Splitting Methods for Distributed Optimization and Control

THÈSE N^{O} 7041 (2016)

PRÉSENTÉE LE 27 MAI 2016

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE D'AUTOMATIQUE 3
PROGRAMME DOCTORAL EN GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Ye PU

acceptée sur proposition du jury:

Dr A. Karimi, président du jury
Prof. C. N. Jones, Prof. M. N. Zeilinger, directeurs de thèse
Prof. F. Allgöwer, rapporteur
Prof. P. Patrinos, rapporteur
Prof. P. Frossard, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE
Suisse
2016

May 11, 2016

Abstract

This thesis contributes towards the design and analysis of fast and distributed optimization algorithms based on splitting techniques, such as proximal gradient methods or alternation minimization algorithms, with the application of solving model predictive control (MPC) problems.

The first part of the thesis focuses on developing an efficient algorithm based on the fast alternating minimization algorithm to solve MPC problems with polytopic and second-order cone constraints. Due to the requirement of bounding the online computation time in the context of real-time MPC, complexity bounds on the number of iterations to achieve a certain accuracy are derived. In addition, a discussion of the computation of the complexity bounds is provided. To further improve the convergence speed of the proposed algorithm, an off-line pre-conditioning method is presented for MPC problems with polyhedral and ellipsoidal constraints.

The inexact alternating minimization algorithm, as well as its accelerated variant, is proposed in the second part of the thesis. Different from standard algorithms, inexact methods allow for errors in the update at each iteration. Complexity upper-bounds on the number of iterations in the presence of errors are derived. By employing the complexity bounds, sufficient conditions on the errors, which guarantee the convergence of the algorithms, are presented. The proposed algorithms are applied for solving distributed optimization problems in the presence of local computation and communication errors, with an emphasis on distributed MPC applications. The convergence properties of the algorithms for this special case are analysed.

Motivated by the complexity upper-bounds of the inexact proximal gradient method, two distributed optimization algorithms with an iteratively refining quantization design are proposed for solving distributed optimization problems with a limited communication data-rate. We show that if the parameters of the quantizers satisfy certain conditions, then the quantization error decreases linearly, while at each iteration only a fixed number of bits is transmitted, and the convergence of the distributed algorithms is guaranteed. The proposed methods are further extended to distributed optimization problems with time-varying parameters.

Keywords: Convex optimization, first-order optimization algorithms, splitting methods, alternation minimization algorithm (AMA), fast alternation minimization algorithm (FAMA), inexact alternation minimization algorithm (IAMA), inexact fast alternation minimization algorithm (IFAMA), distributed optimization, quantization design, model predictive control and distributed model predictive control.

Résumé

Cette thèse a pour sujet la conception et l'analyse des algorithmes d'optimisation rapides et distribués basés sur les techniques dites de "splitting", telles que les méthodes de gradient proximal ou des algorithmes de minimisation alternée, appliquées à la résolution de problèmes de contrôle prédictif (MPC).

La première partie de la thèse porte sur le développement d'un algorithme efficace basé sur l'algorithme de minimisation alternée rapide (fast AMA) pour résoudre les problèmes de type MPC avec des contraintes polytopiques et coniques du second ordre. En raison de l'exigence de limiter le temps de calcul en ligne, par exemple pour une implementation temps-réel de MPC, des bornes sur le nombre d'itérations pour obtenir une certaine précision sont calculées. En outre, une discussion sur le calcul de ces bornes est détaillée. Pour améliorer encore la vitesse de convergence de l'algorithme proposé, une méthode de pré-conditionnement hors-ligne est présentée pour des problèmes MPC avec contraintes polyédriales et ellipsoïdales.

L'algorithme de minimisation alternée inexacte, ainsi que sa variante accélérée, sont proposés dans la deuxième partie de la thèse. A la différence des algorithmes standards, les méthodes inexactes autorisent des erreurs à chaque itération. Des bornes supérieures sur le nombre d'itérations en présence d'erreurs sont calculées. En utilisant ces bornes, des conditions suffisantes sur les erreurs qui garantissent la convergence, sont présentées. Les algorithmes proposés sont appliqués pour résoudre des problèmes d'optimisation distribués en présence d'erreurs de calcul et de communication locales, en particulier pour des problèmes de MPC distribués. Les propriétés de convergence des algorithmes pour ce cas particulier sont analysées.

Motivés par les bornes de complexité de la méthode du gradient proximal inexact, deux algorithmes d'optimisation distribués avec une quantification raffinée itérativement sont proposés pour résoudre des problèmes d'optimisation distribués avec une bande passante limitée. Nous montrons que si les paramètres des quantificateurs satisfont certaines conditions, l'erreur de quantification décroît linéairement, même si à chaque itération un nombre fixe de bits est transmis, et la convergence des algorithmes est garantie. Les méthodes proposées sont étendues à des problèmes d'optimisation distribués avec des paramètres variables dans le temps.

Keywords: Optimisation convexe, algorithmes d'optimisation du premier ordre, méthodes de "splitting", algorithme de minimisation alternée (AMA), algorithme de minimisation alternée rapide (FAMA), algorithme de minimisation alternée inexacte (IAMA), algorithme de minimisation alternée inexacte rapide (IFAMA), optimization distribuée, quantification, commande prédictive et commande prédictive distribuée

Acknowledgements

This thesis could not have been accomplished without the guidance of my supervisor, Prof. Colin Neil Jones. Colin has his unique and admirable way of thinking about research problems and assessing the values and potentials of the results, which has to a great extent influenced my research style. I also learned a lot from his superb presentation and communication skills. During these years, he gives me complete freedom so I can pursue research directions that interest me, and offers invaluable advice and encouragement after usual frustrations. I would like to express my deepest respect and gratitude to him.

My co-supervisor Melanie Nicole Zeilinger plays an equally pivotal role in the process of my PhD study. During the four years, Melanie provides countless invaluable advices, not only regarding researches but also on how to plan my future career and life. Moreover, I am very lucky to be her first official PhD student. I would like to extend my deepest respect and gratitude to Melanie for all her guidance, supports and encouragement.

It was a great honor to have Frank Allgöwer, Panos Patrinos, Pascal Frossard and Alireza Karimi on my thesis committee, and I am very thankful for their careful reading of the thesis and helpful advices and comments on the thesis.

The LA family leaves me many memorable moments in these four years. I would like to express my gratitude to Dominique Bonvin, Roland Longchamp and Alireza Karimi for creating and maintaining a very friendly and research-conducive environment in LA. My gratitude also goes to our secretaries Ruth Benassi and Khava Isaeva for their friendly and helpful administrative help and Christophe Salzmann for his precious help for laboratory setups. I also appreciated various discussion with Philippe Müllhaupt, Ioannis Lymperopoulos and Timm Faulwasser, from which I learned many interesting things.

Many thanks to my lovely officemates, Tomasz, Georgios, Martand, Faran, Francisco, Andrea, Jean-Hubert, Milan, Altug, Harsh, Sanket and Luca for their support and accompanies. It is always very enjoyable to talk with them technically and non-technically. Thanks to Tomasz's home-made cake, Martand's daily lunch reminder, Jean-Hubert's french jokes and every moments we shared in the office.

Finally, I want to thank my family for their continuous supports throughout the years. This thesis is dedicated to Jingge, for his love and patience which makes me a very happy person every day.

Contents

1 Introduction

Outline and Contribution

Due to the increasing computational power of computers and the new development of optimization algorithms, the optimization-based control methods are becoming powerful and promising tools for applications with fast and large dynamics and complex control specifications. One important optimization-based method is Model Predictive Control. The strength of Model Predictive Control (MPC) is that it optimizes an objective over a finite time-horizon in the future and permits constraints on the states and control inputs to be integrated into the optimal controller design. However, the cost is that at each sampling time an optimization problem needs to be solved, which has traditionally restricted MPC to applications with slow dynamics and long sampling times. This limitation has given rise to an increasing interest in the development of new methods to either improve the on-line computation, driven by the increase in computational power of hardware and newly developed optimization techniques, or to approximate the optimum with a sub-optimal but stabilizing solution.

One technique to reduce the on-line computation is multi-parametric programming, which pre-computes the solution for every state off-line, see [1] for more details and references. [2] presents a method combining explicit MPC with on-line computation. However, all explicit and approximate explicit methods are limited to small-scale problems. For medium and larger scale MPC problems, on-line computation methods are used. Various approaches have been proposed to improve the on-line computation time. The authors in [3] employ the fast gradient method introduced in [4] to solve MPC problems with box constraints on inputs. Efficient implementations of interior-point methods have been studied in [5] and [6]. Accelerated gradient methods with dual decomposition are investigated in [7] and in [8] in the context of distributed MPC. [9] and [10] present efficient active set methods for MPC.

Splitting methods, which are also known as alternating direction methods, offer a powerful tool for general mathematical programming and optimization, see e.g. [11], [12] and [13]. Their efficiency results from splitting a complex convex minimization problem into simple sub-problems and solving them in an alternating manner. For a problem with multiple objectives, the main strategy is not to compute the descent

direction of the sum of several objectives, but to take a combination of the descent directions of each objective. This can significantly reduce computation time, in particular when the objectives have different properties, for instance one being a quadratic function, one an l_1-norm and one involving indicator functions, which originate from constraints. In Chapter 4, we will propose efficient optimization algorithms based on splitting methods for solving MPC problems.

