, we choose a sampling time of 10 seconds and specify the order of states as 6 to match the step responses. With the LTI system model (A, B, C),

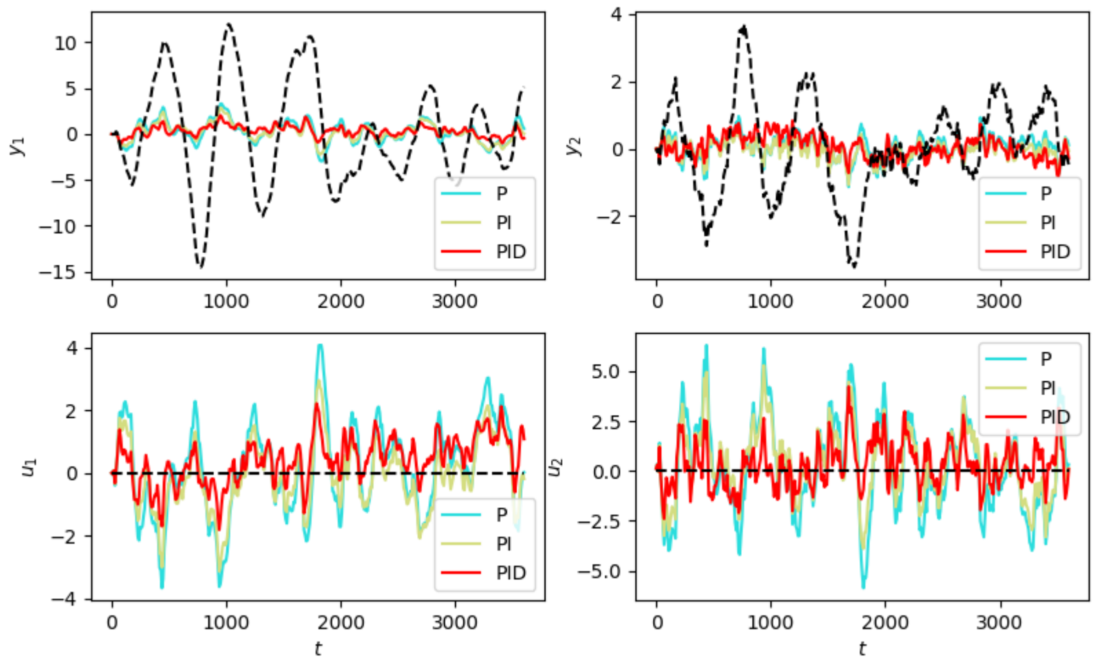


Figure 14.5: Simulated closed-loop input and output trajectories under the DLC-P, DLC-PI, and DLC-PID controllers compared to open-loop trajectories (black dashed).

under a state disturbance covariance matrix $Q = I_{12}$ and output noise covariance matrix $R = 0.1I_2$, a linear quadratic Gaussian (LQG) controller [386] is obtained:

$$\dot{\hat{x}} = A\hat{x} + Bu + K_{\mathrm{f}}(y - C\hat{x}), \;\; u = -K_{\mathrm{r}}\hat{x}, \tag{14.58}$$

in which $\hat{x}$ stands for the filtered state observations, K_{f} is the filter gain, and K_{r} is the regulator gain.

Fig. 14.6 shows the closed-loop input and output trajectories under the LQG controller during a simulation time of 3600 seconds. It is observed that in comparison to the open-loop trajectories, the LQG controller also apparently reduced the impact of disturbances on the system (with an ISE+ISC measure for the LQG controller equal to 2.6766). Although increasing the number of states in the linear system identification to 8, 10, and 12 results in step responses of the identified system that better resemble those of the original nonlinear system, the LQG performance is not improved (with ISE+ISC equal to 2.7535, 2.7806 and 2.8103, respectively). On the other hand, reducing the number of states to 4 results in closed-loop divergence. Hence the performance of the

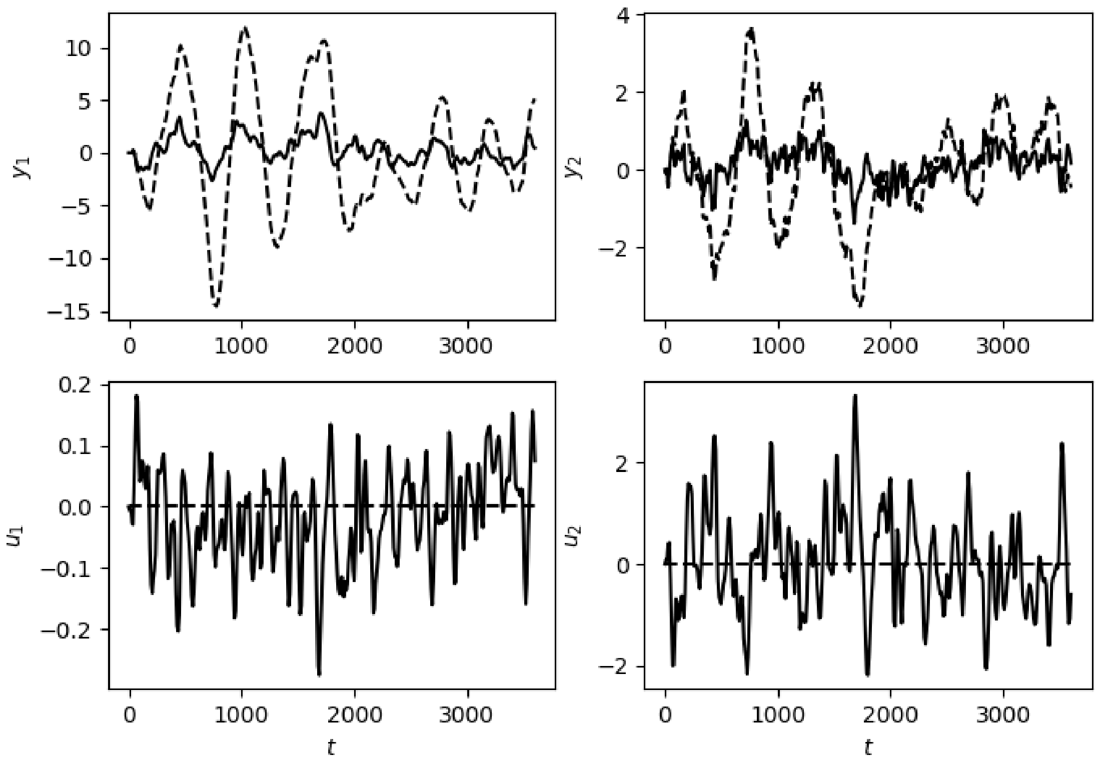


Figure 14.6: Simulated closed-loop trajectories under the LQG control (solid) compared to the open-loop trajectories (dashed).

LQG controller with 6 states is considered as the best achievable by such linear system identification-based optimal control.

Comparing the performance of the DLC-PI and 6-state LQG controllers, we conclude that DLC (ISE + ISC = 2.4316) outperforms LQG (ISE + ISC = 2.6766) by approximately 9.2%. Moreover, by comparing the trajectories in Figures 14.5 and 14.6, we observe that DLC and LQG choose different ways to reject exogenous disturbances. For DLC, the magnitudes of inputs and outputs turn out to be more balanced, while the LQG prefers to tradeoff larger output deviations (especially y_1) for smaller inputs (especially u_1). This may be related to the different ways that these two types of controllers are synthesized. The dissipativity learning relies on the overall (integrated) input–output responses over an excitation time period, while linear system identification is based on the incremental responses during each short sampling time.

14.6 Conclusions

As a continuation of our previous work [411], in this paper we have discussed the DLC framework for input–output data-driven control of chemical processes. The nominal control performance of DLC and the impact of the dissipativity learning procedures, including the sampling of excitation signals and statistical inference of the effective dual dissipativity set, are formalized. Concisely speaking, *DLC achieves nearly L^2-optimal control within an effective reachable domain, assuming that the sampling and statistical inference steps are sufficiently accurate with respect to the effective dual dissipativity set*. The DLC strategy is implemented on a two-phase reactor, where its performance is compared to an optimal controller based on linear system identification. Through the current studies, we have established DLC as a generic and standardized input–output data-driven control strategy for process systems that gives satisfactory control performance.

We note that the current DLC approach is limited to the standard form of (14.1) and the resulting control law is limited to PID forms due to the quadratic supply rate function. Extending the DLC framework to more general dynamics, e.g., time-delay systems, and designing various forms of linear and nonlinear controllers, will be of importance to expand the applications. More importantly, the applications are yet limited to small systems at the process unit level. It is highly desirable to extend DLC to large-scale interconnected chemical plants. These topics will be addressed in our upcoming works.

Chapter 15
Conclusion and Future Directions

(I'll) wait until the thousand miles of snow melts,	等到千里冰雪消融，
until the arrival on the prairie of the breeze of spring.	等到草原上送來春風，
When the land of Kokdala has changed into a brand new look,	可克達拉改變了模樣，
then along with my music the loved one will come to sing.	姑娘就會來伴我的琴聲。

Zhang Jiayi, "*The Night in the Prairie*", 1959.
(張加毅《草原之夜》)

In the three parts of this thesis, three important aspects of process control – structuredness, optimality and intelligence – have been discussed. For these three aspects, conclusions and discussions on potential future directions are made in the following sections, respectively. As a whole, these works provide new perspectives to the future of process control aiming at the three main objectives mentioned in the Introduction section. It is hoped that the methods proposed here are of interest and applicability in the process control practice, and may inspire some beneficial further explorations.

15.1 Deepening the Understanding of Network Structures

The works in Part I have revealed that the community structures play a crucial role in the control of networks by reducing the total control cost under a significant cost of feedback channels. If such a total control cost accounting for sparsity is used as an objective function for network topology and feedback controller design, then the fittest network will become modular with a distributed control architecture. Based on this principle, the design of a distributed control architecture can follow a community detection procedure in network representations of process systems. For such systems three different paths have been proposed, namely input–output bipartite networks using short-time interaction measures, bipartite networks using RTAGA, and variable-constraint bipartite/unipartite networks. With ongoing works in the research group in using community detection in optimization problems, the potential of integrating network theory in a wider range of process systems engineering applications will be further developed.

It appears that studies regarding network dynamics and control have attracted more and more attention from network scientists, electrical engineers, systems biologists, psychologists, and economists. In chemical engineering, the investigations on utilizing network structures for control and optimization had been scarce. Based on the research included in this thesis, the following directions are proposed.

Future direction 1: The role of network structures other than communities. Studies in network science have revealed many other types of macroscopic and mesoscale structures in real-world networks. For example, in [69], it was found that under high costs of feedback channels, the overall distribution of node degrees has a stronger impact on the total control cost than community structures. It was also discussed in [76] that *core-periphery* structures may be useful in designing a hierarchically structured controller or optimizer. *Assortativity* (or disassortativity), i.e., the inclination of nodes to connect to nodes of higher degrees (or lower degrees), namely the positive (or negative) degree correlativity, is another distinctive feature of many networks [291]. The effects of these structures on sparse optimal control of networks have not yet been understood. Moreover, since network topology may be an optimal combination or a tradeoff among these topological features, it is desirable to explore the underlying mechanism, for example in the setting of sparse control, for networks to form such topology. This research at the interface of network science and control theory will facilitate a deeper understanding of the structural and functional organization of biological systems such as metabolic map and brains.

Future direction 2: Elucidation of the effect of decompositions on control and optimization. Although the advantages of finding decompositions through community detection have been examined in simulation studies, the *mechanism* of the effect of decompositions on the algorithms of control or optimization has not been understood through rigorous derivations. For such an investigation, one needs to properly quantify at least (i) the computational time required for calculating each step in iterative algorithms, (ii) the communication effort required for coordinating distributed controllers or optimization solvers, and (iii) the anticipated final result or performance under decompositions. This will help to discover the truly *optimal decomposition* for control and optimization problems from their formulations in general. However, difficulties should be expected to this end. An a priori accurate estimation of the performance before actually solving the problem is usually unavailable. Typically, one may only infer the computational complexity in orders-of-magnitude (e.g., linear/quadratic convergence) from the nature of the problem and the type of algorithm, which does not reflect the effect of decompositions. Moreover, for nonconvex and especially nonconvex constrained problems, the theory of distributed algorithms is not yet well established.

Future direction 3: The role of network structures in control-relevant properties other than sparsity. In the current research, the importance of community structures in network control is based on the argument of sparsity, which is conceptually related to the complexity of the feedback mechanism or the communication load among the controller agents. Apart from sparsity, there are other desirable properties of control that have not been considered here. For example, in the presence of faults in the system, it is desirable that the controller is reconfigured to result in minimal influence on the production, i.e., the controller is *fault-tolerant* [264]. One may also want the controller to be structured in such a way that the disturbance propagation on the process is well suppressed, i.e., the controller is *robust*. Also, it is hoped that the failure of one single or a few communication channels does not cause serious deterioration in the control performance, i.e., the controller is *resilient*. In network science, the studies of epidemic or percolation dynamics [273] and network resilience [318] may be relevant for insights into such properties.

15.2 Exploring the Interface of Optimization and Control

The chapters in Part II of the thesis offered answers to two important questions faced by the state-of-the-art nonlinear control, namely (i) how to solve distributed nonlinear MPC problem with a rigorous algorithm, and (ii) how to analyze the Lyapunov stability and design the Lyapunov function in an optimal sense. For the former question, the ELLADA algorithm is proposed for distributed nonconvex constrained optimization problems, which guarantees the convergence towards a stationary point, while at the same time adopting acceleration and approximation techniques. For the latter problem, bilevel and stochastic programming tools, widely used in the PSE research, are employed to formulate the problems in Lyapunov stability analysis and Lyapunov function and design, through the conceptual analogy between Lyapunov analysis and flexibility analysis. Moreover, based on Lyapunov analysis and design, a concept of Lyapunov dynamic flexibility is introduced, which can be used for integrating process design and control.

In the recent years, optimization algorithms are more and more deeply involved in the theory and applications of automatic control, and apparently, this has resulted in the development of methods and tools that have not attracted the attention of earlier chemical engineers, such as semidefinite programming (SDP) and linear matrix inequalities (LMIs) for linear systems theory [43], sum-of-squares (SOS) programming for polynomial systems theory [61], and robust optimization [29] for robust control. On the other hand, the optimization algorithms are not yet sufficiently well developed and efficient in dealing with control problems. The control and optimization perspectives are often not well combined. Hence the following directions are proposed.

Future direction 4: Efficient optimization algorithms for control. For the incorporation of more and more intricate optimization techniques into the control of chemical processes, algorithms for nonconvex problems and especially *nonconvex constrained problems* are of utmost importance. Although global optimization solvers such as BARON have been well developed, using them for real-time control is not sufficiently efficient, and a compromise to stationary solutions (e.g., using the IPOPT solver for NLP) is usually used. For *stochastic MPC and robust MPC*, solving the resulting NLP problems usually still requires prohibitive computational time. Moreover, for *distributed optimization*, there is no *global algorithm* to the best knowledge of the author, and even the algorithms for stationary solution are scarce. So far, it is unclear whether it is even realistic for distributed optimization algorithms to always reach solutions that are of equal quality to the centralized problem. Developing efficient algorithms under the complexities of optimization formulations may be extremely challenging.

Future direction 5: Simplification of optimization problems in control. As MPC problems as well as other control-related optimization problems may be difficult to solve, actions need to be taken to simplify the formulations without too much compromise in the resulting performance. A typical example is advanced-step MPC, where the solution of the MPC problems is estimated in advance of the sampling time and then corrected by the actual measurements using sensitivity analysis [476]. Apart from the optimization solver itself, the simplification of models may be of interest. Generally, it would be beneficial to explore methods of *model reduction* for nonlinear systems according to the model features, e.g., inertial manifolds for multi-time-scale dynamics [435] and clustering for repetitive subproblems [60]. For the components with complex transport-reaction kinetics, *surrogate models* on an empirical basis may be considered.

Complicating interactions may be encapsulated into uncertain regions if they are not too significant, while dealing with *uncertainties* in a both accurate and parsimonious way seems to be a long-lasting issue.

Future direction 6: Control theory of optimization. While optimization can be used for control, the use of control theory in optimization has been less discussed. Intuitively, it may be easily said that iterative optimization algorithms are in principle dynamic systems, where the iterated points are the states, and the search direction and step size correspond to the control inputs. However, it was only in recent years that the goal of deriving generic and well-tuned convex optimization algorithms motivated contributions from control theorists (e.g., [219]), where the convergence property and its rate are obtained through the stability analysis using the Lyapunov theory or integral quadratic constraints (IQC). For problems with constraints, nonconvexity, or even integer decision variables, it is not yet clear how to carry out analysis based on control perspectives. Especially, global optimization algorithms typically employ a branching strategy to avoid local optima, which does not conform to the existing paradigms of control systems.

15.3 Systematizing Data-Driven Techniques for Control

Part III of the thesis is focused on the topic of data-driven control. First, for the identification of nonlinear processes, it is pointed out that explicitly accounting for Lie derivatives in the objective function that captures the deviation of the identified model from the measured data helps to improve the performance of system identification and hence the performance of identification-based control. Then, for model-free control, an exploration of combining distributed optimization algorithms with adaptive dynamic programming was made. Finally, motivated by the fact that the states of chemical process systems are mostly partially observable and high-dimensional, input–output data-driven model-free control is considered, for which a framework based on learning the dissipativity property is adopted. Successful applications to some chemical process systems are documented, and a preliminary discussion on its statistical and control-theoretic foundations are presented.

The recent development of machine learning, especially deep learning, has provided more and more powerful approaches to empirically describing datasets and exploiting them for a wide spectrum of applications. In chemical engineering, these efforts are mostly focused on process fault detection and diagnosis (e.g., [453]). For process control, it is fair to claim that the application of machine learning techniques for practical data-driven control strategies is still in its infancy. This is because chemical processes typically consist of transport-reaction systems such as reactors, columns and heat exchangers, and have distinctive dynamic features. These units are nonlinear systems, often under disturbances and uncertainties, that may be integrated into large-scale systems. It is apparent that there is much more to explore in this field.

Future direction 7: Improved algorithms of data-driven process control. Despite the successful applications to several chemical processes, the proposed DLC approach is yet limited to small-scale and relatively simple processes with PID forms of controllers. Moreover, prior to the training procedure, offline open-loop trajectories must be collected, which requires to schedule nonproductive periods for plant tests. In the future development of data-driven process control, the following three important problems will need to be solved. First, the methods should be extensible to *large-scale interconnected systems*. In principle, the additivity of the dissipativity property can be used in a similar way to model-based dissipative control [157]. However, the interactions among subsystems through the internal unobservable states are not accounted for in such an approach, and the need to incorporate distributed algorithms in controller synthesis may emerge. Second, the methods should be extensible to controllers *of more complex forms*, e.g., linear controller represented by generic state-space dynamics and nonlinear controllers, which would require a generalization of the dissipativity property from standard quadratic forms. Third, it is desirable to develop the methods into *adaptive algorithms*, where the essential control-relevant information can be updated online in a recursive way, thus avoiding collecting a large number of trajectories for learning.