Networked systems, e.g. power grids and social networks, require vastly more managing and stabilizing services at different levels, due to the increasing number of components. To meet some advanced specifications, e.g. constraint satisfaction and low economical cost, optimal control techniques seem to be a promising approach. One challenge is to design and implement an optimal controller at the network level. As the system network gets larger and more complex, solving an optimal control problem in a centralized way becomes difficult, since it requires full communication to collect information from each sub-system, as well as the computational power to solve the global problem in one central entity.

A promising concept to avoid this problem is to use distributed model predictive control techniques to solve network-level control problems, requiring only neighbour-to-neighbour communication. Distributed MPC, see e.g. [14], [15] and [16], compared to centralized MPC (with full communication), computes the control input based on local measurements and communication between neighbouring systems. The drawback is that distributed MPC may provide a sub-optimal control solution with respect to centralized MPC. Therefore, distributed MPC represents an interesting trade-off between centralized and decentralized MPC (without communication between sub-systems), in other words a trade-off between control specification and communication costs. The initial studies [17] and [18] have shown that distributed MPC is a promising tool for many control applications, e.g. frequency and voltage control in power grids. From the implementation perspective, a key challenge to apply a distributed MPC controller, in particular to systems with fast dynamics, is to develop an efficient distributed optimization algorithm to synthesize the controller. However, in practice distributed optimization algorithms may suffer from inexact local solutions and unreliable communications, see e.g. [19], [12] and [8]. The resulting inexact updates in the distributed optimization algorithms affect the convergence properties, and can even cause divergence of the algorithm.

In this thesis, we will study inexact splitting methods and aim at answering the following two questions: 1) How do errors affect the algorithms and under which conditions can convergence still be guaranteed; 2) How can we develop efficient distributed algorithms subject local computation limitations and communication constraints. Seminal work on inexact optimization algorithms includes [20], [21] and [22]. In [22], an inexact proximal-gradient method, as well as its accelerated version, are introduced. The conceptual idea is to allow errors in the calculation of the gradient and in the proximal minimization. The results in [22] show convergence properties of the inexact proximal-gradient method and provide conditions on the errors, under which convergence of the algorithm can be guaranteed. We will develop new distributed optimization algorithms based on inexact splitting techniques in Chapters 5, 6 and 7.

Chapter 2 - 3: Background

In the background chapters, we introduce the relevant definitions and results for convex optimization, splitting algorithms, inexact splitting algorithms, model predictive control and distributed model predictive control.

Chapter 4: Complexity Certification of the Fast Alternating Minimization Algorithm for Linear Model Predictive Control

In this chapter, the fast alternating minimization algorithm (FAMA) is proposed to solve model predictive control (MPC) problems with polytopic and second-order cone constraints. Two splitting strategies for MPC problems are presented. Both of them satisfy the assumptions of FAMA and result in efficient implementations by reducing each iteration of FAMA to simple operations. We derive computational complexity certificates for both splitting strategies, by providing complexity upper-bounds on the number of iterations required to provide a certain accuracy of the dual function value and, most importantly, of the primal solution. This is of particular relevance in the context of real-time MPC in order to bound the required online computation time. We further address the computation of the complexity bounds, requiring the solution of a non-convex minimization problem. For MPC problems with polyhedral and ellipsoidal constraints, an off-line preconditioning method is presented to further improve the convergence speed of FAMA by reducing the complexity bound and enlarging the step-size of the algorithm. Finally, we demonstrate the performance of FAMA compared to other splitting methods using a quadrotor example.

Chapter 4 is based on the following conference paper and technical report. The majority of the text and content in Chapter 4 has appeared in the following two papers.

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Fast Alternating Minimization Algorithm for Model Predictive Control. 19th IFAC World Congress, Cape Town, 8-24, 2014.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Complexity Certification of the Fast Alternating Minimization Algorithm for Linear Model Predictive Control, March 2016, accepted for publication in to IEEE Transactions on Automatic Control

Chapter 5: Inexact Alternating Minimization Algorithm for Distributed Optimization with Distributed MPC Application

In this chapter, we propose the inexact alternating minimization algorithm (inexact AMA), which allows inexact iterations in the algorithm, and its accelerated variant, called the inexact fast alternating minimization algorithm (inexact FAMA). We show that inexact AMA and inexact FAMA are equivalent to the inexact proximal-gradient method and its accelerated variant applied to the dual problem. Based on this equivalence, we derive complexity upper-bounds on the number of iterations for the inexact algorithms. We apply inexact AMA and inexact FAMA to distributed optimization problems, with an emphasis on distributed MPC applications, and

show the convergence properties for this special case. By employing the complexity upper-bounds on the number of iterations, we provide sufficient conditions on the inexact iterations for the convergence of the algorithms. We further study the special case of quadratic local objectives in the distributed optimization problems, which is a standard form of a distributed MPC problem. For this special case, we allow local computational errors at each iteration. By exploiting a warm-starting strategy and the sufficient conditions on the errors for convergence, we propose an on-line approach to certify the number of iterations for solving local problems, which guarantees that the local computational errors satisfy the sufficient condition and the inexact distributed optimization algorithm converges to the optimal solution.

Chapter 5 is based on the following conference paper and technical report. The majority of the text and content in Chapter 5 has appeared in the following two documents.

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Inexact Fast Alternating Minimization Algorithm for Distributed Model Predictive Control. 53rd IEEE Conference on Decision and Control, Los Angeles, 2014, December 15-17, 2014.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Inexact Alternating Minimization Algorithm for Distributed Optimization with an Application to Distributed MPC, March. 2016, submitted to IEEE Transactions on Automatic Control

Chapter 6: Quantization Design for Distributed Optimization

In this chapter, we consider the problem of solving a distributed optimization problem using a distributed computing platform, where the communication in the network is limited: each node can only communicate with its neighbours and the channel has a limited data-rate. A common technique to address the latter limitation is to apply quantization to the exchanged information. We propose two distributed optimization algorithms with an iteratively refining quantization design based on the inexact proximal gradient method and its accelerated variant. We show that if the parameters of the quantizers, i.e. the number of bits and the initial quantization intervals, satisfy certain conditions, then the quantization error is bounded by a linearly decreasing function and the convergence of the distributed algorithms is guaranteed. Furthermore, we prove that after imposing the quantization scheme, the distributed algorithms still exhibit a linear convergence rate, and show complexity upper-bounds on the number of iterations to achieve a given accuracy. Finally, we demonstrate the performance of the proposed algorithms and the theoretical findings for solving a distributed optimal control problem.

Chapter 6 is based on the following conference paper and technical report. The majority of the text and content in Chapter 6 has appeared in the following two papers.

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Unconstrained Distributed Optimization. American Control Conference, Chicago, 2015.

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Distributed Optimization, March 2016, accepted for publication in IEEE Transactions on Automatic Control.

Chapter 7: Quantization Design for Distributed Optimization with Time-Varying Parameters

We consider the problem of solving a sequence of distributed optimization problems with time-varying parameters and communication constraints, i.e. only neighbour-to-neighbour communication and a limited amount of information exchanged. By extending the results in Chapter 6 and employing a warm-starting strategy, we propose an on-line algorithm for solving optimization problems under the given constraints and show that there exists a trade-off between the number of iterations for solving each problem in the sequence and the accuracy achieved by the algorithm. For a given accuracy ϵ, we can find a number of iterations K, which guarantees that for the sequential realization of the parameter, the sub-optimal solution given by the algorithm satisfies the accuracy. We apply the method to solve a distributed model predictive control problem by considering the state measurement at each sampling time as the time-varying parameter and show that the simulation supports the theoretical results.

Chapter 7 is based on the following conference paper. The majority of the text and content in Chapter 7 has appeared in the following paper.

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Distributed Optimization with time-varying parameters. 54th IEEE Conference on Decision and Control, Osaka, 2015.

Additional Publications

The following papers were published or submitted during the Ph.D. study and are not chosen to be included in this thesis.

- M.N. Zeilinger, **Y. Pu**, S. Riverso, G. Ferrati-Trecate, C. N. Jones. Plug and Play Distributed Model Predictive Control based on Distributed Invariance and Optimization. 52nd IEEE Conference on Decision and Control, Florence, 2013

- G. Stathopoulos, A. Szücs, **Y. Pu** and C. N. Jones. Splitting methods in control. 13th European Control Conference, Strasbourg, June 24-27 2014.

- F. F. C. Rego, **Y. Pu**, A. P. Aguiar and C. N. Jones. A Consensus Algorithm for Networks with Process Noise and Quantization Error. 53rd Annual Allerton Conference 2015.

- F. F. C. Rego, **Y. Pu**, A. P. Aguiar and C. N. Jones. Design of a Distributed Quantized Luenberger Filter for Bounded Noise. American Control Conference, Boston, 2016.

- G. Stathopoulos, H. Shukla, A. Szücs, **Y. Pu** and C. N. Jones. Operator splitting methods in control. Sep. 2015 submitted to Foundations and Trends in Systems and Control.
- L. Ferranti, **Y. Pu**, C. N. Jones, and T. Keviczky. Asynchronous Splitting Design for Model Predictive Control. submitted to 55th IEEE Conference on Decision and Control, 2016

2 Splitting Methods

In this chapter, relevant definitions and results for this thesis will be introduced.

Notations for vector and matrices

Let C be a matrix. $\rho(C)$ denotes the largest eigenvalue of C^TC. For the case that C is a positive definite matrix, $\lambda_{min}(C)$ denotes the smallest eigenvalue of C. Let C_1 and C_2 be two matrices. The Kronecker product of C_1 and C_2 is denoted by $C_1 \otimes C_2$. The operators max, $\leq$ and $\geq$ are defined to work on vectors as well as scalars. For vectors, the operators are defined to be element-wise. Let $x \in \mathbb{R}^n$ be a vector. $\|x\|$ and $\|x\|_\infty$ denote the l_2 and infinity norms of x, respectively. Note that $\|x\|_\infty \leq \|x\|_2 \leq \sqrt{n}\|x\|_\infty$. We refer to [23] for details on the definitions and properties above.