Future direction 8: Complexity, informativeness and reconciliation of data. For the utilization of process data for control, it is necessary to develop a better fundamental understanding of how the data affect the learning procedures and subsequently the control performance. This involves at least the following three aspects. First, it is beneficial to *derive theoretical conditions* (e.g., how much and what kind of data is needed) for the learner to infer the control-relevant information from available data.

From these conditions, the data complexity for obtaining the control-relevant information or the error of such inference is better understood. Second, there always appears to be an instrinsic tradeoff in the data-driven technology – big data and intricate techniques (e.g., deep learning) may give more accurate descriptions of the object, while relatively simpler descriptions based on smaller data may be easier to handle and use for specific purposes (e.g., in optimization problems one prefer to have linear or convex constraints). It is hence desirable to *evaluate the informativeness* of datasets and decide a priori how to generate the appropriate data for the use in control. Third, in the presence of noise, delays, dropouts or other flaws in the process measurements, one may need to know how to *validate or reconcile* the data, i.e., to correct the flawed raw datasets into consistent and informative ones.

Future direction 9: Data-driven optimization. Optimality is an important aspect of control that has been discussed at length in this thesis. The pursuit of control strategies that are in a properly defined sense optimal is reflected in MPC and EMPC techniques. Apart from control, other decision making problems in PSE, including scheduling, planning and process or product design, are typically solved via standardized mathematical programming formulations. So far, it remains highly challenging to deal with optimization problems where the objective value and constraints can only be evaluated as black box functions without explicit descriptions, known as *block-box* or *derivative-free optimization* [5, 42]. It is desirable to consider how to apply data-driven optimization to control or even develop such algorithms with a control perspective. As a long-term goal, the *integrations of multiple data-driven decision making technologies*, such as the integration of process monitoring and control and the integration of control and optimization, may be of interest.

15.4 Widening Applications

Finally, it should be emphasized that applying the already proposed and potential new methods to a wide spectrum of chemical engineering *applications* are important, both for examining the effectiveness of the proposed methods and for identifying the shortcomings, thus motivating more generic and practical approaches. For each type of problems, suitable examples, from low-dimensional, slightly nonlinear, certain, easily understandable and numerically constructed systems to large-scale, highly nonlinear, uncertain, poorly understood and realistic ones, should be found. It is desirable to establish benchmark libraries for system identification, control, monitoring purposes. However, it is fair to claim that many benchmark chemical processes in the available literature are inaccessible, inflexible to use, with errors in the information provided, or already outdated. Compared to the optimization problem libraries that are continually being updated, it was 27 years ago when a number of “industrial challenge problems” was collected[1]. Re-examining industrial processes and identifying current challenges will help researchers to better orient their work.

[1]See Volume 17, Issue 3 of *Computers & Chemical Engineering*.

References

[1] E. Aggelogiannaki and H. Sarimveis, *Nonlinear model predictive control for distributed parameter systems using data driven artificial neural network models*, Comput. Chem. Eng., 32 (2008), pp. 1225–1237.

[2] A. Allman, W. Tang, and P. Daoutidis, *Towards a generic algorithm for identifying high-quality decompositions of optimization problems*, in Computer Aided Chemical Engineering, vol. 44, Elsevier, 2018, pp. 943–948. 13th International Symposium on Process Systems Engineering (PSE 2018).

[3] ———, *DeCODe: A community based algorithm for generating high-quality decompositions of optimization problems*, Optim. Eng., 20 (2019), pp. 1067–1084.

[4] A. A. Alonso and B. E. Ydstie, *Process systems, passivity and the second law of thermodynamics*, Comput. Chem. Eng., 20 (1996), pp. S1119–S1124.

[5] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, *Simulation optimization: a review of algorithms and applications*, Ann. Oper. Res., 240 (2016), pp. 351–380.

[6] M. S. Andersen, S. K. Pakazad, A. Hansson, and A. Rantzer, *Robust stability analysis of sparsely interconnected uncertain systems*, IEEE Trans. Autom. Control, 59 (2014), pp. 2151–2156.

[7] D. G. Anderson, *Iterative procedures for nonlinear integral equations*, J. ACM, 12 (1965), pp. 547–560.

[8] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, *CasADi: a software framework for nonlinear optimization and optimal control*, Math. Prog. Comput., 11 (2019), pp. 1–36.

[9] M. Andreasson, D. V. Dimarogonas, H. Sandberg, and K. H. Johansson, *Distributed control of networked dynamical systems: Static feedback, integral action and consensus*, IEEE Trans. Autom. Control, 59 (2014), pp. 1750–1764.

[10] I. P. Androulakis, C. D. Maranas, and C. A. Floudas, *αBB: A global optimization method for general constrained nonconvex problems*, J. Global Optim., 7 (1995), pp. 337–363.

[11] G. Antonelli, *Interconnected dynamic systems: An overview on distributed control*, IEEE Control Syst., 33 (2013), pp. 76–88.

[12] Y. Arkun and J. Downs, *A general method to calculate input-output gains and the relative gain array for integrating processes*, Comput. Chem. Eng., 14 (1990), pp. 1101–1110.

[13] K. J. Åström and T. Hägglund, *PID controllers: theory, design, and tuning*, ISA, 1995.

[14] A. N. Atassi and H. K. Khalil, *A separation principle for the stabilization of a class of nonlinear systems*, IEEE Trans. Autom. Control, 44 (1999), pp. 1672–1687.

[15] J. Baillieul and P. J. Antsaklis, *Control and communication challenges in networked real-time systems*, Proc. IEEE, 95 (2007), pp. 9–28.

[16] L. Bakule, *Decentralized control: An overview*, Annu. Rev. Control, 32 (2008), pp. 87–98.

[17] E. Balas, *Disjunctive programming and a hierarchy of relaxations for discrete optimization problems*, SIAM J. Alg. Discr. Meth., 6 (1985), pp. 466–486.

[18] M. Baldea and P. Daoutidis, *Dynamics and Nonlinear Control of Integrated Process Systems*, Cambridge University Press, 2012.

[19] M. Baldea and I. Harjunkoski, *Integrated production scheduling and process control: A systematic review*, Comput. Chem. Eng., 71 (2014), pp. 377–390.

[20] J. Bao and P. J. Lee, *Process Control: The Passive Systems Approach*, Springer, 2007.

[21] A.-L. Barabási, *Network Science*, Cambridge University Press, 2016.

[22] A.-L. Barabási and R. Albert, *Emergence of scaling in random networks*, Science, 286 (1999), pp. 509–512.

[23] M. J. Barber, *Modularity and community detection in bipartite networks*, Phys. Rev. E, 76 (2007), p. 066102.

[24] R. Barkley and R. Motard, *Decomposition of nets*, Chem. Eng. J., 3 (1972), pp. 265–275.

[25] J. Barreiro-Gomez, C. Ocampo-Martínez, and N. Quijano, *Partitioning for large-scale systems: A sequential distributed MPC design*, IFAC-PapersOnLine, 50 (2017), pp. 8838–8843. 20th IFAC World Congress.

[26] G. Bartolini and E. Punta, *Reduced-order observer in the sliding-mode control of nonlinear nonaffine systems*, IEEE Trans. Autom. Control, 55 (2010), pp. 2368–2373.

[27] R. W. Beard, G. N. Saridis, and J. T. Wen, *Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation*, Automatica, 33 (1997), pp. 2159–2177.

[28] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[29] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*, Princeton University Press, 2009.

[30] B. W. Bequette, *Nonlinear control of chemical processes: A review*, Ind. Eng. Chem. Res., 30 (1991), pp. 1391–1413.

[31] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 3$^{\text{rd}}$ ed., 2016.

[32] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice Hall, 1989.

[33] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, 1996.

[34] D. Bertsimas, D. B. Brown, and C. Caramanis, *Theory and applications of robust optimization*, SIAM Rev., 53 (2011), pp. 464–501.

[35] N. Bhat and T. J. McAvoy, *Use of neural nets for dynamic modeling and control of chemical process systems*, Comput. Chem. Eng., 14 (1990), pp. 573–582.

[36] L. T. Biegler and D. M. Thierry, *Large-scale optimization formulations and strategies for nonlinear model predictive control*, IFAC-PapersOnLine, 51 (2018), pp. 1–15. The 6th IFAC Conference on Nonlinear Model Predictive Control (NMPC).

[37] L. T. Biegler and V. M. Zavala, *Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization*, Comput. Chem. Eng., 33 (2009), pp. 575–582.

[38] E. G. Birgin, C. A. Floudas, and J. M. Martínez, *Global minimization using an augmented Lagrangian method with variable lower-level constraints*, Math. Program., 125 (2010), pp. 139–162.

[39] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, *Fast unfolding of communities in large networks*, J. Stat. Mech. Theor. Exp., 2008 (2008), p. P10008.

[40] F. Boem, R. M. Ferrari, T. Parisini, and M. M. Polycarpou, *Optimal topology for distributed fault detection of large-scale systems*, IFAC-PapersOnLine, 48 (2015), pp. 60–65.

[41] L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for large-scale machine learning*, SIAM Rev., 60 (2018), pp. 223–311.

[42] F. Boukouvala, R. Misener, and C. A. Floudas, *Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO*, Eur. J. Oper. Res., 252 (2016), pp. 701–727.

[43] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, SIAM, 1994.

[44] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.

[45] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[46] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, *On modularity clustering*, IEEE Trans. Knowl. Data Eng., 20 (2008), pp. 172–188.

[47] E. Bristol, *On a new measure of interaction for multivariable process control*, IEEE Trans. Autom. Control, 11 (1966), pp. 133–134.

[48] D. Bruns and J. Bailey, *Nonlinear feedback control for operating a nonisothermal CSTR near an unstable steady state*, Chem. Eng. Sci., 32 (1977), pp. 257–264.

[49] F. Callier, W. Chan, and C. Desoer, *Input-output stability theory of interconnected systems using decomposition techniques*, IEEE Trans. Circuits Syst., 23 (1976), pp. 714–729.

[50] M. C. Campi, A. Lecchini, and S. M. Savaresi, *Virtual reference feedback tuning: a direct method for the design of feedback controllers*, Automatica, 38 (2002), pp. 1337–1346.

[51] M. C. Campi and S. M. Savaresi, *Direct nonlinear control design: The virtual reference feedback tuning (VRFT) approach*, IEEE Trans. Automat. Control, 51 (2006), pp. 14–27.

[52] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar, *Distributed model predictive control*, IEEE Control Syst., 22 (2002), pp. 44–52.

[53] V. Chandan and A. Alleyne, *Optimal partitioning for the decentralized thermal control of buildings*, IEEE Trans. Control Syst. Technol., 21 (2013), pp. 1756–1770.

[54] J.-W. Chang and C.-C. Yu, *The relative gain for non-square multivariable systems*, Chem. Eng. Sci., 45 (1990), pp. 1309–1323.

[55] N. Chatzipanagiotis and M. M. Zavlanos, *On the convergence of a distributed augmented Lagrangian method for nonconvex optimization*, IEEE Trans. Automatic Control, 62 (2017), pp. 4405–4420.

[56] C. Chen, B. He, Y. Ye, and X. Yuan, *The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent*, Math. Prog., 155 (2016), pp. 57–79.

[57] C.-C. Chen, C. Hwang, and R. Y. K. Yang, *Optimal periodic forcing of nonlinear chemical processes for performance improvements*, Can. J. Chem. Eng., 72 (1994), pp. 672–682.

[58] G. Chen, X. Wang, and X. Li, *Fundamentals of Complex Networks: Models, Structures and Dynamics*, John Wiley & Sons, 2014.

[59] X. Chen, M. Heidarinejad, J. Liu, and P. D. Christofides, *Distributed economic MPC: Application to a nonlinear chemical process network*, J. Process Control, 22 (2012), pp. 689–699.

[60] X. Cheng, Y. Kawano, and J. M. Scherpen, *Model reduction of multiagent systems using dissimilarity-based clustering*, IEEE Trans. Autom. Control, 64 (2018), pp. 1663–1670.

[61] G. Chesi, *Domain of Attraction: Analysis and Control via SOS Programming*, Springer, 2011.

[62] R. Chi, Z. Hou, B. Huang, and S. Jin, *A unified data-driven design framework of optimality-based generalized iterative learning control*, Comput. Chem. Eng., 77 (2015), pp. 10–23.

[63] P. D. Christofides, R. Scattolini, D. Muñoz de la Peña, and J. Liu, *Distributed model predictive control: A tutorial review and future research directions*, Comput. Chem. Eng., 51 (2013), pp. 21–41.

[64] J. Clune, J.-B. Mouret, and H. Lipson, *The evolutionary origins of modularity*, Proc. R. Soc. B, 280 (2013), p. 20122863.

[65] B. Colson, P. Marcotte, and G. Savard, *An overview of bilevel optimization*, Ann. Oper. Res., 153 (2007), pp. 235–256.

[66] A. J. Conejo, E. Castillo, R. Minguez, and R. Garcia-Bertrand, *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*, Springer, 2006.

[67] A. J. Conejo, F. J. Nogales, and F. J. Prieto, *A decomposition procedure based on approximate newton directions*, Math. Prog., 93 (2002), pp. 495–515.

[68] P. H. Constantino and P. Daoutidis, *A control perspective on the evolution of biological modularity*, IFAC-PapersOnLine, 52 (2019), pp. 172–177. 5th IFAC Conference on Intelligent Control and Automation Sciences (ICONS).

[69] P. H. Constantino, W. Tang, and P. Daoutidis, *Topology effects on sparse control of complex networks with Laplacian dynamics*, Sci. Rep., 9 (2019), p. 9034.

[70] J. Currie and D. I. Wilson, *OPTI: Lowering the barrier between open source optimizers and the industrial MATLAB user*, in Foundations of Computer-Aided Process Operations, N. Sahinidis and J. Pinto, eds., Savannah, Georgia, USA, 8–11 January 2012.

[71] C. R. Cutler and B. L. Ramaker, *Dynamic matrix control – a computer control algorithm*, in Joint Autom. Control Conf., 1980, p. 72.

[72] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, and R. Pascanu, *Sobolev training for neural networks*, in Adv. Neural Inf. Process. Syst. (NIPS), 2017, pp. 4278–4287.

[73] R. D'Andrea and G. E. Dullerud, *Distributed control design for spatially interconnected systems*, IEEE Trans. Autom. Control, 48 (2003), pp. 1478–1495.

[74] P. Daoutidis and C. Kravaris, *Structural evaluation of control configurations for multivariable nonlinear processes*, Chem. Eng. Sci., 47 (1992), pp. 1091–1107.

[75] P. Daoutidis, M. Soroush, and C. Kravaris, *Feedforward/feedback control of multivariable nonlinear processes*, AIChE J., 36 (1990), pp. 1471–1484.

[76] P. Daoutidis, W. Tang, and A. Allman, *Decomposition of control and optimization problems by network structure: concepts, methods and inspirations from biology*, AIChE J., 65 (2019), p. e16708.

[77] P. Daoutidis, W. Tang, and S. S. Jogwar, *Decomposing complex plants for distributed control: Perspectives from network theory*, Comput. Chem. Eng., 114 (2018), pp. 43–51.

[78] J. Davis, T. Edgar, J. Porter, J. Bernaden, and M. Sarli, *Smart manufacturing, manufacturing intelligence and demand-dynamic performance*, Comput. Chem. Eng., 47 (2012), pp. 145–156.

[79] C. De Persis and P. Tesi, *Formulas for data-driven control: Stabilization, optimality and robustness*, IEEE Trans. Autom. Control, 65 (2020), pp. 909–924.

[80] B. De Schutter, *Minimal state-space realization in linear system theory: an overview*, J. Comput. Appl. Math., 121 (2000), pp. 331–354.

[81] J. Dean, G. S. Corrado, R. Monga, et al., *Large scale distributed deep networks*, in Proc. 25th Int. Conf. Neural Inf. Process. Syst. (NIPS), 2012, pp. 1223–1231.

[82] W. Deng, M.-J. Lai, Z. Peng, and W. Yin, *Parallel multi-block ADMM with $O(1/k)$ convergence*, J. Sci. Comput., 71 (2017), pp. 712–736.

[83] N. Dhingra, F. Lin, M. Fardad, and M. R. Jovanović, *On identifying sparse representations of consensus networks*, IFAC Proc. Vol., 45 (2012), pp. 305–310. 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys).

[84] N. K. Dhingra, S. Z. Khong, and M. R. Jovanović, *The proximal augmented Lagrangian method for nonsmooth composite optimization*, IEEE Trans. Autom. Control, 64 (2019), pp. 2861–2868.

[85] N. A. Diangelakis, B. Burnak, J. Katz, and E. N. Pistikopoulos, *Process design and control optimization: A simultaneous approach by multi-parametric programming*, AIChE J., 63 (2017), pp. 4827–4846.

[86] L. S. Dias and M. G. Ierapetritou, *From process control to supply chain management: An overview of integrated decision making strategies*, Comput. Chem. Eng., 106 (2017), pp. 826–835.

[87] M. Diehl, H. G. Bock, and J. P. Schlöder, *A real-time iteration scheme for nonlinear optimization in optimal feedback control*, SIAM J. Control Optim., 43 (2005), pp. 1714–1736.

[88] V. D. Dimitriadis and E. N. Pistikopoulos, *Flexibility analysis of dynamic systems*, Ind. Eng. Chem. Res., 34 (1995), pp. 4451–4462.

[89] R. Dobrin, Q. K. Beg, A.-L. Barabási, and Z. N. Oltvai, *Aggregation of topological motifs in the Escherichia coli transcriptional regulatory network*, BMC Bioinform., 5 (2004), p. 10.

[90] D. Dochain, *State and parameter estimation in chemical and biochemical processes: A tutorial*, J. Process Control, 13 (2003), pp. 801–818.

[91] F. J. Doyle III, B. A. Ogunnaike, and R. K. Pearson, *Nonlinear model-based control using second-order Volterra models*, Automatica, 31 (1995), pp. 697–714.

[92] A. Dulmage and N. Mendelsohn, *Two algorithms for bipartite graphs*, J. SIAM, 11 (1963), pp. 183–194.

[93] I. Dunning, J. Huchette, and M. Lubin, *JuMP: A modeling language for mathematical optimization*, SIAM Rev., 59 (2017), pp. 295–320.

[94] J. Eckstein and D. P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Prog., 55 (1992), pp. 293–318.

[95] J. Eckstein and W. Yao, *Approximate ADMM algorithms derived from Lagrangian splitting*, Comput. Optim. Appl., 68 (2017), pp. 363–405.

[96] N. H. El-Farra and P. D. Christofides, *Integrating robustness, optimality and constraints in control of nonlinear processes*, Chem. Eng. Sci., 56 (2001), pp. 1841–1868.

[97] M. Ellis and P. D. Christofides, *Selection of control configurations for economic model predictive control systems*, AIChE J., 60 (2014), pp. 3230–3242.