2.1 Convex optimization

In this section, we will introduce definitions and results in convex optimization. We refer to [24], [25] and [26] for details on the definitions and properties below.

Definition 2.1. *(Convex Set) A set $\mathbb{C} \subseteq \mathbb{R}^n$ is convex, if the line segment between any two points in $\mathbb{C}$ lies in $\mathbb{C}$, i.e., if for any x_1, x_2 and any θ with $0 \leq \theta \leq 1$, we have*

$$\theta x_1 + (1-\theta)x_2 \in \mathbb{C}.$$

Definition 2.2. *(Projection) The projection of any point $x \in \mathbb{R}^n$ onto the set $\mathbb{C}$ is defined by*

$$\text{Proj}_{\mathbb{C}}(x) := \text{argmin}_{y \in \mathbb{C}} \ \|y - x\|.$$

Definition 2.3. *(Cone and Convex Cone) A set $\mathbb{C}$ is called a cone, if for any $x \in \mathbb{C}$ and $\theta \geq 0$ we have $\theta x \in \mathbb{C}$. A set $\mathbb{C}$ is a convex cone, if for any x_1, $x_2 \in \mathbb{C}$ and θ_1, $\theta_2 \geq 0$, we have*

$$\theta_1 x_1 + \theta_2 x_2 \in \mathbb{C}.$$

Definition 2.4. *(Polar Cone) Let $\mathbb{C}$ be a convex cone. The polar cone of $\mathbb{C}$ is defined as*

$$\mathbb{C}^{\circ} := \{w \mid v^T w \leq 0,\ \forall v \in \mathbb{C}\}.$$

Definition 2.5. *(Dual Cone) Let $\mathbb{C}$ be a convex cone. The dual cone of $\mathbb{C}$ is defined as*

$$\mathbb{C}^{\star} := \{w \mid v^T w \geq 0,\ \forall v \in \mathbb{C}\}.$$

Definition 2.6. *(Self-dual Cone) A convex cone $\mathbb{C}$ is called self-dual, if $\mathbb{C} = \mathbb{C}^{\star}$.*

Definition 2.7. *(Normal Cone) Let $\mathbb{C}$ be a non-empty, closed convex set. The normal cone to the set $\mathbb{C}$ at the point $x \in \mathbb{C}$ is defined as*

$$\mathbb{N}_{\mathbb{C}}(x) := \{w \mid w^T(v - x) \leq 0,\ \forall v \in \mathbb{C}\}.$$

Definition 2.8. *(Convex Function) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is convex, if the domain of the function **dom** f is a convex set and if for all x, $y \in$ **dom** f, and all $0 \leq \theta \leq 1$, it holds that*

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta) f(y).$$

Definition 2.9. *(Epigraph) The epigraph of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as*

$$\textbf{\textit{epi}}\ f := \{(x, t) \mid x \in \textbf{\textit{dom}}\ f,\ f(x) < t\}.$$

Definition 2.10. *(Closed Function) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is closed, if the epigraph of f is a closed set for all $x \in$ **dom** f.*

Definition 2.11. *(Sub-gradient and Sub-differential) A sub-gradient to a convex function f at $x \in$ **dom** f is any vector $\xi(x)$ such that*

$$f(y) \geq f(x) + \xi(x)^T (y - x), \tag{2.1}$$

*for all $y \in$ **dom** f. The set of vectors $\xi(x)$ that satisfy (2.1) at x is denoted by $\partial f(x)$ and is called the sub-differential of f at x.*

Definition 2.12. *(Strongly Convex Function) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is strongly convex if there exists a constant $\sigma_f > 0$ such that for all x, $y \in$ **dom** f, we have*

$$\theta f(x) + (1 - \theta) f(y) - f(\theta x + (1 - \theta)y) \geq \sigma_f \cdot \theta(1 - \theta)\|x - y\|^2$$

for all $0 \leq \theta \leq 1$. The constant σ_f is named as the convexity modulus the function f.

Definition 2.13. *(Strongly Monotonic Function) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is strongly monotonic if there exists a constant $\sigma_f > 0$ such that for all x, $y \in$ **dom** f, we have*

$$\langle p - q, x - y \rangle \geq \sigma_f \|x - y\|^2,$$

where $p \in \partial f(x)$, $q \in \partial f(y)$ and $\partial(\cdot)$ denotes the sub-differential of the function at a given point. The constant σ_f is called the convexity modulus the function f.

Remark 2.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is a strongly convex function, if and only if f is a strongly monotonic function.*

Definition 2.14. *(Lipschitz Continuity) A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ is called Lipschitz continuous with Lipschitz constant $L(f)$ on a subset $\mathbb{C}$ of $\boldsymbol{dom}\, f$, if for all $x, y \in \mathbb{C}$, we have*

$$\|f(x) - f(y)\| \leq L(f)\|x - y\|.$$

Definition 2.15. *(Conjugate Function) Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ be a convex function. The conjugate function of f is defined as*

$$f^\star(y) = \sup_{x \in \boldsymbol{dom}\, f} (y^T x - f(x)).$$

Remark 2.2. *We note that for a conjugate function, it holds that $q \in \partial f(p) \Leftrightarrow p \in \partial f^\star(q)$.*

Remark 2.3. *If f is a strongly convex function with the convexity modulus σ_f, then the gradient of the conjugate function of f is Lipschitz continuous with a Lipschitz constant $L(\nabla f^\star) = \sigma_f^{-1}$. For more details, see in Theorem 4.2.1 in [27]*

Definition 2.16. *(Indicator Function) The indicator function on the set $\mathbb{C}$ is defined as*

$$I_{\mathbb{C}}(x) := \begin{cases} 0 & \text{if } x \in \mathbb{C} \\ \infty & \text{if } x \notin \mathbb{C}. \end{cases} \tag{2.2}$$

Remark 2.4. *We note that the indicator function defined in (2.2) is a convex function.*

Definition 2.17. *(Proximity Operator) The proximity operator is defined as*

$$\text{prox}_f(y) = \text{argmin}_x\, f(x) + \frac{1}{2}\|x - y\|^2 \ . \tag{2.3}$$

Remark 2.5. *We note the following equivalence:*

$$x^\star = \text{prox}_f(y) \Longleftrightarrow y - x^\star \in \partial f(x^\star) \tag{2.4}$$

Definition 2.18. *(Linear and Sub-linear Convergence Rates) A sequence $\{e^k\}_{k=0}^\infty$ converges linearly to zero if there exist constant $0 < q < 1$ and $C > 0$ such that*

$$e^k \leq C \cdot q^k \tag{2.5}$$

for all $k = 0, 1, \ldots$. The constant q denotes the convergence ratio.

A sequence $\{e^k\}_{k=0}^\infty$ converges sub-linearly to zero, if it does not converge linearly. Examples for sub-linearly converging sequences include

$$e^k \leq \frac{K}{\sqrt{k+1}}, \quad e^k \leq \frac{K}{k+1}, \quad e^k \leq \frac{K}{(k+1)^2}, \tag{2.6}$$

where K is a positive constant. The sub-linear convergence rate examples are also denoted as as $O(\frac{1}{\sqrt{k}})$, $O(\frac{1}{k})$ and $O(\frac{1}{k^2})$, respectively.

2.2 Splitting methods

In this section, we will introduce splitting methods, including proximal-gradient method, alternating minimization algorithm, alternating direction method of multipliers, as well as their accelerated and inexact variants.

2.2.1 Proximal-gradient method and its accelerated variant

Proximal-gradient method (PGM)

In this section, we will introduce the proximal-gradient method (PGM), which is also called the iterative shrinkage-thresholding algorithm (ISTA) in [28]. It addresses optimization problems of the form given in Problem 2.1 and requires Assumption 2.1 for sub-linear convergence, and Assumption 2.2 for linear convergence. In this thesis, the convergence of an algorithm indicates the convergence of the difference between the sequence generated by the algorithm and the optimal solution. The PGM algorithm is presented in Algorithm 1.

Problem 2.1.

$$\min_{w \in \mathbb{R}^{n_w}} \quad \Phi(w) = \phi(w) + \psi(w).$$

Assumption 2.1. *We assume that*

- ϕ *is a convex function with Lipschitz continuous gradient with Lipschitz constant* $L(\nabla\phi)$
- ψ *is a convex function, not necessarily smooth.*

Assumption 2.2. *We assume that*

- ϕ *is a strongly convex function with a convexity modulus* σ_ϕ *and has a Lipschitz continuous gradient with Lipschitz constant* $L(\nabla\phi)$*.*
- ψ *is a lower semi-continuous convex function, not necessarily smooth.*

For the case that Assumption 2.2 holds, we denote the condition number of the objective ϕ as

$$\gamma = \frac{\sigma_\phi}{L(\nabla\phi)}. \tag{2.7}$$

Remark 2.6. *Due to the fact that the summation of a strongly convex function with a convexity modulus* σ *and a convex function is still a strongly convex function with the same convexity modulus* σ*, we know that if Assumption 2.2 holds, then the function* Φ *is also strongly convex with a convexity modulus* σ_ϕ*. Since* Φ *is a strongly convex function, the optimal solution* $x^\star$ *of Problem 2.1 is unique.*

Algorithm 1 Proximal-Gradient Method

Require: Require $w^0 \in \mathbb{R}^{n_x}$ and $\tau < \frac{1}{L(\nabla\phi)}$
for $k = 1, 2, \cdots$ **do**
1: $w^k = \text{prox}_{\tau\psi}(w^{k-1} - \tau\nabla\phi(w^{k-1}))$
end for

The following propositions state the convergence property of the proximal-gradient method with different assumptions.

Proposition 2.1 (Theorem 3.1 in [28]). *Let $\{w^k\}$ be generated by PGM defined in Algorithm 1. If Assumption 2.1 holds, then for any $k \geq 0$ we have:*

$$\Phi(w^k) - \Phi(w^\star) \leq \frac{L(\nabla\phi)}{2k}\|w^0 - w^\star\|^2$$

where $\Phi(\cdot)$ is defined in Problem 2.1 and w^0 and $w^\star$ denote the initial point of Algorithm 1 and the optimal solution of Problem 2.1, respectively.

Proposition 2.2 (Proposition 3 in [22] for the case without errors). *Let $\{w^k\}$ be generated by PGM defined in Algorithm 1. If Assumption 2.2 holds, then for any $k \geq 0$ we have:*

$$\|w^k - w^\star\| \leq (1-\gamma)^k \cdot \|w^0 - w^\star\| \ , \tag{2.8}$$

where $\gamma = \frac{\sigma_\phi}{L(\nabla\phi)}$ and w^0 and $w^\star$ denote the initial point of Algorithm 1 and the optimal solution of Problem 2.1, respectively.

Accelerated Proximal-Gradient Method (APGM)

In this section, we introduce an accelerated variant of PGM, named the accelerated proximal-gradient method (APGM) proposed in [28]. APGM is also known as fast iterative shrinkage-thresholding algorithm (FISTA) in [28]. It addresses the same problem class as Problem 2.1 and similarly requires Assumption 2.1 for convergence, and Assumption 2.2 for linear convergence.

Differing from PGM, APGM involves one extra linear update in Algorithm 2. If Assumption 2.1 holds, it improves the convergence rate of the complexity upper-bound from $O(\frac{1}{k})$ to $O(\frac{1}{k^2})$. The sequence β^k is updated as follows: for the case that Assumption 2.1 holds, the sequence β^k is updated as

$$\beta^k = \frac{\alpha^k - 1}{\alpha^{k+1}}, \quad \text{with} \quad \alpha^{k+1} = (1 + \sqrt{4\alpha^{k^2} + 1})/2 \ , \tag{2.9}$$

where the sequence α^k is initialized as $\alpha^0 = 1$; and for the case that Assumption 2.2 is satisfied, the sequence β^k is updated as

$$\beta^k = \frac{1 - \sqrt{\gamma}}{1 + \sqrt{\gamma}} \ , \tag{2.10}$$

with γ in (2.7).

Algorithm 2 Accelerated Proximal-Gradient Method

Require: Initialize $v^1 = w^0 \in \mathbb{R}^{n_w}$ and $\tau < \frac{1}{L(\nabla\phi)}$

for $k = 1, 2, \cdots$ **do**

1: $w^k = \text{prox}_{\tau\psi, \epsilon^k}(v^{k-1} - \tau(\nabla\phi(v^{k-1}) + e^k))$

2: $v^k = w^k + \beta^k(w^k - w^{k-1})$

end for

The following proposition states the convergence property of APGM.

Proposition 2.3 (Theorem 2 in [28]). *Let $\{w_k\}$ be generated by APGM defined in Algorithm 2. If Assumption 2.1 holds and the parameter sequence β^k is updated according to (2.9), then for any $k \geq 1$, we have:*

$$\Phi(w^k) - \Phi(w^\star) \leq \frac{2L(\nabla\phi)}{(k+1)^2}\|w^0 - w^\star\|^2 \ ,$$

where $\Phi(\cdot)$ is defined in Problem 2.1 and w^0 and $w^\star$ denote the starting point of Algorithm 2 and the optimal solution of Problem 2.1, respectively.

Proposition 2.4 (Proposition 4 in [22] for the case without errors). *Let $\{w^k\}$ be generated by APGM defined in Algorithm 2. If Assumption 2.2 holds and the parameter sequence β^k is updated according to (2.10), then for any $k \geq 0$ we have:*

$$\Phi(w^k) - \Phi(w^\star) \leq (1 - \sqrt{\gamma})^k \cdot 2\left(\Phi(w^0) - \Phi(w^\star)\right) \ , \tag{2.11}$$

where $\gamma = \frac{\sigma_\phi}{L(\nabla\phi)}$ and w^0 and $w^\star$ denote the initial point of Algorithm 7 and the optimal solution of Problem 2.1, respectively.

2.2.2 Alternating minimization algorithm and its accelerated variant

Alternating minimization algorithm (AMA)

In [29], a splitting method called the alternating minimization algorithm (AMA) is proposed. It addresses optimization problems of the form given in Problem 2.2 and requires Assumption 2.3 for convergence. Compared to Problem 2.1, Problem 2.2 covers more general optimization problems. It allows for an affine coupling constraint and meanwhile maintains the desired property that the objective of the optimization problem can be split into two functions. The AMA algorithm is presented in Algorithm 3. The AMA algorithm is a dual decomposition based method, therefore we need the Lagrange multiplier in the algorithm denoted by λ.

Problem 2.2.

$$\begin{aligned} \min_{v\in\mathbb{R}^{n_v},\, w\in\mathbb{R}^{n_w}} \quad & f(v) + g(w) \ , \\ s.t. \quad & Av + Bw = c \ . \end{aligned}$$

Assumption 2.3. *We assume that*

- *f is a strongly convex function with the convexity modulus σ_f.*
- *g is a convex function, not necessarily smooth.*

Fast alternating minimization algorithm (FAMA)

In this section, we introduce an accelerated variant of AMA, named the fast alternating minimization algorithm (FAMA) presented in [11]. It addresses the same problem class in Problem 2.2 and similarly requires Assumption 2.3 for convergence.

In Algorithm 4, λ denotes the Lagrange multiplier. Differing from AMA, FAMA involves one extra linear update in Algorithm 4. If Assumption 2.3 holds, it improves the convergence rate of the complexity upper-bound from $O(\frac{1}{k})$ to $O(\frac{1}{k^2})$. The sequence α^k is updated according to the same rule as in (2.9).

Algorithm 3 Alternating minimization algorithm (AMA)

Require: Initialize $\lambda^0 \in \mathbb{R}^{N_b}$, and $\tau < \sigma_f/\rho(A)$
for $k = 1, 2, \cdots$ **do**
1: $v_{k+1} = \text{argmin}_v \quad f(v) + \langle \lambda_k, -Av \rangle$
2: $w_{k+1} = \text{argmin}_w \quad g(w) + \langle \lambda_k, -Bw \rangle + \frac{\tau}{2}\|b - Av_{k+1} - Bw\|^2$
3: $\lambda_{k+1} = \lambda_k + \tau(b - Av_{k+1} - Bw_{k+1})$
end for

Algorithm 4 Fast alternating minimization algorithm (FAMA)

Require: Initialize $\alpha^0 = 1$, $\lambda^0 = \hat{\lambda}^1 = \lambda^{start} \in \mathbb{R}^{N_b}$, and $\tau < \sigma_f/\rho(A)$
for $k = 1, 2, \cdots$ **do**
1: $v^k = \text{argmin}_v \quad f(v) + \left\langle \hat{\lambda}^k, -Av \right\rangle$
2: $w^k = \text{argmin}_w \quad g(w) + \left\langle \hat{\lambda}^k, -Bw \right\rangle + \frac{\tau}{2}\|c - Av^k - Bw\|^2$
3: $\lambda^k = \hat{\lambda}^k + \tau(c - Av^k - Bw^k)$
4: $\alpha^{k+1} = (1 + \sqrt{4\alpha^{k^2} + 1})/2$
5: $\hat{\lambda}^{k+1} = \lambda^k + (\alpha^k - 1)(\lambda^k - \lambda^{k-1})/\alpha^{k+1}$
end for

Remark 2.7. *The step-size constraint of FAMA $\tau < \sigma_f/\rho(A)$ originates from the step-size constraint of FISTA, i.e. $\tau < \frac{1}{L(\nabla F)}$. For the case that the Lipschitz constant is not known or very small, the backtracking step-size rule in [28] can be applied. The idea is to update L at every iteration k as $L_k = \eta^{i_k} L_{k-1}$, where η is a positive constant and i^k is the smallest non-negative integer satisfying*

$$f^\star(A^T\bar{\lambda}^k) \leq \Gamma_{\bar{L}^k}(\bar{\lambda}^k, \lambda^{k-1}) \ ,$$

where $\bar{\lambda}^k$ denotes the one-iteration solution of FAMA based on λ^{k-1}, the step-size is $\bar{L}^k = \eta^{i^k} L^{k-1}$, and

$$\Gamma_{\bar{L}^k}(y_1, y_2) := f^\star(A^T y_2) + \langle y_1 - y_2, A\nabla f^\star(A^T y_2)\rangle + \frac{1}{\bar{L}^k}|y_1 - y_2|^2 + g^\star(B^T y_1) \ .$$

2.2.3 Theoretical properties of AMA and FAMA

In this section, we present the theoretical properties of AMA and FAMA. We first show that AMA and FAMA are equivalent to applying the proximal-gradient method and its accelerated variant to the dual problem of Problem 2.2, respectively. Then, we derive the complexity upper-bounds for AMA and FAMA under Assumption 2.3. Finally, we show the KKT conditions of Problem 2.2, which will be important for the computation of complexity bounds in Section 4.4.