[98] M. Ellis, J. Liu, and P. D. Christofides, *Economic Model Predictive Control*, Springer, 2016.

[99] P. Englezos and N. Kalogerakis, *Applied Parameter Estimation for Chemical Engineers*, CRC Press, 2000.

[100] W. R. Esposito and C. A. Floudas, *Deterministic global optimization in nonlinear optimal control problems*, J. Global Optim., 17 (2000), pp. 97–126.

[101] G. A. Ezhilarasi and K. S. Swarup, *Network partitioning using harmony search and equivalencing for distributed computing*, J. Parallel Distrib. Comput., 72 (2012), pp. 936–943.

[102] H.-r. Fang and Y. Saad, *Two classes of multisecant methods for nonlinear acceleration*, Numer. Linear Algebra Appl., 16 (2009), pp. 197–221.

[103] H. Farhangi, *The path of the smart grid*, IEEE Power Energy Mag., 8 (2010), pp. 18–28.

[104] M. Farina and R. Scattolini, *Distributed predictive control: a non-cooperative algorithm with neighbor-to-neighbor communication for linear systems*, Automatica, 48 (2012), pp. 1088–1096.

[105] F. Farokhi, I. Shames, and K. H. Johansson, *Distributed MPC via dual decomposition and alternative direction method of multipliers*, in Distributed model predictive control made easy, Springer, 2014, pp. 115–131.

[106] J. A. Farrell and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*, John Wiley & Sons, 2006.

[107] M. Farza, M. M'Saad, T. Maatoug, and M. Kamoun, *Adaptive observers for nonlinearly parameterized class of nonlinear systems*, Automatica, 45 (2009), pp. 2292–2299.

[108] W. Favoreel, B. De Moor, and P. Van Overschee, *Subspace state space system identification for industrial processes*, J. Process Control, 10 (2000), pp. 149–155.

[109] A. Ferramosca, D. Limón, I. Alvarado, and E. F. Camacho, *Cooperative distributed MPC for tracking*, Automatica, 49 (2013), pp. 906–914.

[110] A. V. Fiacco, *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, Elsevier, 1983.

[111] N. M. Filatov and H. Unbehauen, *Adaptive Dual Control: Theory and Applications*, Springer, 2004.

[112] A. Flores-Tlacuahuac and L. T. Biegler, *Simultaneous mixed-integer dynamic optimization for integrated design and control*, Comput. Chem. Eng., 31 (2007), pp. 588–600.

[113] C. A. Floudas, *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*, Oxford University Press, 1995.

[114] S. Fortunato, *Community detection in graphs*, Phys. Rep., 486 (2010), pp. 75–174.

[115] S. Fortunato and D. Hric, *Community detection in networks: A user guide*, Phys. Rep., 659 (2016), pp. 1–44.

[116] A. S. Foss, *Critique of chemical process control theory*, AIChE J., 19 (1973), pp. 209–214.

[117] A. L. Fradkov, *A scheme of speed gradient and its application in problems of adaptive control*, Avtom. Telemekh., (1979), pp. 90–101. (in Russian).

[118] A. Fu, J. Zhang, and S. Boyd, *Anderson accelerated Douglas-Rachford splitting*, arXiv preprint arXiv:1908.11482, (2019).

[119] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Comput. Math. Appl., 2 (1976), pp. 17–40.

[120] J. P. Gagnepain and D. E. Seborg, *Analysis of process interactions with applications to multiloop control system design*, Ind. Chem. Process Des. Dev., 21 (1982), pp. 5–11.

[121] C. E. Garcia, D. M. Prett, and M. Morari, *Model predictive control: theory and practice – a survey*, Automatica, 25 (1989), pp. 335–348.

[122] D. J. Garcia and F. You, *Supply chain design and optimization: Challenges and opportunities*, Comput. Chem. Eng., 81 (2015), pp. 153–170.

[123] J. P. García-Sandoval, N. Hudon, D. Dochain, and V. Gonzalez-Alvarez, *Stability analysis and passivity properties of a class of thermodynamic processes: An internal entropy production approach*, Chem. Eng. Sci., 139 (2016), pp. 261–272.

[124] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, *A unified framework for the numerical solution of optimal control problems using pseudospectral methods*, Automatica, 46 (2010), pp. 1843–1851.

[125] S. Ghadimi and G. Lan, *Accelerated gradient methods for nonconvex nonlinear and stochastic programming*, Math. Prog., 156 (2016), pp. 59–99.

[126] M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar, *Bayesian reinforcement learning: A survey*, Found. Trends Mach. Learn., 8 (2015), pp. 359–483.

[127] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, Proc. Natl. Acad. Sci. U.S.A., 99 (2002), pp. 7821–7826.

[128] P. Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer, *Accelerated gradient methods and dual decomposition in distributed model predictive control*, Automatica, 49 (2013), pp. 829–833.

[129] R. Glowinski and A. Marroco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires*, Rev. Fr. Autom. Inform. Rech. Opér., Anal. Numér., 9 (1975), pp. 41–76.

[130] V. Goel and I. E. Grossmann, *A class of stochastic programs with decision dependent uncertainty*, Math. Prog., 108 (2006), pp. 355–394.

[131] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, *Fast alternating direction optimization methods*, SIAM J. Imaging Sci., 7 (2014), pp. 1588–1623.

[132] A. Gosavi, *Simulation-Based Optimization*, Springer, 2015.

[133] M. Grant and S. Boyd, *CVX: Matlab software for disciplined convex programming, version 2.1.* `http://cvxr.com/cvx`, Mar. 2014.

[134] P. Grosdidier and M. Morari, *Interaction measures for systems under decentralized control*, Automatica, 22 (1986), pp. 309–319.

[135] I. E. Grossmann, *Enterprise-wide optimization: A new frontier in process systems engineering*, AIChE J., 51 (2005), pp. 1846–1857.

[136] ——, *Advances in mathematical programming models for enterprise-wide optimization*, Comput. Chem. Eng., 47 (2012), pp. 2–18.

[137] I. E. Grossmann, B. A. Calfa, and P. Garcia-Herreros, *Evolution of concepts and models for quantifying resiliency and flexibility of chemical processes*, Comput. Chem. Eng., 70 (2014), pp. 22–34.

[138] I. E. Grossmann and C. A. Floudas, *Active constraint strategy for flexibility analysis in chemical processes*, Comput. Chem. Eng., 11 (1987), pp. 675–693.

[139] I. E. Grossmann and F. Trespalacios, *Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming*, AIChE J., 59 (2013), pp. 3276–3295.

[140] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control*, Springer, 2017.

[141] Z. H. Gümüş and C. A. Floudas, *Global optimization of nonlinear bilevel programming problems*, J. Global Optim., 20 (2001), pp. 1–31.

[142] D. Gupta and C. T. Maravelias, *On deterministic online scheduling: major considerations, paradoxes and remedies*, Comput. Chem. Eng., 94 (2016), pp. 312–330.

[143] R. A. Gupta and M.-Y. Chow, *Networked control system: overview and research trends*, IEEE Trans. Ind. Electron., 57 (2010), pp. 2527–2535.

[144] D. Hajinezhad and M. Hong, *Perturbed proximal primal-dual algorithm for nonconvex nonsmooth optimization*, Math. Prog., 176 (2019), pp. 207–245.

[145] K. M. Hangos, J. Bokor, and G. Szederkényi, *Hamiltonian view on process systems*, AIChE J., 47 (2001), pp. 1819–1831.

[146] K. M. Hangos and Z. Tuza, *Optimal control structure selection for process systems*, Comput. Chem. Eng., 25 (2001), pp. 1521–1536.

[147] W. E. Hart, C. Laird, J.-P. Watson, and D. L. Woodruff, *Pyomo – Optimization Modeling in Python*, Springer, 2012.

[148] B. He and X. Yuan, *On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method*, SIAM J. Numer. Anal., 50 (2012), pp. 700–709.

[149] M.-J. He, W.-J. Cai, W. Ni, and L.-H. Xie, *RNGA based control system configuration for multivariable processes*, J. Process Control, 19 (2009), pp. 1036–1042.

[150] M. Heidarinejad, J. Liu, and P. D. Christofides, *Economic model predictive control of nonlinear process systems using Lyapunov techniques*, AIChE J., 58 (2012), pp. 855–870.

[151] T. A. N. Heirung, B. E. Ydstie, and B. Foss, *Dual adaptive model predictive control*, Automatica, 80 (2017), pp. 340–348.

[152] S. Heo and P. Daoutidis, *Control-relevant decomposition of process networks via optimization-based hierarchical clustering*, AIChE J., 62 (2016), pp. 3177–3188.

[153] S. Heo, W. A. Marvin, and P. Daoutidis, *Automated synthesis of control configurations for process networks based on structural coupling*, Chem. Eng. Sci., 136 (2015), pp. 76–87.

[154] D. J. Hill and P. J. Moylan, *The stability of nonlinear dissipative systems*, IEEE Trans. Autom. Control, 21 (1976), pp. 708–711.

[155] ——, *Dissipative dynamical systems: Basic input-output and state properties*, J. Franklin Inst., 309 (1980), pp. 327–357.

[156] D. Himmelblau, *Decomposition of large scale systems – I. systems composed of lumped parameter elements*, Chem. Eng. Sci., 21 (1966), pp. 425–438.

[157] D. Hioe, J. Bao, and B. E. Ydstie, *Dissipativity analysis for networks of process systems*, Comput. Chem. Eng., 50 (2013), pp. 207–219.

[158] H. Hjalmarsson, *Iterative feedback tuning – an overview*, Int. J. Adapt. Control Signal Process., 16 (2002), pp. 373–395.

[159] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, *Iterative feedback tuning: theory and applications*, IEEE Control Syst. Mag., 18 (1998), pp. 26–41.

[160] T. Homer and P. Mhaskar, *Utilizing null controllable regions to stabilize input-constrained nonlinear systems*, Comput. Chem. Eng., 108 (2018), pp. 24–30.

[161] M. Hong and Z.-Q. Luo, *On the linear convergence of the alternating direction method of multipliers*, Math. Prog., 162 (2017), pp. 165–199.

[162] M. Hong, Z.-Q. Luo, and M. Razaviyayn, *Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems*, SIAM J. Optim., 26 (2016), pp. 337–364.

[163] Z. Hou and S. Jin, *Model Free Adaptive Control: Theory and Applications*, CRC Press, 2013.

[164] Z. Hou and Z. Wang, *From model-based control to data-driven control: survey, classification and perspective*, Inf. Sci., 235 (2013), pp. 3–35.

[165] J.-H. Hours and C. N. Jones, *A parametric nonconvex decomposition algorithm for real-time and distributed NMPC*, IEEE Trans. Autom. Control, 61 (2015), pp. 287–302.

[166] ——, *A parametric nonconvex decomposition algorithm for real-time and distributed NMPC*, IEEE Trans. Autom. Control, 61 (2016), pp. 287–302.

[167] B. Houska, J. Frasch, and M. Diehl, *An augmented Lagrangian based algorithm for distributed nonconvex optimization*, SIAM J. Optim., 26 (2016), pp. 1101–1127.

[168] M. Hovd and S. Skogestad, *Pairing criteria for decentralized control of unstable plants*, Ind. Eng. Chem. Res., 33 (1994), pp. 2134–2139.

[169] W. Hu, W.-J. Cai, and G. Xiao, *Decentralized control system design for mimo processes with integrators/differentiators*, Ind. Eng. Chem. Res., 49 (2010), pp. 12521–12528.

[170] B. Huang and R. Kadali, *Dynamic Modeling, Predictive Control and Performance Monitoring: A Data-Driven Subspace Approach*, Springer, 2008.

[171] A. Hyvärinen and E. Oja, *Independent component analysis: algorithms and applications*, Neural Netw., 13 (2000), pp. 411–430.

[172] P. A. Iglesias and B. P. Ingalls, *Control Theory and Systems Biology*, MIT Press, 2010.

[173] A. Isidori, *Nonlinear Control Systems*, Springer, 3$^{\text{rd}}$ ed., 2013.

[174] D. Jakovetić, J. Xavier, and J. M. F. Moura, *Fast distributed gradient methods*, IEEE Trans. Automat. Control, 59 (2014), pp. 1131–1146.

[175] Z. W. Jarvis-Wloszek, *Lyapunov based analysis and controller synthesis for polynomial systems using sum-of-squares optimization*, PhD thesis, University of California, Berkeley, 2003.

[176] L. Ji, J. B. Rawlings, W. Hu, A. Wynn, and M. Diehl, *Robust stability of moving horizon estimation under bounded disturbances*, IEEE Trans. Autom. Control, 61 (2015), pp. 3509–3514.

[177] B. Jiang, T. Lin, S. Ma, and S. Zhang, *Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis*, Comput. Optim. Appl., 72 (2019), pp. 115–157.

[178] Y. Jiang and Z.-P. Jiang, *Global adaptive dynamic programming for continuous-time nonlinear systems*, IEEE Trans. Autom. Control, 60 (2015), pp. 2917–2929.

[179] S. S. Jogwar, *Distributed control architecture synthesis for integrated process networks through maximization of strength of input–output impact*, J. Process Control, 83 (2019), pp. 77–87.

[180] S. S. Jogwar and P. Daoutidis, *Community-based synthesis of distributed control architectures for integrated process networks*, Chem. Eng. Sci., 172 (2017), pp. 434–443.

[181] K. H. Johansson, *The quadruple-tank process: A multivariable laboratory process with an adjustable zero*, IEEE Trans. Control Syst. Technol., 8 (2000), pp. 456–465.

[182] R. D. Johnston, *Steady-state closed-loop structural interaction analysis*, Int. J. Control, 52 (1990), pp. 1351–1369.

[183] M. R. Jovanović and N. K. Dhingra, *Controller architectures: Tradeoffs between performance and structure*, Eur. J. Control, 30 (2016), pp. 76–91.

[184] M. P. Joy, A. Brock, D. E. Ingber, and S. Huang, *High-betweenness proteins in the yeast protein interaction network*, J. Biomed. Biotechnol., 2005 (2005), pp. 96–103.

[185] M. Kadkhodaie, K. Christakopoulou, M. Sanjabi, and A. Banerjee, *Accelerated alternating direction method of multipliers*, in Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discover. Data Min., ACM, 2015, pp. 497–506.

[186] S. Kamelian and K. Salahshoor, *A novel graph-based partitioning algorithm for large-scale dynamical systems*, Int. J. Syst. Sci., 46 (2015), pp. 227–245.

[187] J. Kang, Y. Cao, D. P. Word, and C. D. Laird, *An interior-point method for efficient solution of block-structured NLP problems using an implicit Schur-complement decomposition*, Comput. Chem. Eng., 71 (2014), pp. 563–573.

[188] L. Kang, W. Tang, Y. Liu, and P. Daoutidis, *Control configuration synthesis using agglomerative hierarchical clustering: A graph-theoretic approach*, J. Process Control, 46 (2016), pp. 43–54.

[189] V. Kariwala, J. F. Forbes, and E. S. Meadows, *Block relative gain: Properties and pairing rules*, Ind. Eng. Chem. Res., 42 (2003), pp. 4564–4574.

[190] N. Kashtan and U. Alon, *Spontaneous evolution of modularity and network motifs*, Proc. Natl. Acad. Sci. U.S.A., 102 (2005), pp. 13773–13778.

[191] N. Kazantzis and C. Kravaris, *Nonlinear observer design using Lyapunov's auxiliary theorem*, Syst. Control Lett., 34 (1998), pp. 241–247.

[192] A. Khaki-Sedigh and B. Moaveni, *Control Configuration Selection for Multivariable Plants*, Springer, 2009.

[193] H. K. Khalil, *Nonlinear Systems*, Pearson, 3$^{\text{rd}}$ ed., 2002.

[194] K.-D. Kim and P. R. Kumar, *Cyber–physical systems: A perspective at the centennial*, Proc. IEEE, 100 (2012), pp. 1287–1308.

[195] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, arXiv preprint, (2014). arXiv:1412.6980.

[196] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing*, Science, 220 (1983), pp. 671–680.

[197] M. R. Kleinberg, K. Miu, N. Segal, H. Lehmann, and T. R. Figura, *A partitioning method for distributed capacitor control of electric power distribution systems*, IEEE Trans. Power Syst., 29 (2014), pp. 637–644.

[198] P. Kokotović and M. Arcak, *Constructive nonlinear control: a historical perspective*, Automatica, 37 (2001), pp. 637–662.

[199] I. K. Kookos and A. I. Lygeros, *An algorithmic method for control structure selection based on the RGA and RIA interaction measures*, Chem. Eng. Res. Des., 76 (1998), pp. 458–464.

[200] I. K. Kookos and J. D. Perkins, *An algorithm for simultaneous process design and control*, Ind. Eng. Chem. Res., 40 (2001), pp. 4079–4088.

[201] ——, *Heuristic-based mathematical programming framework for control structure selection*, Ind. Eng. Chem. Res., 40 (2001), pp. 2079–2088.

[202] M. Korda and I. Mezić, *Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control*, Automatica, 93 (2018), pp. 149–160.

[203] C. Kravaris, J. Hahn, and Y. Chu, *Advances and selected recent developments in state and parameter estimation*, Comput. Chem. Eng., 51 (2013), pp. 111–123.

[204] C. Kravaris and J. C. Kantor, *Geometric methods for nonlinear process control. 1. background*, Ind. Eng. Chem. Res., 29 (1990), pp. 2295–2310.

[205] ——, *Geometric methods for nonlinear process control. 2. controller synthesis*, Ind. Eng. Chem. Res., 29 (1990), pp. 2310–2323.

[206] A. Kreimer, E. Borenstein, U. Gophna, and E. Ruppin, *The evolution of modularity in bacterial metabolic networks*, Proc. Natl. Acad. Sci. U.S.A., 105 (2008), pp. 6976–6981.

[207] M. Krstić, P. V. Kokotović, and I. Kanellakopoulos, *Nonlinear and adaptive control design*, John Wiley & Sons, 1995.

[208] P. Kühl, M. Diehl, T. Kraus, J. P. Schlöder, and H. G. Bock, *A real-time algorithm for moving horizon state and parameter estimation*, Comput. Chem. Eng., 35 (2011), pp. 71–83.

[209] A. Kumar and P. Daoutidis, *Feedback control of nonlinear differential-algebraic-equation systems*, AIChE J., 41 (1995), pp. 619–636.

[210] C. Langbort, R. S. Chandra, and R. D'Andrea, *Distributed control design for systems interconnected over an arbitrary graph*, IEEE Trans. Autom. Control, 49 (2004), pp. 1502–1519.

[211] L. Lao, M. Ellis, and P. D. Christofides, *Economic model predictive control of transport-reaction processes*, Ind. Eng. Chem. Res., 53 (2013), pp. 7382–7396.

[212] J. Lee and T. F. Edgar, *Dynamic interaction measures for decentralized control of multivariable processes*, Ind. Eng. Chem. Res., 43 (2004), pp. 283–287.

[213] J. H. Lee and J. M. Lee, *Approximate dynamic programming based approach to process control and scheduling*, Comput. Chem. Eng., 30 (2006), pp. 1603–1618.

[214] J. H. Lee, J. Shin, and M. J. Realff, *Machine learning: Overview of the recent progresses and implications for the process systems engineering field*, Comput. Chem. Eng., 114 (2018), pp. 111–121.

[215] J. H. Lee and W. Wong, *Approximate dynamic programming approach for process control*, J. Process Control, 20 (2010), pp. 1038–1048.

[216] J. M. Lee and J. H. Lee, *Approximate dynamic programming-based approaches for input–output data-driven control of nonlinear processes*, Automatica, 41 (2005), pp. 1281–1288.

[217] A. M. Lenhoff and M. Morari, *Design of resilient processing plants—i process design under consideration of dynamic aspects*, Chem. Eng. Sci., 37 (1982), pp. 245–258.

[218] C. Leopold, *Parallel and Distributed Computing: A Survey of Models, Paradigms and Approaches*, John Wiley & Sons, 2001.

[219] L. Lessard, B. Recht, and A. Packard, *Analysis and design of optimization algorithms via integral quadratic constraints*, SIAM J. Optim., 26 (2016), pp. 57–95.

[220] A. Levant, *Higher-order sliding modes, differentiation and output-feedback control*, Int. J. Control, 76 (2003), pp. 924–941.

[221] G. Li and T. K. Pong, *Global convergence of splitting methods for nonconvex composite optimization*, SIAM J. Optim., 25 (2015), pp. 2434–2460.

[222] Y. Li and Z. Hou, *Data-driven asymptotic stabilization for discrete-time nonlinear systems*, Syst. Control Lett., 64 (2014), pp. 79–85.

[223] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, *Continuous control with deep reinforcement learning*, arXiv preprint, (2015). arXiv:1509.02971.

[224] F. V. Lima, Z. Jia, M. Ierapetritou, and C. Georgakis, *Similarities and differences between the concepts of operability and flexibility: the steady-state case*, AIChE J., 56 (2010), pp. 702–716.

[225] D. Limón, T. Alamo, F. Salas, and E. F. Camacho, *On the stability of constrained MPC without terminal constraint*, IEEE Trans. Autom. Control, 51 (2006), pp. 832–836.

[226] F. Lin, M. Fardad, and M. R. Jovanović, *Augmented Lagrangian approach to design of structured optimal state feedback gains*, IEEE Trans. Autom. Control, 56 (2011), pp. 2923–2929.

[227] ——, *Identification of sparse communication graphs in consensus networks*, in 50th Ann. Allerton Conf. Commun. Control Comput., IEEE, 2012, pp. 85–89.

[228] ——, *Design of optimal sparse feedback gains via the alternating direction method of multipliers*, IEEE Trans. Autom. Control, 58 (2013), pp. 2426–2431.

[229] T. Lin, S. Ma, and S. Zhang, *On the global linear convergence of the admm with multiblock variables*, SIAM J. Optim., 25 (2015), pp. 1478–1497.

[230] Y. Lin and E. D. Sontag, *A universal formula for stabilization with bounded controls*, Syst. Control Lett., 16 (1991), pp. 393–397.

[231] D. Liu, Q. Wei, D. Wang, X. Yang, and H. Li, *Adaptive Dynamic Programming with Applications in Optimal Control*, Springer, 2017.

[232] J. Liu, X. Chen, D. Muñoz de la Peña, and P. D. Christofides, *Sequential and iterative architectures for distributed model predictive control of nonlinear process systems*, AIChE J., 56 (2010), pp. 2137–2149.

[233] J. Liu, P. Daoutidis, and B. Yang, *Process design and optimization for etherification of glycerol with isobutene*, Chem. Eng. Sci., 144 (2016), pp. 326–335.

[234] J. Liu, D. Muñoz de la Peña, and P. D. Christofides, *Distributed model predictive control of nonlinear process systems*, AIChE J., 55 (2009), pp. 1171–1184.

[235] T.-Y. Liu, W. Chen, and T. Wang, *Recent advances in distributed machine learning*, tutorial, AAAI, February 2017.

[236] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, *Controllability of complex networks*, Nature, 473 (2011), pp. 167–173.

[237] ——, *Control centrality and hierarchical structure in complex networks*, PLoS One, 7 (2012), p. e44459.

[238] L. Ljung, *System Identification: Theory for the User*, Prentice Hall, 1999.

[239] R. Lozano, B. Brogliato, O. Egeland, and B. Maschke, *Dissipative Systems Analysis and Control: Theory and Applications*, Springer, 2013.

[240] S. Lucia and B. Karg, *A deep learning-based approach to robust nonlinear model predictive control*, IFAC-PapersOnLine, 51 (2018), pp. 511–516.

[241] B. Luo, H.-N. Wu, T. Huang, and D. Liu, *Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design*, Automatica, 50 (2014), pp. 3281–3290.

[242] M. L. Luyben and C. A. Floudas, *Analyzing the interaction of design and control – 1. a multiobjective framework and application to binary distillation synthesis*, Comput. Chem. Eng., 18 (1994), pp. 933–969.

[243] M. L. Luyben, B. D. Tyreus, and W. L. Luyben, *Plantwide control design procedure*, AIChE J., 43 (1997), pp. 3161–3174.

[244] S. Magnússon, P. C. Weeraddana, M. G. Rabbat, and C. Fischione, *On the convergence of alternating direction Lagrangian methods for nonconvex structured optimization problems*, IEEE Trans. Control Netw. Syst., 3 (2015), pp. 296–309.

[245] A. Makhdoumi and A. Ozdaglar, *Convergence rate of distributed ADMM over networks*, IEEE Trans. Autom. Control, 62 (2017), pp. 5082–5095.

[246] A. Malcolm, J. Polan, L. Zhang, B. A. Ogunnaike, and A. A. Linninger, *Integrating systems design and control using dynamic flexibility analysis*, AIChE J., 53 (2007), pp. 2048–2061.

[247] M. Malisoff and F. Mazenc, *Constructions of Strict Lyapunov Functions*, Springer, 2009.

[248] V. Manousiouthakis, R. Savage, and Y. Arkun, *Synthesis of decentralized process control structures using the concept of block relative gain*, AIChE J., 32 (1986), pp. 991–1003.

[249] Y. Mao, M. Szmuk, and B. Açikmeşe, *Successive convexification of non-convex optimal control problems and its convergence properties*, in Proceedings of the 55th IEEE Conference on Decision and Control (CDC), IEEE, 2016, pp. 3636–3641.

[250] C. T. Maravelias and C. Sung, *Integration of production planning and scheduling: Overview, challenges and opportunities*, Comput. Chem. Eng., 33 (2009), pp. 1919–1930.

[251] I. Markovsky and P. Rapisarda, *Data-driven simulation and control*, Int. J. Control, 81 (2008), pp. 1946–1959.

[252] S. J. Mason, *Feedback theory – some properties of signal flow graphs*, Proc. IRE, 41 (1953), pp. 1144–1156.

[253] T. Maupong, J. C. Mayo-Maldonado, and P. Rapisarda, *On Lyapunov functions and data-driven dissipativity*, IFAC-PapersOnLine, 50 (2017), pp. 7783–7788. 20th IFAC World Congress.

[254] D. Q. Mayne, *Model predictive control: Recent developments and future promise*, Automatica, 50 (2014), pp. 2967–2986.

[255] D. Q. Mayne and P. Falugi, *Stabilizing conditions for model predictive control*, Int. J. Robust Nonlin. Control, 29 (2019), pp. 894–903.

[256] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, *Constrained model predictive control: Stability and optimality*, Automatica, 36 (2000), pp. 789–814.

[257] A. McAfee and E. Brynjolfsson, *Big data: the management revolution*, Harv. Bus. Rev., 90 (2012), pp. 60–68.

[258] T. J. McAvoy, *Interaction Analysis: Principles and Applications*, ISA, 1983.

[259] D. Melo and G. Marroig, *Directional selection can drive the evolution of modularity in complex traits*, Proc. Natl. Acad. Sci. U.S.A., 112 (2015), pp. 470–475.

[260] H. Mengistu, J. Huizinga, J.-B. Mouret, and J. Clune, *The evolutionary origins of hierarchy*, PLoS Comput. Biol., 12 (2016), p. e1004829.

[261] A. Mesbah, *Stochastic model predictive control with active uncertainty learning: A survey on dual control*, Annu. Rev. Control, 45 (2018), pp. 107–117.

[262] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*, Princeton University Press, 2010.

[263] P. Mhaskar, N. H. El-Farra, and P. D. Christofides, *Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control*, Syst. Control Lett., 55 (2006), pp. 650–659.

[264] P. Mhaskar, J. Liu, and P. D. Christofides, *Fault-Tolerant Process Control: Methods and Applications*, Springer, 2012.

[265] A. Michel, R. Miller, and W. Tang, *Lyapunov stability of interconnected systems: Decomposition into strongly connected subsystems*, IEEE Trans. Circuits Syst., 25 (1978), pp. 799–809.

[266] A. Mitsos, P. Lemonidis, and P. I. Barton, *Global solution of bilevel programs with a nonconvex inner program*, J. Global Optim., 42 (2008), pp. 475–513.

[267] M. Moharir, L. Kang, P. Daoutidis, and A. Almansoori, *Graph representation and decomposition of ODE/hyperbolic PDE systems*, Comput. Chem. Eng., 106 (2017), pp. 532–543.

[268] M. Moharir, D. B. Pourkargar, A. Almansoori, and P. Daoutidis, *Distributed model predictive control of an amine gas sweetening plant*, Ind. Eng. Chem. Res., 57 (2018), pp. 13103–13115.

[269] ———, *Graph representation and distributed control of diffusion-convection-reaction system networks*, Chem. Eng. Sci., 219 (2019), pp. 128–139.

[270] M. J. Mohideen, J. D. Perkins, and E. N. Pistikopoulos, *Optimal design of dynamic systems under uncertainty*, AIChE J., 42 (1996), pp. 2251–2272.

[271] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, *A survey of distributed optimization and control algorithms for electric power systems*, IEEE Trans. Smart Grid, 8 (2017), pp. 2941–2962.

[272] N. Monshizadeh-Naini, A. Fatehi, and A. Khaki-Sedigh, *Input-output pairing using effective relative energy array*, Ind. Eng. Chem. Res., 48 (2009), pp. 7137–7144.

[273] C. Moore and M. E. Newman, *Epidemics and percolation in small-world networks*, Phys. Rev. E, 61 (2000), p. 5678.

[274] M. Morari and J. H. Lee, *Model predictive control: past, present and future*, Comput. Chem. Eng., 23 (1999), pp. 667–682.

[275] M. Morari and E. Zafiriou, *Robust Process Control*, Prentice Hall, 1989.

[276] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, *Distributed optimization with local domains: applications in MPC and network flows*, IEEE Trans. Autom. Control, 60 (2014), pp. 2004–2009.

[277] N. Motee and A. Jadbabaie, *Optimal control of spatially distributed systems*, IEEE Trans. Autom. Control, 53 (2008), pp. 1616–1629.

[278] P. Moylan and D. Hill, *Stability criteria for large-scale systems*, IEEE Trans. Autom. Control, 23 (1978), pp. 143–149.

[279] C. Mu, Z. Ni, C. Sun, and H. He, *Data-driven tracking control with adaptive dynamic programming for a class of continuous-time nonlinear systems*, IEEE Trans. Cybern., 47 (2017), pp. 1460–1470.

[280] M. A. Müller, *Nonlinear moving horizon estimation in the presence of bounded disturbances*, Automatica, 79 (2017), pp. 306–314.

[281] M. A. Müller, D. Angeli, and F. Allgöwer, *On necessity and robustness of dissipativity in economic model predictive control*, IEEE Trans. Autom. Control, 60 (2015), pp. 1671–1676.

[282] A. Narasingam and J. S.-I. Kwon, *Data-driven identification of interpretable reduced-order models using sparse regression*, Comput. Chem. Eng., 119 (2018), pp. 101–111.

[283] ——, *Koopman Lyapunov-based model predictive control of nonlinear chemical process systems*, AIChE J., 65 (2019), p. e16743.

[284] M. Nayeripour, H. Fallahzadeh-Abarghouei, E. Waffenschmidt, and S. Hasanvand, *Coordinated online voltage management of distributed generation using network partitioning*, Electr. Power Syst. Res., 141 (2016), pp. 202–209.

[285] I. Necoara and D. Clipici, *Efficient parallel coordinate descent algorithm for convex optimization problems with separable constraints: application to distributed MPC*, J. Process Control, 23 (2013), pp. 243–253.

[286] A. Nedić and A. Olshevsky, *Distributed optimization over time-varying directed graphs*, IEEE Trans. Autom. Control, 60 (2014), pp. 601–615.

[287] A. Nedić and A. Ozdaglar, *Distributed subgradient methods for multi-agent optimization*, IEEE Trans. Automat. Control, 54 (2009), pp. 48–61.

[288] R. R. Negenborn and J. Maestre, *Distributed model predictive control: An overview and roadmap of future research opportunities*, IEEE Control Syst., 34 (2014), pp. 87–97.

[289] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, *Robust stochastic approximation approach to stochastic programming*, SIAM J. Optim., 19 (2009), pp. 1574–1609.

[290] Yu. E. Nesterov, *A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$*, Dokl. Akad. Nauk SSSR, 269 (1983), pp. 543–547.

[291] M. E. J. Newman, *Assortative mixing in networks*, Phys. Rev. Lett., 89 (2002), p. 208701.

[292] ——, *Analysis of weighted networks*, Phys. Rev. E, 70 (2004), p. 056131.

[293] ——, *Modularity and community structure in networks*, Proc. Natl. Acad. Sci., 103 (2006), pp. 8577–8582.

[294] ——, *Communities, modules and large-scale structure in networks*, Nature Phys., 8 (2012), pp. 25–31.

[295] ——, *Networks*, Oxford University Press, 2nd ed., 2018.

[296] M. E. J. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E, 69 (2004), p. 026113.

[297] B. Nicholson, J. D. Siirola, J.-P. Watson, V. M. Zavala, and L. T. Biegler, `pyomo.dae`*: a modeling and automatic discretization framework for optimization with differential and algebraic equations*, Math. Prog. Comput., 10 (2018), pp. 187–223.

[298] A. Niederliński, *A heuristic approach to the design of linear multivariable interacting control systems*, Automatica, 7 (1971), pp. 691–701.

[299] C. Ning and F. You, *Data-driven decision making under uncertainty integrating robust optimization with principal component analysis and kernel smoothing methods*, Comput. Chem. Eng., 112 (2018), pp. 190–210.

[300] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, 2006.

[301] C. Novara and S. Formentin, *Data-driven inversion-based control of nonlinear systems with guaranteed closed-loop stability*, IEEE Trans. Autom. Control, 63 (2018), pp. 1147–1154.

[302] C. Novara, A. Nicolì, and G. C. Calafiore, *Nonlinear system identification in Sobolev spaces*, arXiv preprint, (2019). arXiv:1911.02930.

[303] C. Ocampo-Martínez, S. Bovo, and V. Puig, *Partitioning approach oriented to the decentralised predictive control of large-scale systems*, J. Process Control, 21 (2011), pp. 775–786.

[304] C. Ocampo-Martínez and V. Puig, *Partitioning Approaches for Large-Scale Water Transport Networks*, Springer, 2017, pp. 321–339.

[305] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, *Conic optimization via operator splitting and homogeneous self-dual embedding*, J. Optim. Theor. Appl., 169 (2016), pp. 1042–1068.

[306] K.-K. Oh, M.-C. Park, and H.-S. Ahn, *A survey of multi-agent formation control*, Automatica, 53 (2015), pp. 424–440.

[307] R. Olfati-Saber, J. A. Fax, and R. M. Murray, *Consensus and cooperation in networked multi-agent systems*, Proc. IEEE, 95 (2007), pp. 215–233.

[308] R. Olfati-Saber and R. M. Murray, *Consensus problems in networks of agents with switching topology and time-delays*, IEEE Trans. Autom. Control, 49 (2004), pp. 1520–1533.

[309] A. Olshevsky, *Minimal controllability problems*, IEEE Trans. Control Netw. Syst., 1 (2014), pp. 249–258.

[310] Y. Ouyang, Y. Chen, G. Lan, and E. Pasiliao Jr, *An accelerated linearized alternating direction method of multipliers*, SIAM J. Imaging Sci., 8 (2015), pp. 644–681.

[311] F. Özgülşen, R. A. Adomaitis, and A. Çinar, *A numerical method for determining optimal parameter values in forced periodic operation*, Chem. Eng. Sci., 47 (1992), pp. 605–613.

[312] K. P. Papalexandri and E. N. Pistikopoulos, *A multiperiod MINLP model for the synthesis of flexible heat and mass exchange networks*, Comput. Chem. Eng., 18 (1994), pp. 1125–1139.

[313] N. Parikh and S. Boyd, *Proximal algorithms*, Found. Trends Optim., 1 (2014), pp. 127–239.

[314] E. Parzen, *On estimation of a probability density function and mode*, Ann. Math. Stat., 33 (1962), pp. 1065–1076.

[315] F. Pasqualetti, S. Zampieri, and F. Bullo, *Controllability metrics, limitations and algorithms for complex networks*, IEEE Trans. Control Netw. Syst., 1 (2014), pp. 40–52.

[316] M. A. Patterson and A. V. Rao, *GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using h_p-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming*, ACM Trans. Math. Softw. (TOMS), 41 (2014), pp. 1–37.

[317] R. C. Pattison, C. R. Touretzky, I. Harjunkoski, and M. Baldea, *Moving horizon closed-loop production scheduling using dynamic process models*, AIChE J., 63 (2017), pp. 639–651.

[318] G. Paul, S. Sreenivasan, and H. E. Stanley, *Resilience of complex networks to random breakdown*, Phys. Rev. E, 72 (2005), p. 056130.

[319] K. Pavlikov and S. Uryasev, *CVaR norm and applications in optimization*, Optim. Lett., 8 (2014), pp. 1999–2020.

[320] T. Pho and L. Lapidus, *Topics in computer-aided design: Part I. an optimum tearing algorithm for recycle systems*, AIChE J., 19 (1973), pp. 1170–1181.

[321] V. Pichai, M. E. Sezer, and D. D. Šiljak, *A graph-theoretic algorithm for hierarchical decomposition of dynamic systems with applications to estimation and control*, IEEE Trans. Syst., Man, Cybern., (1983), pp. 197–207.

[322] E. N. Pistikopoulos and N. A. Diangelakis, *Towards the integration of process design, control and scheduling: are we getting closer?*, Comput. Chem. Eng., 91 (2016), pp. 85–92.

[323] I. G. Polushin, A. L. Fradkov, and D. J. Hill, *Passivity and passification of nonlinear systems*, Autom. Remote Control, 61 (2000), pp. 355–388.

[324] L. S. Pontryagin, *Mathematical Theory of Optimal Processes*, John Wiley & Sons, 1962.