Relationship between AMA and PGM

The Lagrangian of the optimization problem solved by AMA and FAMA in Problem 2.2 is:

$$L(v, w, \lambda) = f(v) + g(w) + \lambda^T(Av + Bw - c) \ . \tag{2.12}$$

The dual problem of Problem 2.2 is expressed as:

Problem 2.3.

$$\max \quad \mathbf{D}(\lambda) = \underbrace{-f^\star(A^T\lambda)}_{-\phi(\lambda)} + \underbrace{b^T\lambda - g^\star(B^T\lambda)}_{-\psi(\lambda)} \ . \tag{2.13}$$

where $\mathbf{D}(\cdot)$ denotes the dual function and λ denotes the Lagrange multiplier. The functions $f^\star$ and $g^\star$ denote the conjugate functions of f and g. Furthermore, the dual problem is equivalent to

$$\min \quad \phi(\lambda) + \psi(\lambda) \ . \tag{2.14}$$

Remark 2.8. *Note that if f is a strongly convex function with the convexity modulus σ_f, then the gradient of the conjugate function of $\phi(\lambda) = f^\star(A^T\lambda)$ is Lipschitz continuous with a Lipschitz constant $L(\nabla\phi) = \sigma_f^{-1} \cdot \rho(A)$.*

Previous work in [29] and [11] has shown that the alternating minimization algorithm is equivalent to applying the proximal-gradient method in Algorithm 1 to the dual problem. The proof of Lemma 2.1 can be found in the proof of Theorem 2 in [11] and the proof in Section 3 in [29].

Lemma 2.1. *If Assumption 2.3 is satisfied and AMA and PGM are initialized with the same dual and primal starting point, then applying AMA in Algorithm 3 to Problem 2.2 is equivalent to applying PGM in Algorithm 1 to the dual problem defined in Problem 2.3 .*

Proof: In order to show the equivalence, we prove that Steps 1, 2 and 3 in Algorithm 3 are equivalent to Step 1 in Algorithm 1, i.e., the following equality holds:

$$\lambda^k = \operatorname{prox}_{\tau\psi}(\lambda^{k-1} - \tau \cdot \nabla\phi(\lambda^{k-1})) \ . \tag{2.15}$$

Step 2 in Algorithm 3 implies:

$$B^T\lambda^{k-1} + \tau B^T(c - Av^k - Bz^k) \in \partial g(z^k) \ .$$

From the property of the conjugate function $p \in \partial f(q) \Leftrightarrow q \in \partial f^\star(p)$, it follows:

$$z^k \in \partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - Av^k - Bz^k)).$$

By multiplying with B and subtracting c on both sides, we obtain:

$$Bz^k - c \in B\partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - Av^k - Bz^k)) - c.$$

By multiplying with τ and adding $\lambda^{k-1} + \tau(c - Av^k - Bz^k)$ on both sides, we get:

$$\begin{aligned}\lambda^{k-1} - \tau Av^k \in \tau B\partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - Av^k - Bz^k)) \\ - \tau c + \lambda^{k-1} + \tau(c - Av^k - Bz^k).\end{aligned}$$

Since $\psi(\lambda) = g^\star(B^T\lambda) - c^T\lambda$, we have $\partial\psi(\lambda) = B\partial g^\star(B^T\lambda) - c$, which implies:

$$\begin{aligned}\lambda^{k-1} - \tau Av^k \in \tau\partial\psi(\lambda^{k-1} + \tau(c - Av^k - Bz^k)) \\ + \lambda^{k-1} + \tau(c - Av^k - Bz^k).\end{aligned}$$

By Step 3 in Algorithm 3, the above equation results in:

$$\lambda^{k-1} - \tau A v^k \in \tau \partial \psi(\lambda^k) + \lambda^k \ .$$

From Step 1 in Algorithm 3 and the property of the conjugate function $p \in \partial f(q) \Leftrightarrow q \in \partial f^\star(p)$, we obtain:

$$\lambda^{k-1} - \tau A \cdot \nabla f^\star(A^T \lambda^k) \in \tau \partial \psi(\lambda^k) + \lambda^k \ .$$

By definition of the function ϕ, we get:

$$\lambda^{k-1} - \tau \cdot \nabla \phi(\lambda^{k-1}) \in \tau \partial \psi(\lambda^k) + \lambda^k \ ,$$

which is equivalent to:

$$\lambda^k = \text{prox}_{\tau\psi}(\lambda^{k-1} - \tau \cdot \nabla \phi(\lambda^{k-1})) \ .$$

■

Relationship between FAMA and APGM

By extending the results in Lemma 2.1, we show the equivalence between FAMA and applying APGM to the dual problem in Lemma 2.2.

Lemma 2.2. *If Assumption 2.3 is satisfied and FAMA and APGM are initialized with the same dual and primal starting point, then applying the FAMA in Algorithm 4 to Problem 2.2 is equivalent to applying APGM in Algorithm 2 to the dual problem defined in Problem 2.3.*

The proof of Lemma 2.2 follows the same flow as the proof of Lemma 2.1 by replacing (2.15) with

$$\lambda^k = \text{prox}_{\tau\psi}(\hat{\lambda}^k - \tau \cdot \nabla \phi(\hat{\lambda}^k)) \ . \tag{2.16}$$

Computational complexity upper-bounds for the dual sequences $\{\lambda^k\}$ generated by AMA and FAMA

Based on the equivalence shown in Lemma 2.1 and 2.2, we can now present upper-bounds on the difference of the dual function value of the sequence $\{\lambda^k\}$ generated by Algorithms 3 and 4 in Theorems 2.4 and 2.5.

Theorem 2.4. *Let $\{\lambda^k\}$ be generated by AMA in Algorithm 3. If Assumption 2.3 is satisfied, then for any $k \geq 1$ we have*

$$\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^k) \leq \frac{\rho(A)\|\lambda^0 - \lambda^\star\|^2}{2\sigma_f k} \ , \tag{2.17}$$

where $\mathbf{D}(\cdot)$ is the dual function in Problem 2.3, and λ^0 and $\lambda^\star$ denote the starting point and the optimizer, respectively.

Proof: Lemma 2.1 shows that applying AMA to Problem 2.2 is equivalent to applying PGM to Problem 2.3. The functions ϕ and ψ in Problem 2.3 are both convex, since the conjugate functions and linear functions as well as their weighted sum are always convex (conjugate function is the point-wise supremum of a set of affine functions). By Assumption 2.3, we know that f is strongly convex with modulus σ_f. By the property of the conjugate function, a Lipschitz constant of $\nabla f^\star$ is given by

$$L(\nabla f^\star) = \sigma_f^{-1} \ .$$

This provides a Lipschitz constant of $\nabla\phi$,

$$L(\nabla\phi(\lambda)) = \sigma_f^{-1} \cdot \rho(A) \ .$$

By Proposition 2.1, it follows that the sequence $\{\lambda_k\}$ generated by AMA satisfies the complexity bound (2.17). ■

Theorem 2.5. *Let $\{\lambda^k\}$ be generated by FAMA in Algorithm 4. If Assumption 2.3 is satisfied, then for any $k \geq 1$ we have*

$$\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^k) \leq \frac{2\rho(A)\|\lambda^0 - \lambda^\star\|^2}{\sigma_f(k+1)^2} \ , \tag{2.18}$$

where $\mathbf{D}(\cdot)$ is the dual function in Problem 2.3, and λ^0 and $\lambda^\star$ denote the starting point and the optimizer, respectively.

The proof of Theorem 2.5 follows the same flow of the proof of Theorem 2.4 by replace Proposition 2.1 by Proposition 2.3.

KKT conditions for Problem 2.2

In this section, we state the KKT conditions of Problem 2.2, which will be used in Section 4.4 for estimating the complexity bound in (2.18). The KKT conditions of Problem 2.2 are given by:

(i) $Av^\star + Bw^\star = c$,

(ii) $0 \in \partial f(v^\star) + A^T\lambda^\star$,

(iii) $0 \in \partial g(w^\star) + B^T\lambda^\star$.

Condition (i) represents the feasibility condition, and (ii) and (iii) represent optimality conditions. In Chapter 4, we will apply FAMA to MPC problems with polytopic and second-order cone constraints and propose two splitting strategies. In both splitting strategies, the second objective g is considered to be a set of indicator functions on convex constraints given by non-empty, closed and convex cones. In order to compute the explicit form of Condition (iii) for this case, we introduce the subdifferential of the indicator function on a convex cone, which is defined as

$$\partial I_{\mathbb{C}}(x) = \begin{cases} \mathbb{N}_{\mathbb{C}}(x) & \text{if } x \in \mathbb{C} \\ \emptyset & \text{if } x \notin \mathbb{C} \ , \end{cases} \tag{2.19}$$

where $\mathbb{N}_{\mathbb{C}}(x)$ denotes the normal cone of $\mathbb{C}$ at x. The following proposition states that being in the normal cone $\mathbb{N}_{\mathbb{C}}(x)$ is equivalent to two conditions: x is in the polar cone of $\mathbb{C}$ and the complementary slackness condition holds. This results will be used in Section 4.4 to express the explicit KKT conditions for MPC problems.