[325] M. Pósfai, Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, *Effect of correlations on network controllability*, Sci. Rep., 3 (2013), p. 1067.

[326] D. B. Pourkargar, A. Almansoori, and P. Daoutidis, *Distributed model predictive control of process networks: Impact of control architecture*, IFAC-PapersOnLine, 50 (2017), pp. 12452–12457. 20th IFAC World Congress.

[327] ———, *Impact of decomposition on distributed model predictive control: a process network case study*, Ind. Eng. Chem. Res., 56 (2017), pp. 9606–9616.

[328] ———, *Comprehensive study of decomposition effects on distributed output tracking of an integrated process over a wide operating range*, Chem. Eng. Res. Des., 134 (2018), pp. 553–563.

[329] D. B. Pourkargar, M. Moharir, A. Almansoori, and P. Daoutidis, *Distributed estimation and nonlinear model predictive control using community detection*, Ind. Eng. Chem. Res., 58 (2019), pp. 13495–13507.

[330] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, John Wiley & Sons, 2007.

[331] J. L. Proctor, S. L. Brunton, and J. N. Kutz, *Generalizing Koopman theory to allow for inputs and control*, SIAM J. Appl. Dyn. Syst., 17 (2018), pp. 909–930.

[332] Y. Pu, M. N. Zeilinger, and C. N. Jones, *Inexact fast alternating minimization algorithm for distributed model predictive control*, in 53rd Conf. Decis. Control (CDC), IEEE, 2014, pp. 5915–5921.

[333] A. Pukrittayakamee, M. Hagan, L. Raff, S. T. Bukkapatnam, and R. Komanduri, *Practical training framework for fitting a function and its derivatives*, IEEE Trans. Neural Netw., 22 (2011), pp. 936–947.

[334] P. Pulay, *Convergence acceleration of iterative sequences. The case of SCF iteration*, Chem. Phys. Lett., 73 (1980), pp. 393–398.

[335] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, 2005.

[336] S. J. Qin, *Process data analytics in the era of big data*, AIChE J., 60 (2014), pp. 3092–3100.

[337] S. J. Qin and T. A. Badgwell, *A survey of industrial model predictive control technology*, Control Eng. Pract., 11 (2003), pp. 733–764.

[338] S. J. Qin and L. H. Chiang, *Advances and opportunities in machine learning for process data analytics*, Comput. Chem. Eng., 126 (2019), pp. 465–473.

[339] H. Ramirez, B. Maschke, and D. Sbarbaro, *Irreversible port-Hamiltonian systems: A general formulation of irreversible processes with application to the CSTR*, Chem. Eng. Sci., 89 (2013), pp. 223–234.

[340] A. V. Rao, D. A. Benson, C. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, *Algorithm 902: GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the gauss pseudospectral method*, ACM Trans. Math. Softw., 37 (2010), p. 22.

[341] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, *Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations*, IEEE Trans. Autom. Control, 48 (2003), pp. 246–258.

[342] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási, *Hierarchical organization of modularity in metabolic networks*, Science, 297 (2002), pp. 1551–1555.

[343] J. B. Rawlings and R. Amrit, *Optimizing process economic performance using model predictive control*, in Nonlinear Model Predictive Control, Springer, 2009, pp. 119–138.

[344] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: Theory and design*, 2017.

[345] D. E. Reeves and Y. Arkun, *Interaction measures for nonsquare decentralized control structures*, AIChE J., 35 (1989), pp. 603–613.

[346] K. J. Reinschke, *Multivariable Control: A Graph Theoretic Approach*, Springer-Verlag, 1988.

[347] W. Ren and R. W. Beard, *Distributed Consensus in Multi-Vehicle Cooperative Control*, Springer, 2008.

[348] L. A. Ricardez-Sandoval, H. Budman, and P. Douglas, *Integration of design and control for chemical processes: A review of the literature and some recent results*, Annu. Rev. Control, 33 (2009), pp. 158–171.

[349] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, Springer, 1998.

[350] J. S. Rodriguez, B. Nicholson, C. Laird, and V. M. Zavala, *Benchmarking ADMM in nonconvex NLPs*, Comput. Chem. Eng., 119 (2018), pp. 315–325.

[351] P. Rombach, M. A. Porter, J. H. Fowler, and P. J. Mucha, *Core-periphery structure in networks (revisited)*, SIAM Rev., 59 (2017), pp. 619–646.

[352] A. Romer, J. Berberich, J. Köhler, and F. Allgöwer, *One-shot verification of dissipativity properties from input–output data*, IEEE Control Syst. Lett., 3 (2019), pp. 709–714.

[353] A. Romer, J. M. Montenbruck, and F. Allgöwer, *Determining dissipation inequalities from input-output samples*, IFAC-PapersOnLine, 50 (2017), pp. 7789–7794. 20th IFAC World Congress.

[354] H. H. Rosenbrock, *Distinctive problems of process control*, Chem. Eng. Progr., 58 (1962), pp. 43–50.

[355] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. Vandermeulen, A. Binder, E. Müller, and M. Kloft, *Deep one-class classification*, Proc. Mach. Learn. Res., 80 (2018), pp. 4393–4402. Proc. 35th Intl. Conf. Mach. Learn. (ICML).

[356] A. Ruszczyński, *Decomposition methods in stochastic programming*, Math. Prog., 79 (1997), pp. 333–353.

[357] M. Ruszkowski, V. Garcia-Osorio, and B. E. Ydstie, *Passivity based control of transport reaction systems*, AIChE J., 51 (2005), pp. 3147–3166.

[358] N. V. Sahinidis, *BARON: A general purpose global optimization software package*, J. Global Optim., 8 (1996), pp. 201–205.

[359] K. Sánchez-Sánchez and L. Ricardez-Sandoval, *Simultaneous process synthesis and control design under uncertainty: a worst-case performance approach*, AIChE J., 59 (2013), pp. 2497–2514.

[360] G. N. Saridis and C.-S. G. Lee, *An approximation theory of optimal control for trainable manipulators*, IEEE Trans. Syst., Man, Cybern., 9 (1979), pp. 152–159.

[361] A. Savitzky and M. J. E. Golay, *Smoothing and differentiation of data by simplified least squares procedures*, Anal. Chem., 36 (1964), pp. 1627–1639.

[362] R. Scattolini, *Architectures for distributed and hierarchical model predictive control–a review*, J. Process Control, 19 (2009), pp. 723–731.

[363] T. Schné and K. M. Hangos, *Decentralised controller structure design and retrofit of process systems based on graph theory*, Int. J. Syst. Sci., 42 (2011), pp. 1023–1033.

[364] C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann, *Mitigation of malicious attacks on networks*, Proc. Natl. Acad. Sci. U.S.A., 108 (2011), pp. 3838–3841.

[365] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, 2002.

[366] T. B. Schön, A. Wills, and B. Ninness, *System identification of nonlinear state-space models*, Automatica, 47 (2011), pp. 39–49.

[367] J. Schoukens and L. Ljung, *Nonlinear system identification: A user-oriented roadmap*, IEEE Control Syst. Mag., 39 (2019), pp. 28–99.

[368] P. O. Scokaert, D. Q. Mayne, and J. B. Rawlings, *Suboptimal model predictive control (feasibility implies stability)*, IEEE Trans. Autom. Control, 44 (1999), pp. 648–654.

[369] G. Scutari, F. Facchinei, and L. Lampariello, *Parallel and distributed methods for constrained nonconvex optimization – part I: theory*, IEEE Trans. Signal Process., 65 (2016), pp. 1929–1944.

[370] P. Seferlis and M. C. Georgiadis, *The Integration of Process Design and Control*, Elsevier, 2004.

[371] R. Sepulchre, M. Janković, and P. V. Kokotović, *Constructive Nonlinear Control*, Springer, 1997.

[372] M. E. Sezer and D. Šiljak, *Nested ε-decompositions and clustering of complex systems*, Automatica, 22 (1986), pp. 321–331.

[373] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*, SIAM, 2009.

[374] M. Sharifzadeh, *Integration of process design and control: A review*, Chem. Eng. Res. Des., 91 (2013), pp. 2515–2549.

[375] X. Shen, S. Diamond, M. Udell, Y. Gu, and S. Boyd, *Disciplined multi-convex programming*, in 29th Chin. Control Decis. Conf. (CCDC), IEEE, 2017, pp. 895–900.

[376] J. Shin, T. A. Badgwell, K.-H. Liu, and J. H. Lee, *Reinforcement learning – overview of recent progress and implications for process control*, Comput. Chem. Eng., 127 (2019), pp. 282–294.

[377] F. G. Shinskey, *Process Control Systems: Application, Design, Adjustment*, McGraw-Hill, 1967.

[378] ——, *Process control: as taught vs as practiced*, Ind. Eng. Chem. Res., 41 (2002), pp. 3745–3750.

[379] D. D. Šiljak, *Large-Scale Dynamic Systems: Stability and Structure*, North Holland, 1978.

[380] ——, *Decentralized Control of Complex Systems*, Academic Press, 1991.

[381] D. D. Šiljak and A. I. Zečević, *Control of large-scale systems: Beyond decentralized feedback*, Annu. Rev. Control, 29 (2005), pp. 169–179.

[382] J. M. Simkoff and M. Baldea, *Parameterizations of data-driven nonlinear dynamic process models for fast scheduling calculations*, Comput. Chem. Eng., 129 (2019), p. 106498.

[383] D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*, John Wiley & Sons, 2006.

[384] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky, *Nonlinear black-box modeling in system identification: a unified overview*, Automatica, 31 (1995), pp. 1691–1724.

[385] S. Skogestad and M. Morari, *Implications of large RGA-elements on control performance*, Ind. Eng. Chem. Res., 26 (1987), pp. 2323–2330.

[386] S. Skogestad and I. Postlethwaite, *Multivariable Feedback control: Analysis and Design*, Wiley, 2005.

[387] M. Soroush, *State and parameter estimations and their applications in process control*, Comput. Chem. Eng., 23 (1998), pp. 229–245.

[388] M. Soroush and C. Kravaris, *Nonlinear control of a batch polymerization reactor: an experimental study*, AIChE J., 38 (1992), pp. 1429–1448.

[389] ———, *MPC formulation of GLC*, AIChE J., 42 (1996), pp. 2377–2381.

[390] F. Sorrentino, M. di Bernardo, F. Garofalo, and G. Chen, *Controllability of complex networks via pinning*, Phys. Rev. E, 75 (2007), p. 046103.

[391] S. Spielberg, A. Tulsyan, N. P. Lawrence, P. D. Loewen, and R. B. Gopaluni, *Toward self-driving processes: A deep reinforcement learning approach to control*, AIChE J., 65 (2019), p. e16689.

[392] ———, *Towards self-driving processes: A deep reinforcement learning approach to control*, AIChE J., 65 (2019), p. e16689.

[393] S. S. Stanković and D. D. Šiljak, *Sequential LQG optimization of hierarchically structured systems*, Automatica, 25 (1989), pp. 545–559.

[394] G. Stephanopoulos and C. Ng, *Perspectives on the synthesis of plant-wide control structures*, J. Process Control, 10 (2000), pp. 97–111.

[395] G. Stephanopoulos and G. V. Reklaitis, *Process systems engineering: From Solvay to modern bio-and nanotechnology: A history of development, successes and prospects for the future*, Chem. Eng. Sci., 66 (2011), pp. 4272–4306.

[396] D. V. Steward, *Partitioning and tearing systems of equations*, J. SIAM Ser. B Numer. Anal., 2 (1965), pp. 345–365.

[397] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, *Cooperative distributed model predictive control*, Syst. Control Lett., 59 (2010), pp. 460–469.

[398] S. H. Strogatz, *Exploring complex networks*, Nature, 410 (2001), pp. 268–276.

[399] T. H. Summers, F. L. Cortesi, and J. Lygeros, *On submodularity and controllability in complex dynamical networks*, IEEE Trans. Control Netw. Syst., 3 (2016), pp. 91–101.

[400] K. Sun and X. A. Sun, *A two-level distributed algorithm for general constrained nonconvex optimization with global convergence*, arXiv preprint arXiv:1902.07654, (2019).

[401] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[402] W. Tan and A. Packard, *Stability region analysis using polynomial and composite polynomial Lyapunov functions and sum-of-squares programming*, IEEE Trans. Autom. Control, 53 (2008), p. 565.

[403] M. Tanaskovic, L. Fagiano, C. Novara, and M. Morari, *Data-driven control of nonlinear systems: An on-line direct approach*, Automatica, 75 (2017), pp. 1–10.

[404] W. Tang, A. Allman, D. B. Pourkargar, and P. Daoutidis, *Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection*, Comput. Chem. Eng., 111 (2018), pp. 43–54.

[405] W. Tang, P. H. Constantino, and P. Daoutidis, *Optimal sparse network topology under sparse control in Laplacian network*, IFAC-PapersOnLine, 52 (2019), pp. 273–278. 8th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys).

[406] W. Tang and P. Daoutidis, *Distributed/hierarchical control architecture design*, IFAC-PapersOnLine, 50 (2017), pp. 12015–12020. 20th IFAC World Congress.

[407] ———, *Distributed adaptive dynamic programming for data-driven optimal control*, Syst. Control Lett., 120 (2018), pp. 36–43.

[408] ———, *Network decomposition for distributed control through community detection in input–output bipartite graphs*, J. Process Control, 64 (2018), pp. 7–14.

[409] ———, *The role of community structures in sparse feedback control*, in Am. Control Conf. (ACC), 2018, pp. 1790–1795.

[410] ———, *A bilevel programming approach to the convergence analysis of control-Lyapunov functions*, IEEE Trans. Autom. Control, 64 (2019), pp. 4174–4179.

[411] ———, *Dissipativity learning control (DLC): A framework of input–output data-driven control*, Comput. Chem. Eng., 130 (2019), p. 106576.

[412] ———, *Distributed control and optimization of process system networks: A review and perspective*, Chin. J. Chem. Eng., 27 (2019), pp. 1461–1473.

[413] ———, *Distributed nonlinear model predictive control through accelerated parallel ADMM*, in Am. Control Conf. (ACC), IEEE, 2019, pp. 1406–1411.

[414] ———, *Input-output data-driven control through dissipativity learning*, in Am. Control Conf. (ACC), 2019, pp. 4217–4222.

[415] ———, *Lyapunov dynamic flexibility of nonlinear processes*, in Computer Aided Chemical Engineering, vol. 47, Elsevier, 2019, pp. 35–40. 9th International Conference on Foundations of Computer-Aided Process Design (FOCAPD).

[416] ———, *Fast and stable nonconvex constrained distributed optimization: The ELLADA algorithm*, arXiv preprint arXiv:2004.01977, (2020).

[417] W. Tang, D. B. Pourkargar, and P. Daoutidis, *Relative time-averaged gain array (RTAGA) for distributed control-oriented network decomposition*, AIChE J., 64 (2018), pp. 1682–1690.

[418] Y. Tang, F. Qian, H. Gao, and J. Kurths, *Synchronization in complex networks and its application–a survey of recent advances and challenges*, Annu. Rev. Control, 38 (2014), pp. 184–198.

[419] T. Tatarenko and B. Touri, *Non-convex distributed optimization*, IEEE Trans. Autom. Control, 62 (2017), pp. 3744–3757.

[420] M. Tawarmalani and N. V. Sahinidis, *A polyhedral branch-and-cut approach to global optimization*, Math. Prog., 103 (2005), pp. 225–249.

[421] M. J. Tippett and J. Bao, *Distributed model predictive control based on dissipativity*, AIChE J., 59 (2013), pp. 787–804.

[422] ———, *Distributed dissipative model predictive control for process networks with imperfect communication*, AIChE J., 60 (2014), pp. 1682–1699.

[423] A. Toth and C. Kelley, *Convergence analysis for Anderson acceleration*, SIAM J. Numer. Anal., 53 (2015), pp. 805–819.

[424] L. Tung and T. Edgar, *Analysis of control-output interactions in dynamic systems*, AIChE J., 27 (1981), pp. 690–693.

[425] I. Y. Tyukin, E. Steur, H. Nijmeijer, and C. Van Leeuwen, *Adaptive observers and parameter estimation for a class of systems nonlinear in the parameters*, Automatica, 49 (2013), pp. 2409–2423.

[426] R. S. Upadhye and E. A. Grens, *Selection of decompositions for chemical process simulation*, AIChE J., 21 (1975), pp. 136–143.

[427] A. J. van der Schaft, *L_2-Gain and Passivity Techniques in Nonlinear Control*, Springer, 3^{rd} ed., 2017.

[428] J. van Helvoort, B. de Jager, and M. Steinbuch, *Direct data-driven recursive controller unfalsification with analytic update*, Automatica, 43 (2007), pp. 2034–2046.

[429] H. J. Van Waarde, J. Eising, H. L. Trentelman, and M. K. Camlibel, *Data informativity: a new perspective on data-driven analysis and control*, IEEE Trans. Autom. Control, (2020). in press.

[430] D. K. Varvarezos, I. E. Grossmann, and L. T. Biegler, *An outer-approximation method for multiperiod design optimization*, Ind. Eng. Chem. Res., 31 (1992), pp. 1466–1477.

[431] A. N. Venkat, J. B. Rawlings, and S. J. Wright, *Stability and optimality of distributed model predictive control*, in 44th Conf. Decis. Control (CDC), IEEE, 2005, pp. 6680–6685.

[432] V. Venkatasubramanian, *The promise of artificial intelligence in chemical engineering: Is it here, finally?*, AIChE J., 65 (2019), pp. 466–478.

[433] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, *A review of process fault detection and diagnosis: Part III: Process history based methods*, Comput. Chem. Eng., 27 (2003), pp. 327–346.

[434] M. Vidyasagar, *Decomposition techniques for large-scale systems with nonadditive interactions: Stability and stabilizability*, IEEE Trans. Autom. Control, 25 (1980), pp. 773–779.

[435] N. Vora and P. Daoutidis, *Nonlinear model reduction of chemical reaction systems*, AIChE J., 47 (2001), pp. 2320–2332.

[436] A. Wächter and L. T. Biegler, *Line search filter methods for nonlinear programming: Motivation and global convergence*, SIAM J. Optim., 16 (2005), pp. 1–31.

[437] ——, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Math. Prog., 106 (2006), pp. 25–57.

[438] G. P. Wagner, M. Pavlicev, and J. M. Cheverud, *The road to modularity*, Nature Rev. Genet., 8 (2007), pp. 921–931.

[439] B. Wahlberg, M. B. Syberg, and H. Hjalmarsson, *Non-parametric methods for l_2-gain estimation using iterative experiments*, Automatica, 46 (2010), pp. 1376–1381.

[440] W. Wan, J. P. Eason, B. Nicholson, and L. T. Biegler, *Parallel cyclic reduction decomposition for dynamic optimization problems*, Comput. Chem. Eng., 120 (2019), pp. 54–69.

[441] H. Wang, Q. Li, G. D'Agostino, S. Havlin, H. E. Stanley, and P. Van Mieghem, *Effect of the interconnected network structure on the epidemic threshold*, Phys. Rev. E, 88 (2013), p. 022801.