Proposition 2.5 ([30], Proposition 2.51)**.** *Let $\mathbb{C}$ be a non-empty closed convex cone. Then, for any point $\bar{x} \in \mathbb{C}$, $x \in \mathbb{N}_{\mathbb{C}}(\bar{x})$ is equivalent to $x^T\bar{x} = 0$ and $x \in \mathbb{C}^\circ$, where $\mathbb{C}^\circ$ denotes the polar cone of $\mathbb{C}$.*

2.2.4 Alternating direction method of multipliers and its accelerated variant

In Chapter 4, we will compare the proposed algorithm with another popular splitting method, the alternating direction method of multipliers (ADMM) as well as its accelerated variant. Hence, in this section, we will briefly introduce ADMM and its accelerated variant and provide a comparison between FAMA and ADMM as well as its accelerated variant FADMM from a theoretical perspective, highlighting the key differences.

Alternating Direction Method of Multipliers (ADMM)

In this section, we present another popular splitting method, called the Alternating Direction Method of Multipliers (ADMM), which was introduced first by Glowinski and Marocco in [31] and an extensive study is provided in [12]. ADMM addresses optimization problems of the form given in Problem 2.2 and requires Assumption 2.4 for convergence. The ADMM algorithm is presented in Algorithm 5. The detailed proof of convergence can be found in [12].

Assumption 2.4. *We assume that both functions f and g in Problem 2.2 are closed and convex functions.*

Algorithm 5 Alternating direction method of multipliers (ADMM)

Require: Initialize $\lambda^0 \in \mathbb{R}^{N_b}$, and $\tau > 0$

for $k = 1, 2, \cdots$ **do**

1: $v_{k+1} = \operatorname{argmin}_v \quad f(v) + \langle \lambda_k, -Av \rangle + \frac{\tau}{2}\|b - Av - Bz_k\|^2$

2: $z_{k+1} = \operatorname{argmin}_z \quad g(z) + \langle \lambda_k, -Bz \rangle + \frac{\tau}{2}\|b - Av_{k+1} - Bz\|^2$

3: $\lambda_{k+1} = \lambda_k + \tau(b - Av_{k+1} - Bz_{k+1})$

end for

Remark 2.9. *However, with Assumption 2.4, it is not clear how to derive complexity upper-bound for the iterates generated by ADMM. Requiring a stronger assumption that both function f and g are strongly convex with modulus σ_f and σ_g, a complexity bound on the iterates λ_k generated by ADMM exists and can be verified as:*

$$D(\lambda^\star) - D(\lambda_k) \leq \frac{\tau\|\lambda^\star - \lambda_0\|}{2(k-1)} \tag{2.20}$$

where τ is the stepsize, which satisfies $\tau^3 \leq \frac{\sigma_f \sigma_g^2}{\rho(A^TA)\rho(B^TB)^2}$.

The convergence properties of ADMM for different assumptions are summarized in [11].

Fast Alternating Direction Method of Multipliers (FADMM)

The fast ADMM algorithm is an accelerated variant of ADMM with a predictor-corrector type acceleration step, which is given in Algorithm 6. Comparing with ADMM, FADMM has one more restarting step, which can improve the convergence performance of the algorithm or can improve the convergence rate in the case that Assumption 2.3 is satisfied. The predictor-corrector type acceleration step is as follows: If $E_k < 1$, which is chosen from the options defined in (2.21) - (2.22), FADMM updates the parameter α and applies the standard acceleration step. The first choice for the function E_k is shown in (2.21).

$$E_k^p = \max(\|s_{k-1}\|, \|r_{k-1}\|) - \max(\|s_k\|, \|r_k\|), \tag{2.21}$$

with $r_k = b - Av_k - Bz_k$ and $s_k = \tau A^T B(z_k - z_{k-1})$. Choosing $E_k = E_k^p$ is equivalent to enforcing monotonicity on the dominant residual. The acceleration step is activated when the largest residual has decreased. This restarting rule results in faster convergence in practice, but a better convergence rate $O(\frac{1}{k^2})$ will be achieved only when both functions f and g are strongly convex. The second restart rule E_k^h in (2.22) requires Assumption 2.3, i.e., the first objective function f is strongly convex.

$$E_k^h = \|\lambda_k - \hat{\lambda}_k\| - \frac{\rho(A^T A\tau^3)}{\sigma_f}\|B\nu_k - B\hat{\nu}_k\|^2. \tag{2.22}$$

In return, it not only improves performance of the algorithm practically, but also guarantees a better convergence rate $O(\frac{1}{k^2})$.

Algorithm 6 Fast alternating direction method of multiplier (FADMM)

Require: Initialize $\alpha_0 = 1$, $\hat{w}_0 = w_{-1} \in \mathbb{R}^{N_w}$, $\hat{\lambda}_0 = \lambda_{-1} \in \mathbb{R}^{N_b}$, and $\tau > 0$

for $k = 1, 2, \cdots$ **do**

1: $v_k = \text{argmin}_v \quad f(v) + \left\langle \hat{\lambda}_k, -Av \right\rangle + \frac{\tau}{2}\|b - Av - B\hat{w}_k\|^2$

2: $w_k = \text{argmin}_w \quad g(w) + \left\langle \hat{\lambda}_k, -Bw \right\rangle + \frac{\tau}{2}\|b - Av_k - Bw\|^2$

3: $\lambda_k = \hat{\lambda}_k + \tau(b - Av_k - Bw_k)$

4: if $E_k > 0$ then

5: $\quad \alpha_{k+1} = (1 + \sqrt{4\alpha_k^2 + 1})/2$

6: $\quad \hat{\lambda}_{k+1} = \lambda_k + (\alpha_k - 1)(\lambda_k - \lambda_{k-1})/\alpha_{k+1}$

7: $\quad \hat{w}_{k+1} = w_k + (\alpha_k - 1)(w_k - w_{k-1})/\alpha_{k+1}$

8: else

9: $\quad \alpha_{k+1} = 1$, $\hat{w}_{k+1} = w_k$ and $\hat{\lambda}_{k+1} = \lambda_k$

end for

2.2.5 Comparison among AMA, FAMA, ADMM and FADMM

In this section, we provide a brief comparison between FAMA and the popular technique ADMM as well as its accelerated variant FADMM from a theoretical perspective, highlighting the key differences. Firstly, they require different assumptions

Methods	Assumptions	Convergence rate	Complexity bound	Condition on step-size
AMA	one obj strongly convex	$O(1/k)$	yes	yes
FAMA	one obj strongly convex	$O(1/k^2)$	yes	yes
ADMM	both obj convex	$O(1/k)$	no	no
FADMM	both obj strongly convex	$O(1/k^2)$	no	no

Table 2.1 – Summary of assumptions and properties of ADMM, FADMM and FAMA.

and have different convergence rates. Note that in this thesis, we denote the bound on the worst-case convergence rate as the convergence rate for short.

ADMM and FADMM require the objectives to be convex functions, and under this assumption both algorithms guarantee the same theoretical convergence rate $O(\frac{1}{k})$ in dual function value. If both objectives are strongly convex, ADMM and FADMM provide a linear convergence rate. However, this assumption is rarely satisfied by our target applications, i.e., MPC problems, since MPC problems have constraints, which can be considered as indicator functions and are therefore not strongly convex. FAMA requires one objective to be strongly convex and the other to be convex, which is stronger than the basic assumption of ADMM and FADMM, but in return, it achieves a faster convergence rate $O(\frac{1}{k^2})$ in the dual function value.

The second difference is that FAMA provides a complexity upper-bound on the number of iterations for a given accuracy (Section 2.2.3) in the dual function value, which allows for a real-time solution guarantee, i.e., a certificate that the problem can be solved in the given fixed amount of time. For ADMM and FADMM, it is not clear how to derive such a complexity bound.

The third difference is that for ADMM and FADMM, the question of how to best tune the step-size in general still remains largely unclear. Theoretically, any positive step-size guarantees convergence, however, not any positive step-size practically results in good performance. This issue has been studied for some special cases, e.g., QP problems in [32], however not suitable for problems with more general conic constraints, e.g., the second-order cone constraints. FAMA, in contrast, has a clear step-size rule, i.e., it requires the step-size to be smaller than the reciprocal of the Lipschitz constant of the gradient of the dual objectives. This condition simplifies the selection of the step-size, and together with the complexity bound allows for preconditioning the problem to speed up the algorithm, which will be discussed in Section 4.5.

The differences among AMA, FAMA, ADMM and FADMM are summarized in Table 2.1.

2.3 Inexact splitting methods

2.3.1 Inexact proximal gradient method and its accelerated variant

Notations

In this section, we use $\tilde{\cdot}$ to denote an inexact solution of an optimization problem. The proximity operator, which is defined in (2.3) with an extra subscript ϵ, i.e., $\tilde{\mu} = \text{prox}_{f,\epsilon}(v)$, means that a maximum computation error ϵ is allowed in the proximal objective function:

$$f(\tilde{\mu}) + \frac{1}{2}\|\tilde{\mu} - v\|^2 \;\leq\; \epsilon + \min_w \left\{ f(w) + \frac{1}{2}\|w - v\|^2 \right\} \tag{2.23}$$

Inexact Proximal-Gradient Method (inexact PGM)

In this section, we will introduce the inexact proximal-gradient method (inexact PGM) proposed in [22]. It addresses optimization problems of the form given in Problem 2.1 and requires Assumption 2.1 for convergence, and Assumption 2.2 for linear convergence. The algorithm is presented in Algorithm 7.

Algorithm 7 Inexact Proximal-Gradient Method

Require: Require $\tilde{w}^0 \in \mathbb{R}^{n_x}$ and $\tau < \frac{1}{L(\nabla\phi)}$
for $k = 1, 2, \cdots$ **do**
1: $\tilde{w}^k = \text{prox}_{\tau\psi,\epsilon^k}(\tilde{w}^{k-1} - \tau(\nabla\phi(\tilde{w}^{k-1}) + e^k))$
end for

Inexact PGM in Algorithm 7 allows two kinds of errors: $\{e^k\}$ represents the error in the gradient calculations of ϕ, and $\{\epsilon^k\}$ represents the error in the computation of the proximal minimization in (2.23) at every iteration k. The following propositions state the convergence property of inexact PGM for different assumptions.