[442] Y. Wang, W. Yin, and J. Zeng, *Global convergence of ADMM in nonconvex nonsmooth optimization*, J. Sci. Comput., 78 (2019), pp. 29–63.

[443] ——, *Global convergence of ADMM in nonconvex nonsmooth optimization*, J. Sci. Comput., 78 (2019), pp. 29–63.

[444] D. J. Watts and S. H. Strogatz, *Collective dynamics of 'small-world' networks*, Nature, 393 (1998), p. 440.

[445] Q. Wei and D. Liu, *A novel iterative θ-adaptive dynamic programming for discrete-time nonlinear systems*, IEEE Trans. Autom. Sci. Eng., 11 (2014), pp. 1176–1190.

[446] P. J. Werbos, *Neural networks for control and system identification*, in Proc. 28th IEEE Conf. Decis. Control (CDC), IEEE, 1989, pp. 260–265.

[447] D. B. West, *Introduction to Graph Theory*, Prentice Hall, 2001.

[448] J. C. Willems, *Dissipative dynamical systems. II. linear systems with quadratic supply rates*, Arch. Ration. Mech. Anal., 45 (1972), pp. 352–393.

[449] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, *A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition*, J. Nonlin. Sci., 25 (2015), pp. 1307–1346.

[450] M. Witcher and T. McAvoy, *Interacting control-systems-steady-state and dynamic measurement of interaction*, ISA Trans., 16 (1977), pp. 35–41.

[451] P. Woolverton, *How to use relative gain analysis in systems with integrating variables*, Instrum. Technol., 27 (1980), pp. 63–65.

[452] S. J. Wright, *Coordinate descent algorithms*, Math. Program., 151 (2015), pp. 3–34.

[453] H. Wu and J. Zhao, *Deep convolutional neural network model based chemical process fault diagnosis*, Comput. Chem. Eng., 115 (2018), pp. 185–197.

[454] Y.-G. Xi, D.-W. Li, and S. Lin, *Model predictive control – status and challenges*, Acta Autom. Sin., 39 (2013), pp. 222–236.

[455] J. Xie, A. Liao, and X. Yang, *An inexact alternating direction method of multipliers with relative error criteria*, Optim. Lett., 11 (2017), pp. 583–596.

[456] L. Xie, X. Cai, J. Chen, and H. Su, *GA based decomposition of large scale distributed model predictive control systems*, Control Eng. Pract., 57 (2016), pp. 111–125.

[457] Q. Xiong, W.-J. Cai, and M.-J. He, *A practical loop pairing criterion for multivariable processes*, J. Process Control, 15 (2005), pp. 741–747.

[458] Y. Xu, *Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming*, SIAM J. Optim., 27 (2017), pp. 1459–1484.

[459] V. A. Yakubovich, *Theory of adaptive systems*, Dokl. Akad. Nauk SSSR, 182 (1968), pp. 518–521. (in Russian).

[460] G. Yan, J. Ren, Y.-C. Lai, C.-H. Lai, and B. Li, *Controlling complex networks: How much energy is needed?*, Phys. Rev. Lett., 108 (2012), p. 218703.

[461] X. Yang and L. T. Biegler, *Advanced-multi-step nonlinear model predictive control*, J. Process Control, 23 (2013), pp. 1116–1128.

[462] X. Yang, D. Liu, and D. Wang, *Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints*, Int. J. Control, 87 (2014), pp. 553–566.

[463] Y. Yang, G. Hu, and C. J. Spanos, *A proximal linearization-based fecentralized mthod for nonconvex problems with nonlinear constraints*, arXiv preprint arXiv:2001.00767, (2020).

[464] B. E. Ydstie, *Passivity based control via the second law*, Comput. Chem. Eng., 26 (2002), pp. 1037–1048.

[465] Y. Ye, I. E. Grossmann, and J. M. Pinto, *Mixed-integer nonlinear programming models for optimal design of reliable chemical plants*, Comput. Chem. Eng., 116 (2018), pp. 3–16.

[466] S. Yin and O. Kaynak, *Big data for modern industry: challenges and trends [point of view]*, Proc. IEEE, 103 (2015), pp. 143–146.

[467] X. Yin, K. Arulmaran, J. Liu, and J. Zeng, *Subsystem decomposition and configuration for distributed state estimation*, AIChE J., 62 (2016), pp. 1995–2003.

[468] X. Yin and J. Liu, *Input–output pairing accounting for both structure and strength in coupling*, AIChE J., 63 (2017), pp. 1226–1235.

[469] ——, *Subsystem decomposition of process networks for simultaneous distributed state estimation and control*, AIChE J., 65 (2019), pp. 904–914.

[470] C.-C. Yu and M. K. Fan, *Decentralized integral controllability and D-stability*, Chem. Eng. Sci., 45 (1990), pp. 3299–3309.

[471] W. Yu, G. Chen, and J. Lü, *On pinning synchronization of complex dynamical networks*, Automatica, 45 (2009), pp. 429–435.

[472] D. Yuan, D. W. C. Ho, and S. Xu, *Inexact dual averaging method for distributed multi-agent optimization*, Syst. Control Lett., 71 (2014), pp. 23–30.

[473] Z. Yuan, B. Chen, G. Sin, and R. Gani, *State-of-the-art and progress in the optimization-based simultaneous design and control for chemical processes*, AIChE J., 58 (2012), pp. 1640–1659.

[474] V. M. Zavala and M. Anitescu, *Real-time nonlinear optimization as a generalized equation*, SIAM J. Control Optim., 48 (2010), pp. 5444–5467.

[475] V. M. Zavala and L. T. Biegler, *Large-scale parameter estimation in low-density polyethylene tubular reactors*, Ind. Eng. Chem. Res., 45 (2006), pp. 7867–7881.

[476] ——, *The advanced-step NMPC controller: Optimality, stability and robustness*, Automatica, 45 (2009), pp. 86–93.

[477] V. M. Zavala, C. D. Laird, and L. T. Biegler, *Interior-point decomposition approaches for parallel solution of large-scale nonlinear parameter estimation problems*, Chem. Eng. Sci., 63 (2008), pp. 4834–4845.

[478] A. I. Zečević and D. D. Šiljak, *Balanced decompositions of sparse systems for multilevel parallel processing*, IEEE Trans. Circuits Syst. I, Fundam. Theor. Appl., 41 (1994), pp. 220–233.

[479] B. Zeng and L. Zhao, *Solving two-stage robust optimization problems using a column-and-constraint generation method*, Oper. Res. Lett., 41 (2013), pp. 457–461.

[480] H. Zhang, L. Cui, X. Zhang, and Y. Luo, *Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method*, IEEE Trans. Neural Netw., 22 (2011), pp. 2226–2236.

[481] H. Zhang, C. Qin, and Y. Luo, *Neural-network-based constrained optimal control scheme for discrete-time switched nonlinear system using dual heuristic programming*, IEEE Trans. Autom. Sci. Eng., 11 (2014), pp. 839–849.

[482] H. Zhang, J. Zhang, G.-H. Yang, and Y. Luo, *Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming*, IEEE Trans. Fuzzy Syst., 23 (2015), pp. 152–163.

[483] J. Zhang, B. O'Donoghue, and S. Boyd, *Globally convergent type-I Anderson acceleration for non-smooth fixed-point iterations*, arXiv preprint arXiv:1808.03971, (2018).

[484] J. Zhang, Y. Peng, W. Ouyang, and B. Deng, *Accelerating ADMM for efficient simulation and optimization*, ACM Trans. Graph., 38 (2019), p. 163.

[485] Q. Zhang, I. E. Grossmann, A. Sundaramoorthy, and J. M. Pinto, *Data-driven construction of convex region surrogate models*, Optim. Eng., 17 (2016), pp. 289–332.

[486] R. Y. Zhang and J. K. White, *GMRES-accelerated ADMM for quadratic objectives*, SIAM J. Optim., 28 (2018), pp. 3025–3056.

[487] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, *Bipartite network projection and personal recommendation*, Phys. Rev. E, 76 (2007), p. 046115.

[488] T. Zhou and Y. Zhang, *On the stability and robust stability of networked dynamic systems*, IEEE Trans. Autom. Control, 61 (2016), pp. 1595–1600.

[489] Y. Zhu, *Multivariable process identification for MPC: the asymptotic method and its applications*, J. Process Control, 8 (1998), pp. 101–115.

[490] J. G. Ziegler and N. B. Nichols, *Optimum settings for automatic controllers*, Trans. ASME, 64 (1942), pp. 759–768.

Appendices

Appendix A
Nonlinear MPC of the Reactor-Separator Process

The differential equations governing the process are:

$$\begin{aligned}
\frac{dV_1}{dt} &= F_{f1} + F_R - F_1 \\
\frac{dV_2}{dt} &= F_{f2} + F_1 - F_2 \\
\frac{dV_3}{dt} &= F_2 - F_P - F_R - F_3 \\
\frac{dT_1}{dt} &= \frac{F_{f1}}{V_1}(T_f - F_1) + \frac{F_R}{V_1}(T_3 - T_1) + \frac{Q_1}{\rho C_p V_1} - \frac{m}{C_p}(k_{11}x_{A1}\Delta H_1 + k_{21}x_{B1}\Delta H_2) \\
\frac{dT_2}{dt} &= \frac{F_{f2}}{V_2}(T_f - F_2) + \frac{F_1}{V_2}(T_1 - T_2) + \frac{Q_2}{\rho C_p V_2} - \frac{m}{C_p}(k_{12}x_{A2}\Delta H_1 + k_{22}x_{B2}\Delta H_2) \\
\frac{dT_3}{dt} &= \frac{F_2}{V_3}(T_2 - T_3) + \frac{Q_3}{\rho C_p V_3} \\
\frac{dx_{A1}}{dt} &= \frac{F_{f1}}{V_1}(x_{Af} - x_{A1}) + \frac{F_R}{V_1}(x_{AR} - x_{A1}) - k_{11}x_{A1} \\
\frac{dx_{B1}}{dt} &= \frac{F_R}{V_1}(x_{BR} - x_{B1}) - \frac{F_{f1}}{V_1}x_{B1} + k_{11}x_{A1} - k_{21}x_{B1} \\
\frac{dx_{A2}}{dt} &= \frac{F_{f2}}{V_2}(x_{Af} - x_{A2}) + \frac{F_1}{V_2}(x_{A1} - x_{A2}) - k_{12}x_{A2} \\
\frac{dx_{B2}}{dt} &= \frac{F_1}{V_2}(x_{B1} - x_{B2}) - \frac{F_{f2}}{V_2}x_{B2} + k_{12}x_{A2} - k_{22}x_{B2} \\
\frac{dx_{A3}}{dt} &= \frac{F_2}{V_3}(x_{A2} - x_{A3}) + \frac{F_P + F_R}{V_3}(x_{AR} - x_{A3}) \\
\frac{dx_{B3}}{dt} &= \frac{F_2}{V_3}(x_{B2} - x_{B3}) - \frac{F_P + F_R}{V_3}(x_{BR} - x_{B3}).
\end{aligned} \tag{A.1}$$

The percentage of the purge flow rate is fixed at $F_P/F_R = 0.02$. The process parameters and the setpoint is given in Table A.1. The inputs and states are then translated so that the origin is the operational steady state. The initial conditions for simulation are given in Table A.2. We consider a state regulation problem which aims to bring the system states to the origin.

For the NMPC controller, the sampling time is determined as $\Delta t = 90$ seconds. We use a simple backwards discretization:

$$x^{t+1} = x^t + f_0(x^t, u^t)\Delta t. \tag{A.2}$$

Table A.1: Process parameters and operational steady state

Parameters	Value	Variables	Value
T_f	359.1 K	F_{f1}	5.04 m^3/h
x_{Af}	1	F_{f2}	5.04 m^3/h
m	2.79 mol/kg	F_3	9.74 m^3/h
ρ	1000 $\mathrm{kg/m}^3$	F_R	17 m^3/h
C_p	4.2 kJ/(kg·K)	Q_1	715.3 MJ/h
k_1^0	$9.97 \times 10^6\ \mathrm{h}^{-1}$	Q_2	579.8 MJ/h
k_2^0	$9 \times 10^6\ \mathrm{h}^{-1}$	Q_3	568.7 MJ/h
E_1	50 kJ/mol	V_1	1.0 m^3
E_2	60 kJ/mol	V_2	0.5 m^3
ΔH_1	-60 kJ/mol	V_3	1.0 m^3
ΔH_2	-70 kJ/mol	T_1	432.4 K
α_A	5.0	T_2	427.1 K
α_B	1.0	T_3	432.1 K
α_C	0.5	x_{B3}	0.670

Table A.2: Initial conditions for simulation

State	Initial Value	State	Initial value
V_1	0.7 m^3	x_{A1}	0.890
V_2	0.7 m^3	x_{A2}	0.886
V_3	1.5 m^3	x_{A3}	0.748
T_1	388.7 K	x_{B1}	0.110
T_2	386.3 K	x_{B2}	0.113
T_3	390.6 K	x_{B3}	0.251

The receding horizon is $N = 15$ (22.5 min). The input variables are bound between 20% and 180% of their steady state absolute values. To regulate the holdup volumes, we impose three additional constraints on the flow rates such that the holdup volumes of all units approach the corresponding steady state values with a time constant of 0.1 hour, i.e.,

$$\begin{aligned} F_{f1} + F_R - F_1 &= -V_1/0.1\ \mathrm{h}, \\ F_{f2} + F_1 - F_2 &= -V_2/0.1\ \mathrm{h}, \\ F_2 - F_3 - F_P - F_R &= -V_3/0.1\ \mathrm{h}. \end{aligned} \tag{A.3}$$

The above descriptions and requirements amount to 315 variables and 225 equality constraints in total. The MPC controllers use a quadratic stage cost function $\Phi(u^t, x^t) =$

$(x^t)^\top Q x^t + (u^t)^\top R u^t$, in which the weight matrices for the states Q and the inputs R are diagonal:

$$\begin{aligned} Q &= \mathrm{diag}(10^3, 10^3, 10^3, 10^1, 10^1, 10^1, 10^3, 10^3, 10^3, 10^3, 10^3, 10^3) \\ R &= \mathrm{diag}(10^2, 10^2, 10^2, 10^2, 10^2, 10^2, 10^{-3}, 10^{-3}, 10^{-3}) \end{aligned} \tag{A.4}$$

when the units for flow rates, heat exchange rates, holdup volume, temperature are m^3/h, kJ/h, m^3, K, respectively, and the concentrations are measured by molar fractions. The terminal cost is also a quadratic function:

$$\Psi(x^N) = (x^N)^\top P x^N, \;\; P = \mathrm{diag}(10^3, 10^3, 10^3, 10^1, 10^1, 10^1, 10^3, 10^3, 10^3, 10^3, 10^3, 10^3) \tag{A.5}$$

For the distributed MPC under ADMM (6.17)–(6.19) used in Subsection 6.4.2, the parameters $\beta = 0.5$ and $\delta = 0.5$. α_0, the initial step size for gradient line search, is the largest possible value such that the resulting succeeding point stays within the feasible set. To maintain the well-conditioning of the augmented Lagrangian as well as a reasonably small deviation from the feasible set, the penalty coefficient γ is empirically tuned to 30. The ADMM iterations are terminated if either the relative error $\|v_{\text{new}} - v_{\text{old}}\|_2 / \|v_{\text{old}}\|_2$ or the absolute error $\|v_{\text{new}} - v_{\text{old}}\|_2$ is sufficiently small. We found that a lower bound smaller than 10^{-2} may increase the iteration number without significantly improving the control performance, while relaxing this bound to 0.05 or 0.1 will significantly deteriorate the solution optimality; hence we set this error lower bound as 10^{-2}. The computation is forced to terminate after 100 iterations if the previously mentioned termination condition has not been satisfied. After solving the optimization problem at a sampling time, the solution of primal variables is saved as the initial guess for the next sampling time. The initial dual variables are always reset to zero.

Remark A.1. *To guarantee the optimality of the solution, the terminating condition should be set as such that the constraints are well satisfied, i.e., $\|c(v)\| < \epsilon$ for a small $\epsilon > 0$. However, here the terminating condition is based on the change in primal variables $\|v_{\text{new}} - v_{\text{old}}\|$ instead of the constraint tolerance $\|c(v)\|$. Since the manipulated inputs to be determined are contained in the primal variables, when the change in the primal variables becomes small, it would be desirable to terminate the iterations rather than proceed to the optimal solution.*

Appendix B
Proofs for the ELLADA Algorithm

B.1 Proof of Lemma 7.1

First, since $x^{k,r+1}$ is chosen as the minimizer of the augmented Lagrangian with respect to x (Line 9, Algorithm 7.1), the update of x leads to a decrease in L:

$$L(x^{k,r+1}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}) \leq L(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}). \tag{B.1}$$

Second, we consider the decrease resulted from $\bar{x}$-update:

$$\begin{aligned} &L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) - L(x^{k,r+1}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}) \\ &= g(\bar{x}^{k,r+1}) - g(\bar{x}^{k,r}) + y^{k,r\top}(B\bar{x}^{k,r+1} - B\bar{x}^{k,r}) \\ &\quad + (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}\|^2 - (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r} + z^{k,r}\|^2 \\ &= g(\bar{x}^{k,r+1}) - g(\bar{x}^{k,r}) - (\rho^k/2)\|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2 \\ &\quad - \rho^k(\bar{x}^{k,r} - \bar{x}^{k,r+1})^\top B^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k\right). \end{aligned} \tag{B.2}$$

The minimization of $\bar{x}$ (Line 10, Algorithm 7.1) should satisfy the optimality condition

$$0 \in \partial g(\bar{x}^{k,r+1}) + \rho^k B^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k\right) + \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k,r+1}), \tag{B.3}$$

i.e., there exist vectors $v_1 \in \partial g(\bar{x}^{k,r+1})$ and $v_2 \in \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k,r+1})$ with

$$\rho^k B^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k\right) = -v_1 - v_2. \tag{B.4}$$

Since $v_1 \in \partial g(\bar{x}^{k,r+1})$ and g is convex, $v_1^\top(\bar{x}^{k,r} - \bar{x}^{k,r+1}) \leq g(\bar{x}^{k,r}) - g(\bar{x}^{k,r+1})$. And $v_2 \in \mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}^{k,r+1})$ implies $v_2^\top(\bar{x}^{k,r} - \bar{x}^{k,r+1}) \leq 0$. Hence

$$\begin{aligned} &\rho^k(\bar{x}^{k,r} - \bar{x}^{k,r+1})^\top B^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k\right) \\ &= -v_1^\top(\bar{x}^{k,r} - \bar{x}^{k,r+1}) - v_2^\top(\bar{x}^{k,r} - \bar{x}^{k,r+1}) \geq -(g(\bar{x}^{k,r}) - g(\bar{x}^{k,r+1})). \end{aligned} \tag{B.5}$$

Substituting the above inequality in (B.2), we obtain

$$L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) \leq L(x^{k,r+1}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}) - (\rho^k/2)\|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2. \tag{B.6}$$

Third, we consider the decrease resulted from z- and y-updates:

$$\begin{aligned}
&L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) - L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) = \\
&\lambda^{k\top}(z^{k,r+1} - z^{k,r}) + (\beta^k/2)(\|z^{k,r+1}\|^2 - \|z^{k,r}\|^2) \\
&+ y^{k,r+1\top}(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}) - y^{k,r\top}(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}) \\
&+ (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\|^2 - (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}\|^2.
\end{aligned} \tag{B.7}$$

Since $\upsilon(z; \lambda, \beta) = \lambda^\top z + \frac{\beta}{2}\|z\|^2$ is a convex function, whose gradient is $\nabla\upsilon(z; \lambda, \beta) = \lambda + \beta z$,

$$\upsilon(z^{k,r+1}; \lambda^k, \beta^k) - \upsilon(z^{k,r}; \lambda^k, \beta^k) \leq (\lambda^k + \beta^k z^{k,r+1})^\top (z^{k,r+1} - z^{k,r}), \tag{B.8}$$

From Line 11 of Algorithm 7.1 it can be obtained

$$\lambda^k + \beta^k z^{k,r+1} = -y^{k,r+1}. \tag{B.9}$$

Substituting into (B.7), we obtain

$$\begin{aligned}
&L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) - L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) \\
&\leq (y^{k,r+1} - y^{k,r})^\top (Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}) \\
&\quad + (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\|^2 - (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}\|^2 \\
&= (\rho^k/2)(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1})^\top (Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}) \\
&\quad + (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\|^2 - (\rho^k/2)\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r}\|^2 \\
&= -(\rho^k/2)\|z^{k,r+1} - z^{k,r}\|^2 + \rho^k\|Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\|^2
\end{aligned} \tag{B.10}$$

From (B.9),

$$Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1} = (y^{k,r+1} - y^{k,r})/\rho_k = -(\beta^k/\rho^k)(z^{k,r+1} - z^{k,r}). \tag{B.11}$$

Then (B.10) becomes

$$\begin{aligned}
&L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) - L(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) \\
&\leq -\left(\rho^k/2 - (\beta^k)^2/\rho^k\right)\|z^{k,r+1} - z^{k,r}\|^2 = -(\beta^k/2)\|z^{k,r+1} - z^{k,r}\|^2.
\end{aligned} \tag{B.12}$$

Summing up the inequalities (B.1), (B.6) and (B.12), we have proved the inequality

(7.21). Next, we show that the augmented Lagrangian is lower bounded, and hence is convergent towards some $\underline{L}^k \in \mathbb{R}$. We note that $\upsilon(z; \lambda, \beta)$ is a convex function of modulus β, it can be easily verified that

$$\upsilon(z^{k,r}; \lambda^k, \beta^k) + (\lambda^k + \beta^k z^{k,r})^\top (z' - z^{k,r}) + (\rho^k/2)\|z' - z^{k,r}\|^2 \geq \upsilon(z'; \lambda^k, \beta^k) \quad \text{(B.13)}$$

for any z', i.e.,

$$\upsilon(z^{k,r}; \lambda^k, \beta^k) + y^{k,r\top}(z^{k,r} - z') \geq \upsilon(z'; \lambda^k, \beta^k) - (\rho^k/2)\|z' - z^{k,r}\|^2. \quad \text{(B.14)}$$

Let $z' = -(Ax^{k,r} + B\bar{x}^{k,r})$ and remove the last term on the right-hand side. Then

$$\upsilon(z^{k,r}; \lambda^k, \beta^k) + y^{k,r\top}(Ax^{k,r} + B\bar{x}^{k,r} + z^{k,r}) \geq \upsilon(-(Ax^{k,r} + B\bar{x}^{k,r}); \lambda^k, \beta^k). \quad \text{(B.15)}$$

Hence

$$\begin{aligned} &L(x^{k,r}, \bar{x}^{k,r+1}, z^{k,r}, y^{k,r}) = f(x^{k,r}) + g(\bar{x}^{k,r}) + \upsilon(z^{k,r}; \lambda^k, \beta^k) \\ &+ y^{k,r\top}(Ax^{k,r} + B\bar{x}^{k,r} + z^{k,r}) + (\rho^k/2)\|Ax^{k,r} + B\bar{x}^{k,r} + z^{k,r}\|^2 \\ &\geq f(x^{k,r}) + g(\bar{x}^{k,r}) + \upsilon(-(Ax^{k,r} + B\bar{x}^{k,r}); \lambda^k, \beta^k). \end{aligned} \quad \text{(B.16)}$$

Since $\upsilon(z) = \lambda^\top z + \frac{\beta}{2}\|z\|^2 \geq -\|\lambda\|^2/(2\beta)$, λ is bounded in $[\underline{\lambda}, \overline{\lambda}]$, $\beta^k \geq \beta^1$, and f and g are bounded below, L has a lower bound. Lemma 7.1 is proved.

B.2 Proof of Corollary 7.1

Taking the limit $r \to \infty$ on the both sides of inequality (7.21), it becomes obvious that $B\bar{x}^{k,r+1} - B\bar{x}^{k,r}$ and $z^{k,r+1} - z^{k,r}$ converge to 0. Due to (B.11), we have $Ax^{k,r} + B\bar{x}^{k,r} + z^{k,r} \to 0$. Hence there must exist a r such that (7.20) is met. At this time, the optimality conditions for $x^{k,r+1}$ is written as

$$0 \in \partial f(x^{k,r+1}) + \mathcal{N}_{\mathcal{X}}(x^{k,r+1}) + A^\top y^{k,r} + \rho^k A^\top (Ax^{k,r+1} + B\bar{x}^{k,r} + z^{k,r}). \quad \text{(B.17)}$$

According to the update rule of $y^{k,r}$, the above expression is equivalent to

$$0 \in \partial f(x^{k,r+1}) + \mathcal{N}_{\mathcal{X}}(x^{k,r+1}) + A^\top y^{k,r+1} - \rho^k A^\top (B\bar{x}^{k,r+1} + z^{k,r+1} - B\bar{x}^{k,r} - z^{k,r}), \quad \text{(B.18)}$$

i.e.,

$$\rho^k A^\top (B\bar{x}^{k,r+1} + z^{k,r+1} - B\bar{x}^{k,r} - z^{k,r}) \in \partial f(x^{k,r+1}) + \mathcal{N}_\mathcal{X}(x^{k,r+1}) + A^\top y^{k,r+1}. \quad \text{(B.19)}$$

According to the first inequality of (7.20), the norm of the left hand side above is not larger than ϵ_1^k, which directly implies the first condition in (7.22).

In a similar manner, the second condition in (7.22) can be established. The third one follows from (B.9) and the fourth condition is obvious.

B.3 Proof of Lemma 7.2

We first consider the situation when β^k is unbounded. From (B.16), we have

$$\overline{L} \geq f(x^{k+1}) + g(x^{k+1}) - \lambda^{k\top}(Ax^{k+1} + B\bar{x}^{k+1}) + \frac{\beta^k}{2}\|Ax^{k+1} + B\bar{x}^{k+1}\|^2. \quad \text{(B.20)}$$

Since f and g are both lower bounded, as $\beta^k \to \infty$, we have $Ax^{k+1} + B\bar{x}^{k+1} \to 0$. Combined with the first two conditions of (7.22) in the limit of ϵ_1^k, ϵ_2^k, $\epsilon_3^k \downarrow 0$, we have reached (7.24). Then we suppose that β^k is bounded, i.e., the amplification step $\beta^{k+1} = \gamma\beta^k$ is executed for only a finite number of outer iterations. According to Lines 17–21 of Algorithm 7.1, expect for some finite choices of k, $\|z^{k+1}\| \leq \omega\|z^k\|$ always hold. Therefore $z^{k+1} \to 0$. Apparently, (7.24) follows from the limit of (7.22).

B.4 Proof of Lemma 7.3

From Lemma 7.1 one knows that within R inner iterations

$$\overline{L} - \underline{L}^k \geq \beta^k \sum_{r=1}^{R} \|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2 + \frac{\beta^k}{2}\sum_{r=1}^{R}\|\bar{z}^{k,r+1} - z^{k,r}\|^2. \quad \text{(B.21)}$$

Hence

$$\min_{r=1,\dots,R} \|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2, \min_{r=1,\dots,R} \|z^{k,r+1} - z^{k,r}\|^2 \sim \mathcal{O}(1/\beta^k R). \quad \text{(B.22)}$$

Then

$$\|B\bar{x}^{k,R+1} - B\bar{x}^{k,R}\|,\ \|z^{k,R+1} - z^{k,R}\| \sim \mathcal{O}(1/\sqrt{\beta^k R}). \quad \text{(B.23)}$$

For the k-th outer iteration, its inner iterations are terminated when (7.20) is met, which is translated into the following relations:

$$\begin{aligned}
&\mathcal{O}(\rho^k/\sqrt{\beta^k R^k}) \leq \epsilon_1^k \sim \mathcal{O}(\vartheta^k), \\
&\mathcal{O}(\rho^k/\sqrt{\beta^k R^k}) \leq \epsilon_2^k \sim \mathcal{O}(\vartheta^k), \\
&\mathcal{O}(1/\sqrt{\beta^k R^k}) \leq \epsilon_3^k \sim \mathcal{O}(\vartheta^k/\beta^k).
\end{aligned} \tag{B.24}$$

where the last relation uses (B.11) with $\rho^k = 2\beta^k$. Therefore

$$R^k \sim \mathcal{O}(\beta^k/\vartheta^{2k}). \tag{B.25}$$

At the end of the k-th iteration, suppose that Lines 19–20 and Lines 17–18 of Algorithm 7.1 have been executed for k_1 and k_2 times, respectively ($k_1 + k_2 = k$). Then the obtained z^{k+1} satisfies $\|z^{k+1}\| \sim \mathcal{O}(\omega^{k_1})$, and $\|Ax^{k+1} + B\bar{x}^{k+1} + z^{k+1}\| \leq \epsilon_3^k \sim \mathcal{O}(\vartheta^k/\beta^k)$, which imply

$$\|Ax^{k+1} + B\bar{x}^{k+1}\| \leq \mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1}). \tag{B.26}$$

From (B.20),

$$\beta^k\|Ax^{k+1} + B\bar{x}^{k+1}\|^2 \sim \beta^k(\mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1}))^2 \sim \mathcal{O}(1). \tag{B.27}$$

Substituting (B.27) into (B.25), we obtain

$$R^k \sim \mathcal{O}\left(\frac{1}{\vartheta^{2k}} \frac{1}{(\mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1}))^2}\right). \tag{B.28}$$

When $\vartheta \leq \omega$, $\vartheta^k \leq \omega^k \leq \omega^{k_1}\gamma^{k_2}$, and hence $\gamma^{k_2}\vartheta^k \leq \omega^{k_1}$, i.e., ω^{k_1} dominates over ϑ^k/β^k, leading to

$$R^k \sim \mathcal{O}(1/\vartheta^{2k}\omega^{2k_1}) \sim \mathcal{O}(1/\vartheta^{2k}\omega^{2k}). \tag{B.29}$$

For K outer iterations, the total number of inner iterations is

$$R = \sum_{k=1}^{K} R^k \sim \mathcal{O}\left(\sum_{k=1}^{K} \frac{1}{\vartheta^{2k}\omega^{2k}}\right) \sim \mathcal{O}\left(\frac{1}{\vartheta^{2K}\omega^{2K}}\right). \tag{B.30}$$

The number of outer iterations needed to reach an ϵ-approximate stationary point is obviously $K \sim \mathcal{O}(\log_\vartheta \epsilon)$. Then

$$R \sim \mathcal{O}(\epsilon^{-2(1+\varsigma)}). \tag{B.31}$$

B.5 Proof of Lemma 7.6

Through the inner iterations, only Anderson acceleration might lead to an increase in the barrier augmented Lagrangian. Combining Assumption 7.3, Assumption 7.5, and the safeguarding criterion (7.46), we obtain

$$L_{b^k}(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) \leq \overline{L} + \tilde{L}_0 \eta_L \sum_{r=0}^{\infty} \frac{1}{r^{1+\sigma}} < +\infty, \tag{B.32}$$

Together with Assumptions 7.1 and 7.2, L_{b^k} is also bounded below. Therefore L_{b^k} is bounded in a closed interval and must have converging subsequences. Therefore we can choose a subsequence converging to the lower limit $\underline{L}$. For any $\varepsilon > 0$ there exists an index R of inner iteration in this subsequence, such that $\tilde{L}_0 \eta_L \sum_{r=R}^{\infty} r^{-(1+\sigma)} < \varepsilon/2$ and $L_{b^k}(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) < \underline{L} + \varepsilon/2$ for any $r \geq R$ on this subsequence. It then follows that for any $r \geq R$, whether on the subsequence or not, it holds that

$$L_{b^k}(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) < \underline{L} + \varepsilon. \tag{B.33}$$

Hence the upper limit is not larger than $\underline{L} + \varepsilon$. Due to the arbitrariness of $\varepsilon > 0$, the lower limit coincides with the upper limit, and hence the sequence of barrier augmented Lagrangian is convergent.

The Lagrangian convergence implies that as $r \to \infty$,

$$L_{b^k}(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}) - L_{b^k}(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}) \to 0. \tag{B.34}$$

Suppose that r is not an accelerated iteration, then since this quantity does not exceed $-\beta^k \|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\|^2 - (\beta^k/2)\|z^{k,r+1} - z^{k,r}\|^2$, we must have $B\bar{x}^{k,r+1} - B\bar{x}^{k,r} \to 0$ and $z^{k,r+1} - z^{k,r} \to 0$. Otherwise if inner iteration r is accelerated, the convergence of $B\bar{x}^{k,r+1} - B\bar{x}^{k,r}$ and $z^{k,r+1} - z^{k,r}$ are automatically guaranteed by the second criterion (7.48) of accepting Anderson acceleration. The convergence properties of these two sequences naturally fall into the paradigm of Lemma 7.1 for establishing the convergence to approximate KKT conditions of the relaxed problem.

Appendix C
Supplementary Information to Lie-Sobolev System Identification

C.1 Performance of Lie-Sobolev Luenberger for Linear Systems with Structural Errors

Consider a linear system

$$\dot{x} = Ax + Bu + \varphi(x), \;\; y = Cx, \tag{C.1}$$

where the structural error $\varphi(x)$ is unknown but assumed to be upper bounded in its norm. Following the discussions in Subsection 2.4 of the main text, the Luenberger observer results in an observation error e that satisfies

$$\dot{e} = (A - LC)e + \varphi(x), \tag{C.2}$$

in which $\bar{A} := A - LC$ is Hurwitz. The performance of the observer is captured by the norm of $\bar{A}^{-1}\varphi(x)$. Assuming that the structural error $\varphi(x)$ does not affect the relative degrees, when a Lie-Sobolev formulation is used, the resulting observation errors $\tilde{e}$ will satisfy

$$\dot{\tilde{e}} = (A - \tilde{L}\tilde{C})\tilde{e} + \varphi(x) - \sum_{i=1}^{d_y}\sum_{r=1}^{\rho_i} \ell_i^r c_i L_{Ax}^{r-1}\varphi(x), \tag{C.3}$$

where ℓ_i^r is the column of $\tilde{L}$ corresponding to the augmented $y_i^{(r)}$ in the Luenberger observer, and L stands for Lie derivatives.

We note that the last term on the right-hand side of the above formula, which is related to the output derivatives in the Lie-Sobolev observer, can be written as $\delta\bar{A} \cdot \tilde{\varphi}(x)$ for a proper vector $\tilde{\varphi}$ of Lie derivatives of $\varphi(x)$. Consider the squared norm of $\bar{A}^{-1}\varphi(x)$. With an infinitesimal Lie-Sobolev modification (meaning that the weights v_i^r, $r = 1, \dots, \rho_i$, $i = 1, \dots, d_y$ corresponding to the output derivatives are set as sufficiently large values), which leads to infinitesimal changes in Υ and thereafter Y and L, the resulting change in $\|\bar{A}^{-1}\varphi(x)\|^2$ can be calculated as the following quadratic form:

$$\begin{aligned}\delta(\|\bar{A}^{-1}\varphi(x)\|^2) =& (\tilde{\varphi}(x) - \bar{A}^{-1}\varphi(x))^\top \cdot \delta\bar{A}^\top \cdot \bar{A}^{-\top}\bar{A}^{-1}\varphi(x) \\ &+ \varphi(x)^\top \bar{A}^{-\top}\bar{A}^{-1} \cdot \delta\bar{A} \cdot (\tilde{\varphi}(x) - \bar{A}^{-1}\varphi(x)).\end{aligned} \tag{C.4}$$

Since $\delta\bar{A}+\delta\bar{A}^\top \preceq 0$, as long as $\tilde{\varphi}(x)$ is appropriately oriented, e.g., is such that $\tilde{\varphi}(x) - \bar{A}^{-1}\varphi(x)$ is approximately parallel to $\bar{A}^{-\top}\bar{A}^{-1}\varphi(x)$, by adjusting the closed-loop transfer matrix $\bar{A}$ through the tuning of the observer gain L, the above formula becomes non-positive. Therefore $\|\bar{A}^{-1}\varphi(x)\|^2$ is reduced and hence the performance of the Luenberger observer can be improved by adopting a Lie-Sobolev formulation.

C.2 Proofs of Propositions

C.2.1 Proof of Proposition 11.1

The design of the gradient descent-based observer-estimator results in the time derivative of Q equal to

$$\dot{Q} = -\left\|\dot{\hat{x}} - \hat{f} - \hat{g}u\right\|_{\Gamma_\sigma}^2 - \left\|\frac{\partial Q}{\partial \hat{\theta}}\right\|_{\Gamma_\pi}^2 + \frac{\partial S}{\partial \hat{x}}(\hat{f}+\hat{g}u) + \frac{\partial S}{\partial u}\dot{u} + \sum_{i=1}^{d_y}\sum_{r=0}^{\rho_i}\frac{\partial S}{\partial y_i^{(r)}}y_i^{(r+1)}. \quad \text{(C.5)}$$

The first two terms are negative whenever $\dot{\hat{x}}(t) = \hat{f}(\hat{x}(t)|\hat{\theta}(t)) + \hat{g}(\hat{x}(t)|\hat{\theta}(t))u(t)$ and $\partial Q/\partial\hat{\theta} = 0$ are not satisfied, i.e., when there exist state observer errors and the parameter estimates are not at their temporal stationary values, respectively. The remaining three terms are not manipulable by the construction of σ and π, in which the first term is the change of S resulted from the flow of the state observations, and the other two terms are resulted from the exogenous changes in the input and output and output derivative signals, respectively. For convergence, these non-manipulable terms should be small enough compared to the negative definite terms. By finding the partial derivatives of S and substituting them into (C.5), after simplifications we have

$$\begin{aligned}\dot{Q} = &-\left\|\dot{\hat{x}} - \hat{f} - \hat{g}u\right\|_{\Gamma_\sigma}^2 - \left\|\frac{\partial Q}{\partial \hat{\theta}}\right\|_{\Gamma_\pi}^2 \\ &+ \sum_{i=1}^{d_y}\left\{\sum_{r=0}^{\rho_i-1} w_i^r \Delta(L_f^r h_i)\left[\Delta\left(\frac{\partial L_f^r h_i}{\partial x}\right)(f+gu) + \frac{\partial L_f^r h_i}{\partial x}(\Delta f + \Delta g u)\right]\right. \\ &+ w_i^{\rho_i}\left[\Delta(L_f^{\rho_i}h_i) + \Delta(L_g L_f^{\rho_i}h_i)u\right]\left[\Delta\left(\frac{\partial L_f^{\rho_i}h_i}{\partial x} + \frac{\partial L_g L_f^{\rho_i-1}h_i}{\partial x}u\right)(f+gu)\right. \\ &\left.\left.+\left(\frac{\partial L_f^{\rho_i}h_i}{\partial x} + \frac{\partial L_g L_f^{\rho_i-1}h_i}{\partial x}u\right)(\Delta f + \Delta g u) + \Delta(L_g L_f^{\rho_i-1}h_i)\dot{u}\right]\right\}.\end{aligned} \quad \text{(C.6)}$$

where Δ denotes the differences between the true dynamics evaluated at true states $x(t)$ and the estimated dynamics at observed states $\hat{x}(t)$.