Proposition 2.6 (Proposition 1 in [22])**.** *Let $\{\tilde{w}^k\}$ be generated by inexact PGM defined in Algorithm 7. If Assumption 2.1 holds, then for any $k \geq 1$ we have:*

$$\Phi(\frac{1}{k}\sum_{p=1}^{k}\tilde{w}^p) - \Phi(w^\star) \leq \frac{L(\nabla\phi)}{2k}\left(\|\tilde{w}^0 - w^\star\| + 2\Gamma^k + \sqrt{2\Lambda^k}\right)^2$$

where $\Phi(\cdot)$ is defined in Problem 2.1,

$$\Gamma^k = \sum_{p=1}^{k}\left(\frac{\|e^p\|}{L(\nabla\phi)} + \sqrt{\frac{2\epsilon^p}{L(\nabla\phi)}}\right),\; \Lambda^k = \sum_{p=1}^{k}\frac{\epsilon^p}{L(\nabla\phi)},$$

and $\tilde{w}^0$ and $w^\star$ denote the initial point of Algorithm 7 and the optimal solution of Problem 2.1, respectively.

As discussed in [22], the complexity upper-bound in Proposition 2.6 permits to derive sufficient conditions on the error sequences $\{e^k\}$ and $\{\epsilon^k\}$ for the convergence of the algorithm to the optimal solution $w^\star$:

- The series $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ are finite summable, i.e., $\sum_{k=1}^{\infty}\|e^k\| < \infty$ and $\sum_{k=0}^{\infty}\sqrt{\epsilon^k} < \infty$.
- The sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at the rate $O(\frac{1}{k^{1+\kappa}})$ for $\kappa \geq 0$.

Proposition 2.7 (Proposition 3 in [22])**.** *Let $\{\tilde{w}^k\}$ be generated by inexact PGM defined in Algorithm 7. If Assumption 2.2 holds, then for any $k \geq 0$ we have:*

$$\|\tilde{w}^k - w^\star\| \leq (1-\gamma)^k \cdot (\|w^0 - w^\star\| + \Gamma^k) \ , \tag{2.24}$$

where $\gamma = \frac{\sigma_\phi}{L(\nabla\phi)}$ and w^0 and $w^\star$ denote the initial point of Algorithm 7 and the optimal solution of Problem 2.1, respectively, and

$$\Gamma^k = \sum_{p=1}^{k}(1-\gamma)^{-p} \cdot \left(\frac{1}{L(\nabla\phi)}\|e^p\| + \sqrt{\frac{2}{L(\nabla\phi)}}\sqrt{\epsilon^p}\right) \ .$$

As discussed in [22], the upper-bound in Proposition 2.6 permits to derive sufficient conditions on the error sequences $\{e^k\}$ and $\{\epsilon^k\}$ for convergence of the algorithm to the optimal solution $w^\star$, where $\mu = 1-\gamma$:

- If the sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at a linear rate with the constant $\rho < \mu$, then $\|\tilde{w}^k - w^\star\|$ converges at a linear rate with the constant μ.
- If the sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at a linear rate with the constant $\mu < \rho < 1$, then $\|\tilde{w}^k - w^\star\|$ converges at the same rate with the constant ρ.
- If the sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at a linear rate with the constant $\rho = \mu$, then $\|\tilde{w}^k - w^\star\|$ converges at a rate of $O(k \cdot \mu^k)$.

Inexact Accelerated Proximal-Gradient Method (inexact APGM)

In this section, we introduce an accelerated variant of inexact PGM, named the inexact accelerated proximal-gradient method (inexact APGM) proposed in [22]. It addresses the same problem class in Problem 2.1 and similarly requires Assumption 2.1 for convergence, and Assumption 2.2 for linear convergence.

Algorithm 8 Inexact Accelerated Proximal-Gradient Method

Require: Initialize $v^1 = \tilde{w}^0 \in \mathbb{R}^{n_w}$ and $\tau < \frac{1}{L(\nabla\phi)}$
for $k = 1, 2, \cdots$ **do**
1: $\tilde{w}^k = \text{prox}_{\tau\psi,\epsilon^k}(v^{k-1} - \tau(\nabla\phi(v^{k-1}) + e^k))$
2: $v^k = \tilde{w}^k + \frac{k-1}{k+2}(\tilde{w}^k - \tilde{w}^{k-1})$
end for

Differing from inexact PGM, inexact APGM involves one extra linear update in Algorithm 2. If Assumption 2.1 holds, it improves the convergence rate of the complexity upper-bound from $O(\frac{1}{k})$ to $O(\frac{1}{k^2})$. If Assumption 2.2 holds, it changes the constant of the linear convergence rate from $(1-\gamma)$ to $\sqrt{1-\sqrt{\gamma}}$. The following two propositions state the convergence properties of inexact APGM.

Proposition 2.8 (Proposition 2 in [22]). *Let $\{\tilde{w}^k\}$ be generated by inexact APGM defined in Algorithm 8. If Assumption 2.1 holds, then for any $k \geq 1$ we have:*

$$\Phi(\tilde{w}^k) - \Phi(w^\star) \leq \frac{2L(\nabla\phi)}{(k+1)^2}\left(\|\tilde{w}^0 - ws^\star\| + 2\Gamma^k + \sqrt{2\Lambda^k}\right)^2$$

where $\Phi(\cdot)$ is defined in Problem 2.1

$$\Gamma^k = \sum_{p=1}^{k} p\left(\frac{\|e^p\|}{L(\nabla\phi)} + \sqrt{\frac{2\epsilon^p}{L(\nabla\phi)}}\right),\ \Lambda^k = \sum_{p=1}^{k}\frac{p^2\epsilon^p}{L(\nabla\phi)},$$

and $\tilde{w}^0$ and $w^\star$ denote the starting point of Algorithm 8 and the optimal solution of Problem 2.1, respectively.

The complexity upper-bound in Proposition 2.8 provides similar sufficient conditions on the error sequences $\{e^k\}$ and $\{\epsilon^k\}$ for the convergence of Algorithm 8:

- The series $\{k\|e^k\|\}$ and $\{k\sqrt{\epsilon^k}\}$ are finitely summable.
- The sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at the rate $O(\frac{1}{k^{2+\kappa}})$ for $\kappa \geq 0$.
- If the sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ have a decrease rate of $O(\frac{1}{k^2})$, then $\{k\|e^k\|\}$ and $\{k\sqrt{\epsilon^k}\}$ decrease at the rate $O(\frac{1}{k})$ and Γ^k increase at $O(\log k)$. Since $\sqrt{\Lambda^k}$ is always smaller than Γ^k, then we can conclude that the convergence rate is $O(\frac{\log^2 k}{k^2})$, which is poor but still converges.

If one of the above conditions holds, then the algorithm converges to the optimal solution. For $\kappa = 0$, i.e., the errors decrease at the rate $O(\frac{1}{k^2})$, the algorithm converges, but the convergence rate is poor $O(\frac{\log^2 k}{k^2})$.

Proposition 2.9. *Let $\{\tilde{w}^k\}$ be generated by inexact APGM defined in Algorithm 8. If Assumption 2.1 holds, then for any $k \geq 0$ we have:*

$$\|\tilde{w}^k - w^\star\| \leq (1-\sqrt{\gamma})^{\frac{k+1}{2}} \cdot \left(\frac{2\sqrt{\Phi(\tilde{w}^0) - \Phi(w^\star)}}{\sqrt{\sigma_\phi}} + \Theta^k\right)\ , \tag{2.25}$$

where $\gamma = \frac{\sigma_\phi}{L(\nabla\phi)}$, $\tilde{w}^0$ and $w^\star$ denote the initial point of Algorithm 7 and the optimal solution of Problem 2.1, respectively, and

$$\Theta^k = \frac{2}{\sigma_\phi}\cdot\sum_{p=0}^{k}(1-\sqrt{\gamma})^{\frac{-p-1}{2}}\cdot\left(\|e^p\| + (\sqrt{2L(\nabla\phi)} + \sqrt{\frac{\sigma_\phi}{2}})\cdot\sqrt{\epsilon^p}\right)\ .$$

Proof: From Remark 2.6, we know that the function Φ is strongly convex with the convexity modulus σ_ϕ. Thus we have

$$\frac{\sigma_\phi}{2}\|\tilde{w}^{k+1} - w^\star\|^2 \leq \Phi(\tilde{w}^{k+1}) - \Phi(w^\star)\ .$$

From Proposition 4 in [22], it follows that

$$\begin{aligned}\|\tilde{w}^{k+1} - w^\star\|^2 \leq & \frac{2}{\sigma_\phi}(1-\sqrt{\gamma})^{k+1}\Bigg(\sqrt{2(\Phi(\tilde{w}^0) - \Phi(w^\star))} \\ & + \sqrt{\frac{2}{\sigma_\Phi}}\sum_{p=0}^{k}(\|e^p\| + \sqrt{2L(\nabla\phi)\epsilon^p})(1-\sqrt{\gamma})^{-\frac{p+1}{2}} \\ & + \sqrt{\sum_{p=0}^{k}\epsilon^p(1-\sqrt{\gamma})^{-p-1}}\Bigg)^2 .\end{aligned}$$