Under the assumptions of Proposition 11.1, there exist constants $c_\sigma, c_\pi > 0$ such that

$$\dot{Q} \leq -c_\sigma \|\dot{\hat{x}} - \hat{f} - \hat{g}u\|^2 - c_\pi \|\hat{\theta} - \theta\|^2, \tag{C.7}$$

and hence the criterion $Q(t)$ becomes a Lyapunov function. As $t \to 0$, we have $\dot{\hat{x}}(t) - \hat{f}(\hat{x}(t)|\hat{\theta}(t)) - \hat{g}(\hat{x}(t)|\hat{\theta}(t))u(t) \to 0$ and $\hat{\theta}(t) - \theta \to 0$, and therefore $\hat{x}(t) - x(t) \to 0$. Of course, in real situations the assumptions may not hold globally on $\mathcal{X} \times \mathcal{U}$, but rather locally in a neighborhood of zero errors in state observations and parameter estimates. Hence the matrices Γ_σ and Γ_π can not be chosen arbitrarily large when the errors in the solutions are not sufficiently small.

C.2.2 Proof of Proposition 11.2

Under the first two assumptions, we may specify constants $M_{11}, M_{12}, M_{22}, M_0 > 0$ such that

$$\begin{aligned}\dot{Q} \leq &-\|\dot{\hat{x}} - \hat{f} - \hat{g}u\|^2_{\Gamma_\sigma} - \left\| \left(\frac{\partial Q}{\partial \hat{\theta}}\right)_0 + \epsilon_Q^\top \right\|^2_{\Gamma_\pi} + M_{11}\|\dot{\hat{x}} - \hat{f} - \hat{g}u\|^2 \\ &+ 2M_{12}\|\dot{\hat{x}} - \hat{f} - \hat{g}u\|\|\hat{\theta} - \theta\| + M_{22}\|\hat{\theta} - \theta\|^2 + M_0,\end{aligned} \tag{C.8}$$

where the second term on the right hand side is further bounded by $-\mu\lambda_{\min}(\Gamma_\pi)\|\hat{\theta} - \theta\|^2 + 2c_Q\lambda_{\max}(\Gamma_\pi)(\partial Q/\partial\hat{\theta})$, where $\partial Q/\partial\hat{\theta}$ can be bounded linearly with $m_Q, \ell_Q, c_Q > 0$ as in the first assumption. It follows that

$$\dot{Q} \leq -M_2\left(\left\|\dot{\hat{x}} - \hat{f} - \hat{g}u\right\|^2 + \mu\|\hat{\theta} - \theta\|^2\right) + M_1\left(\left\|\dot{\hat{x}} - \hat{f} - \hat{g}u\right\|^2 + \mu\|\hat{\theta} - \theta\|^2\right)^{1/2} + M_0, \tag{C.9}$$

where

$$\begin{aligned} M_1 &= 2c_Q\lambda_{\max}(\Gamma_\pi)(\ell_Q + m_Q/\mu)\max(\ell_Q, m_Q), \\ M_2 &= \min\left(\lambda_{\min}(\Gamma_\sigma) - M_{11} - M_{12}/\mu, \lambda_{\min}(\Gamma_\pi) - M_{12} - M_{22}/\mu\right). \end{aligned} \tag{C.10}$$

Therefore Q is guaranteed to decrease unless

$$\left\|\dot{\hat{x}} - \hat{f} - \hat{g}u\right\|^2 + \mu\|\hat{\theta} - \theta\|^2 \leq \left(\frac{M_1 + (M_1^2 + 4M_0M_2)^{1/2}}{2M_2}\right)^2. \tag{C.11}$$

When this inequality is satisfied, according to the first assumption, there will be a corresponding upper bound $Q_{\max} > 0$ of Q. In other words, whenever $Q \geq Q_{\max}$, $\dot{Q} < 0$, implying that Q is ultimately bounded, and hence the errors are ultimately bounded. Thus we have proved the proposition. We note that the right-hand side expression above is always lower-bounded by a constant:

$$M_1/M_2 \geq c_Q(\ell_Q + m_Q/\mu) \max(\ell_Q, m_Q). \tag{C.12}$$

This implies that the use of high-gain observers and estimators can not eliminate the intrinsic uncertainties due to the existence of non-vanishing structural errors.

C.2.3 Proof of Proposition 11.3

Having assumed the Lipschitz continuity of f, g, h and Lie derivatives, we can claim that the difference between the estimated model and the true dynamics and their Lie derivatives will be ultimately bounded. That is, $\exists B > 0$, $\forall \epsilon > 0$, $\exists T_\epsilon > 0$ such that $\forall t > T_\epsilon$,

$$\begin{aligned}
&\|f(x(t)) - \hat{f}(x(t)|\hat{\theta}(t)) + g(x(t))u(t) - \hat{g}(x(t)|\hat{\theta}(t))u(t)\|^2 + \\
&\sum_{i=1}^{d_y} \left[\sum_{r=0}^{\rho_i - 1} w_i^r \|L_f^r h_i(x(t)) - L_{\hat{f}}^r \hat{h}_i(x(t))\|^2 + w_i^{\rho_i} \|L_f^{\rho_i} h_i(x(t)) - L_{\hat{f}}^{\rho_i} \hat{h}_i(x(t)) \right. \\
&\left. + L_g L_f^{\rho_i - 1} h_i(x(t))u(t) - L_{\hat{g}} L_{\hat{f}}^{\rho_i - 1} \hat{h}_i(x(t))u(t)\|^2 \right] \leq B + \epsilon.
\end{aligned} \tag{C.13}$$

Under the assumptions of Propositions 11.2 and 11.3, replacing the $x(t)$ and $u(t)$ with any $x \in \mathcal{X}$ and $u \in \mathcal{U}$ leads to an increase linearly bounded by η on the right-hand side of the above formula. That is, $\forall x \in \mathcal{X}$, $\forall u \in \mathcal{U}$,

$$\begin{aligned}
&\|f(x) - \hat{f}(x|\theta) + g(x)u - \hat{g}(\hat{x}|\theta)u\|^2 + \sum_{i=1}^{d_y} \left[\sum_{r=0}^{\rho_i - 1} w_i^r \|L_f^r h_i(x) - L_{\hat{f}}^r \hat{h}_i(x|\theta)\|^2 \right. \\
&\left. + w_i^{\rho_i} \|L_f^{\rho_i} h_i(x) - L_{\hat{f}}^{\rho_i} \hat{h}_i(x|\theta) + L_g L_f^{\rho_i - 1} h_i(x)u - L_{\hat{g}} L_{\hat{f}}^{\rho_i - 1} \hat{h}_i(x|\theta)u\|^2 \right] \leq B(\epsilon, \eta).
\end{aligned} \tag{C.14}$$

Thus proving the proposition.

C.3 Dynamics of the Etherification Reactor

In the glycerol etherification process, glycerol reacts with isobutene to yield mono-, di-, and tri-*tert*-butyl ethers of glycerol with a side dimerization reaction of isobutene:

$$\mathrm{G} + \mathrm{IB} \rightleftharpoons \mathrm{ME}, \quad \mathrm{ME} + \mathrm{IB} \rightleftharpoons \mathrm{DE}, \mathrm{DE} + \mathrm{IB} \rightleftharpoons \mathrm{TE}, \quad 2\,\mathrm{IB} \longrightarrow \mathrm{DIB}.$$

Apart from these components, water (W) exists in the system as the solvent. The stoichiometric constants ν_{ij} of the 7 species (1=G, 2=IB, 3=ME, 4=DE, 5=TE, 6=DIB, 7=W, subscripted by i) in the 4 reactions (subscripted by j) are given as

$$\begin{aligned}
&\nu_{11} = -1,\ \nu_{21} = -1,\ \nu_{31} = 1,\ \nu_{22} = -1,\ \nu_{32} = -1,\ \nu_{42} = 1 \\
&\nu_{23} = -1,\ \nu_{43} = -1,\ \nu_{53} = 1,\ \nu_{24} = -2,\ \nu_{64} = 1,\ \text{other } \nu_{ij} = 0.
\end{aligned} \tag{C.15}$$

The rates of the 4 reactions, in terms of total extent per unit time per mole of active catalytic site ($\mathrm{kmol} \cdot \mathrm{h}^{-1} \cdot (\mathrm{mol\ H})^{-1}$), are

$$\begin{aligned}
r_1 &= \frac{k_1(a_1a_2 - a_3/K_{\mathrm{e}1})}{(1 + K_{\mathrm{a}1}a_1 + K_{\mathrm{a}2}a_2)^2},\ r_2 = \frac{k_2(a_2a_3^2 - a_4/K_{\mathrm{e}2})}{1 + K_{\mathrm{a}1}a_1 + K_{\mathrm{a}2}a_2}, \\
r_3 &= \frac{k_3(a_2a_4^2 - a_5/K_{\mathrm{e}3})}{1 + K_{\mathrm{a}1}a_1 + K_{\mathrm{a}2}a_2},\ r_4 = \frac{k_4a_2^2}{(1 + K_{\mathrm{a}1}a_1 + K_{\mathrm{a}2}a_2)^2},
\end{aligned} \tag{C.16}$$

where k and K_{e} stands for the rate constants, related to pre-exponential factors and activation energies, and equilibrium constants, related to the enthalpy and entropy changes, respectively. The adsorption equilibrium constants K_{a} follow similar rules:

$$\begin{aligned}
&k_j = k_j^\circ \exp(-E_j/RT),\ K_{\mathrm{e}j} = \exp(-\Delta H_j/RT + \Delta S/R), \\
&K_{\mathrm{a}i} = K_{\mathrm{a}i}^\circ \exp(-\Delta H_{\mathrm{a}i}/RT).
\end{aligned} \tag{C.17}$$

The reaction rate for each species is therefore

$$R_i = \sum_{j=1}^{4} \nu_{ij} r_j V \rho_{\mathrm{cat}} q, \quad V = M \sum_{j=1}^{7} \frac{w_j x_j}{\rho_j}. \tag{C.18}$$

with q being the quantity of active sites in moles per mass of catalyst, ρ_{cat} the density of catalyst, and V the volume of reacting liquid linked to the total molar holdup M and molar fractions x_j by densities ρ_j and molar weights w_j. The true values of these constants are listed in Table C.1. We assume that the thermodynamic constants E_j,

Table C.1: Parameters related to the reaction kinetics

E_1	21.3 kJ/mol	k_1°	2.56×10^2 kmol/(h · mol H)
E_2	39.1 kJ/mol	k_2°	1.14×10^5 kmol/(h · mol H)
E_3	25.6 kJ/mol	k_3°	7.52×10^3 kmol/(h · mol H)
E_4	39.9 kJ/mol	k_4°	5.59×10^3 kmol/(h · mol H)
ΔH_1	−49.4 kJ/mol	ΔS_1	−119.1 J/(mol · K)
ΔH_2	−6.0 kJ/mol	ΔS_2	−36.9 J/(mol · K)
ΔH_3	−27.1 kJ/mol	ΔS_3	−89.8 J/(mol · K)
ΔH_{a1}	−7.6 kJ/mol	K_{a1}°	2.51×10^{-1}
ΔH_{a2}	−12.3 kJ/mol	K_{a2}°	2.24×10^{-4}
w_1	92 kg/mol	ρ_1	1261 kg/m^3
w_2	56 kg/mol	ρ_2	588 kg/m^3
w_3	148 kg/mol	ρ_3	1015 kg/m^3
w_4	204 kg/mol	ρ_4	920 kg/m^3
w_5	260 kg/mol	ρ_5	880 kg/m^3
w_6	112 kg/mol	ρ_6	718 kg/m^3
w_7	18 kg/mol	ρ_7	1000 kg/m^3
ρ_{cat}	60 kg/m^3	q	4.7 mol H/kg

ΔH_j, ΔS_j, ΔH_{ai} are known exactly, while the pre-exponential factors k_j° $(j = 1, 2, 3, 4)$ for the reaction rates and K_{ai}° $(i = 1, 2)$ for adsorption equilibria are to be estimated.

For the activity of the chemical species in the multicomponent liquid mixture, the NRTL model is used:

$$\begin{aligned} a_i =& \gamma_i x_i, \\ \ln \gamma_i =& \frac{\sum_j x_j \frac{A_{ji}}{RT} \exp\left(-\frac{\alpha_{ji} A_{ji}}{RT}\right)}{\sum_j x_j \exp\left(-\frac{\alpha_{ji} A_{ji}}{RT}\right)} + \sum_j \frac{x_j \exp\left(-\frac{\alpha_{ij} A_{ij}}{RT}\right)}{\sum_k x_k \exp\left(-\frac{\alpha_{ik} A_{ik}}{RT}\right)} \\ & \times \left(\frac{A_{ij}}{RT} - \frac{\sum_k x_k A_{kj} \exp\left(-\frac{\alpha_{kj} A_{kj}}{RT}\right)}{\sum_k x_k \exp\left(-\frac{\alpha_{kj} A_{kj}}{RT}\right)} \right), \quad i = 1, \ldots, 7. \end{aligned} \tag{C.19}$$

The parameters in the NRTL model is listed in Table C.2. For $i = 1, \ldots, 7$, $A_{ii} = 0$ and $\alpha_{ii} = 0$. For $i, j = 1, \ldots, 7$, $\alpha_{ij} = \alpha_{ji}$. We assume that the NRTL model is unknown, and in the identification the mixture is considered ideal, i.e., $\gamma_i = 1$, $\forall i$.

For simplicity we consider an isothermal reactor with constant temperature $T =$ 353 K and constant molar holdup M. The inlet stream to the reactor is mixed by 4 streams, in which one is the fresh feed of pure isobutene, whose molar flow rate is

Table C.2: Parameters in the NRTL model

A_{12}	6000.6295	A_{21}	7790.3843	α_{12}	0.2
A_{13}	5093.9878	A_{31}	−261.0596	α_{13}	0.2
A_{14}	15394.2024	A_{41}	3470.2636	α_{14}	0.2
A_{15}	18947.6060	A_{51}	9748.1650	α_{15}	0.2
A_{16}	10108.9095	A_{61}	16721.0340	α_{16}	0.2
A_{17}	−2280.9459	A_{71}	2145.9265	α_{17}	1.011
A_{23}	10225.3886	A_{32}	−2579.3354	α_{23}	0.2
A_{24}	−3867.1740	A_{42}	−6172.8956	α_{24}	0.2
A_{25}	−3867.1740	A_{52}	−6172.8956	α_{25}	0.2
A_{26}	−735.1239	A_{62}	472.8172	α_{26}	0.329
A_{27}	11654.4821	A_{72}	11799.1457	α_{27}	0.255
A_{34}	−4605.9560	A_{43}	8587.5306	α_{34}	0.2
A_{35}	7737.8398	A_{53}	−1327.7458	α_{35}	0.2
A_{36}	2163.8848	A_{63}	10377.8670	α_{36}	0.275
A_{37}	−3457.2938	A_{73}	13410.9808	α_{37}	0.392
A_{45}	9937.7242	A_{54}	−3728.8290	α_{45}	0.2
A_{46}	−1867.1581	A_{64}	8318.6558	α_{46}	0.286
A_{47}	−168.9405	A_{74}	20784.1686	α_{47}	0.345
A_{56}	−3141.0292	A_{65}	6209.7266	α_{56}	0.398
A_{57}	−386.1022	A_{75}	20784.9169	α_{57}	0.202
A_{67}	9981.2064	A_{76}	20784.9169	α_{67}	0.2

considered as the manipulated input. The flow rates and compositions of other 3 streams to the reactor are fixed. The molar flow rate of the outlet stream is adjusted accordingly to keep the molar holdup constant. Hence the dynamic model has 6 states standing for the molar fractions of the previous 6 components (with the 7th one dependent). Let the controlled output be the total reaction rate $R = \sum_{i=1}^{7} R_i$, namely the difference between the molar flow rates of the inlet stream and the outlet, which are assumed measurable. Hence the model is

$$\dot{x}_i = \frac{F_0 + u}{M}(x_{0,i} - x_i) + \sum_{l=1}^{3} \frac{F_l}{M}(x_{l,i} - x_i) + \frac{R_i(x)}{M}, \quad y = \sum_{i=1}^{7} R_i(x). \tag{C.20}$$

Under the nominal input $u = 0$, the steady states along with the parameters are given in Table C.3. The approximate model is the one with R_i in the above accurate model substituted with an ideal mixture (with all $\gamma_i = 1$) and kinetic coefficients $k_1^\circ/10^4$, $k_2^\circ/10^3$, $k_3^\circ/10^3$, $k_4^\circ/10^3$, K_{a1}, $K_{a2}/10^{-4}$ (scaled by orders of magnitudes that are assumed

Table C.3: Parameters and steady states in the reactor dynamics

x_1	0.0035	x_2	0.7267	x_3	0.0436
x_4	0.1045	x_5	0.0156	x_6	0.0887
$x_{1,1}$	0.0000	$x_{1,2}$	0.9198	$x_{1,3}$	0.0000
$x_{1,4}$	0.0000	$x_{1,5}$	0.0000	$x_{1,6}$	0.0584
$x_{2,1}$	0.0000	$x_{2,2}$	0.9731	$x_{2,3}$	0.0000
$x_{2,4}$	0.0000	$x_{2,5}$	0.0000	$x_{2,6}$	0.0192
$x_{3,1}$	0.7674	$x_{3,2}$	0.0000	$x_{3,3}$	0.2294
$x_{3,4}$	0.0001	$x_{3,5}$	0.0000	$x_{3,6}$	0.0000
F_1	185.4900	F_2	1.7267	F_3	40.1762

to be known) left as 5 unknown parameters, represented as $\hat{R}(x|\theta)$.