By the fact that $\sqrt{v+\mu} \leq \sqrt{v} + \sqrt{\mu}$ for any $v, \mu \in \mathbb{R}_+$, we simplify the inequality above as

$$\begin{aligned}\|\tilde{w}^{k+1} - w^\star\|^2 \leq & \frac{2}{\sigma_\phi}(1-\sqrt{\gamma})^{k+1}\Bigg(\sqrt{2(\Phi(\tilde{w}^0) - \Phi(w^\star))} \\ & + \sqrt{\frac{2}{\sigma_\phi}}\sum_{p=0}^{k}(\|e^p\| + (\sqrt{2L(\nabla\phi)} \\ & + \sqrt{\frac{\sigma_\phi}{2}})\sqrt{\epsilon^p})(1-\sqrt{\gamma})^{-\frac{p+1}{2}}\Bigg)^2 .\end{aligned}$$

Taking the square-root of both sides of the inequality above, we get inequality (2.25). ■

Proposition 2.9 is an extension of the results in Proposition 4 in [22], presenting a complexity upper-bound on the sequence of the function value $\{\Phi(w^k) - \Phi(w^\star)\}$, where the sequence $\{w^k\}$ is generated by inexact APGM. From Remark 2.6, we know that the function Φ is a strongly convex function with the convexity modulus σ_ϕ. By using this fact, we extend the result in Proposition 4 in [22] and states a complexity upper-bound on $\|w^{k+1} - w^\star\|$.

The upper-bound in Proposition 2.9 provides similar sufficient conditions on the error sequences $\{e^k\}$ and $\{\epsilon^k\}$ for the convergence of Algorithm 8, which are obtained by replacing $\mu = 1 - \gamma$ in the sufficient conditions for Algorithm 7 in Section 2.3.1 with $\mu = \sqrt{1-\sqrt{\gamma}}$.

3 Model Predictive Control

3.1 Linear Model predictive control (MPC)

Model predictive control (MPC) has its roots in the chemical process industry, where it has been studied as a subject of dynamic matrix control since the late 1970s. Due to the ability to handle constraints, MPC has been successfully applied in practice. For a thorough description of MPC, the reader is referred to [33] and [34]. In this section, we will introduce the MPC problems considered in this thesis.

Problem 3.1.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & \sum_{t=0}^{N-1} l(x(t),u(t)) + l^f(x(N)) \\ s.t. \quad & x(t+1) = Ax(t) + Bu(t), \quad t = 0,1,\cdots,N-1 \\ & x(t) \in \mathbb{X}, \quad t = 1,2,\cdots,N-1 \\ & u(t) \in \mathbb{U}, \quad t = 0,1,\cdots,N-1 \\ & x(N) \in \mathbb{X}^f, \\ & x(0) = \bar{x} \end{aligned}$$

where $\mathbf{x} = [x^T(0),\cdots,x^T(N)]^T$ and $\mathbf{u} = [u^T(0),\cdots,u^T(N-1)]^T$ denote the state and input sequences, t denotes the discrete time index and N is the horizon of the MPC controller. The equality constraints $x(t+1) = Ax(t)+Bu(t)$ denote the linear dynamics of the discrete-time system, where A and B denote the dynamic and input matrices dynamic matrices, respectively. The variables $x(t) \in \mathbb{R}^{x_n}$ and $u(t) \in \mathbb{R}^{u_n}$ are the state and the control input of the dynamical system. l and l^f are the stage and terminal cost functions. The sets $\mathbb{X}$, $\mathbb{U}$ and $\mathbb{X}^f$ are the state, input and terminal state constraint sets, respectively. $\bar{x}$ denotes the measurement of the current state.

Assumption 3.1. *In this thesis, we assume that the MPC problem satisfies the following two properties:*

- *The stage and terminal cost functions l and l^f are convex functions.*

- *The state, input and terminal state constraint* $\mathbb{X}$, $\mathbb{U}$ *and* $\mathbb{X}^f$ *are convex sets.*

One widely used example for the stage and terminal cost functions l and l^f is quadratic cost function:

$$l(x,u) \triangleq x^T P x + u^T R u, \quad l^f(x) \triangleq x^T P^f x \ , \tag{3.1}$$

where P, P^f and R are symmetric positive definite matrices.

3.2 Distributed model predictive control

In this section, we will introduce the distributed MPC problems considered in this thesis. We consider a network of M sub-systems (nodes). The sub-systems communicate according to a fixed undirected graph $G = (\mathcal{V}, \mathcal{E})$. The vertex set $\mathcal{V} = \{1, 2, \cdots, M\}$ represents the sub-systems and the edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ specifies pairs of sub-systems that can communicate. If $(i,j) \in \mathcal{E}$, we say that sub-systems i and j are neighbours, and we denote by $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}\}$ the set of the neighbours of sub-system i. Note that $\mathcal{N}_i$ includes i. The degree of the graph G is defined as the maximum number of connections of each node, i.e., $d := \max_{1 \le i \le M} |\mathcal{N}_i|$, where $|\mathcal{N}_i|$ denotes the number of elements in the set $\mathcal{N}_i$. The distributed MPC problem, as e.g., considered in [14], is given in Problem 3.2. The reader is referred to [35], [18] and [36] for more information about distributed MPC.

Problem 3.2.

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{u}} \quad & \sum_{i=1}^{M} \sum_{t=0}^{N-1} l_i(x_i(t), u_i(t)) + \sum_{i=1}^{M} l_i^f(x_i(N)) \\
s.t. \quad & x_i(t+1) = \sum_{j \in \mathcal{N}_i} A_{ij} x_j(t) + B_{ij} u_j(t) \\
& x_i(t) \in \mathbb{X}_i, \quad u_i(t) \in \mathbb{U}_i, \\
& x_i(N) \in \mathbb{X}_i^f, \quad x_i(0) = \bar{x}_i, \quad i = 1, 2, \cdots, M.
\end{aligned}$$

where $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ are local stage and terminal cost functions and N is the horizon for the MPC problem. The state and input sequences along the horizon of agent i are denoted by $x_i = [x_i^T(0), \cdots, x_i^T(N)]^T$ and $u_i = [u_i^T(0), \cdots, u_i^T(N-1)]^T$, and the state and input sequences are denoted by $\mathbf{x} = [x_1^T, \cdots, x_M^T]^T$ and $\mathbf{u} = [u_1^T, \cdots, u_M^T]^T$. We denote the concatenations of the state sequences and input sequences of agent i and its neighbours by $x_{\mathcal{N}_i}$ and $u_{\mathcal{N}_i}$. The dynamics of the ith agent are given by the discrete time linear dynamics $x_i(t+1) = \sum_{j \in \mathcal{N}_i} A_{ij} x_j(t) + B_{ij} u_j(t)$, $i = 1, 2, \cdots, M$, where A_{ij} and B_{ij} are the dynamical matrices. The states and inputs of agent i are subject to local convex constraints $x_i(t) \in \mathbb{X}_i \quad u_i(t) \in \mathbb{U}_i$, $i = 1, 2, \cdots, M$. $\bar{x}_i$ denotes the measurement of the local current state.

Assumption 3.2. *In this thesis, we assume that the distributed MPC problem satisfies the following two properties:*

- *The local stage and terminal cost functions* l_i *and* l_i^f *are convex functions.*
- *The state, input and terminal state constraint* $\mathbb{X}_i$, $\mathbb{U}_i$ *and* $\mathbb{X}_i^f$ *are convex sets.*

Similarly to Problem 3.1, we can set the local stage cost and terminal cost functions $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ to be quadratic cost functions:

$$l_i(x_i,u_i) \triangleq x_i^T P_i x_i + u_i^T R_i u_i, \quad l_i^f(x_i) \triangleq x_i^T P_i^f x_i \ , \tag{3.2}$$

where P_i, P_i^f and R_i are symmetric positive definite matrices.

Complexity Certification of the Fast Alternating Minimization Algorithm for Linear Model Predictive Control

4

The majority of the text and content in Chapter 4 has appeared in [37] and [38].

4.1 Introduction

Fast numerical solvers for model predictive control (MPC) have attracted significant research attention due to the increasing interest in applying MPC to problems with fast dynamics and the rapidly developing computational power of embedded systems.

Various on-line optimization methods for solving MPC problems have been proposed aiming at improving the computation time or at approximating the optimum with a sub-optimal but stabilizing solution. The fast gradient method introduced in [4] has been employed to solve MPC problems with box constraints on inputs in [3]. In [5] and [6], efficient implementations of interior-point methods have been studied. Accelerated gradient methods with dual decomposition are investigated in [7] and in [8] in the context of distributed MPC. In [9] and [10], efficient active set methods for MPC have been discussed.

In this chapter, we focus on first-order methods, see e.g. [4], [28] and [11], because they offer simple iteration schemes that only require information of the function value and the gradient, and have shown good performance for solving medium and large-scale problems with moderate accuracy requirements. An efficient first-order method has the following two properties: 1. Each sub-problem, i.e. the computation of the gradient at each iteration, can be solved efficiently; 2. The optimization problem is well-posed, i.e. well conditioned, as the conditioning (geometry) of the optimization problem has a strong impact on the convergence speed of the algorithm. In this chapter, we investigate a sub-group of first-order methods, called splitting methods, and apply them to MPC problems. The efficiency of splitting methods results from splitting a complex convex minimization problem into simple sub-problems and solving them in an alternating manner. We will show how the two desired properties discussed above can be achieved by using the splitting methods.

A variety of different spitting methods exist, requiring different assumptions on the problem setup, while exhibiting different properties, see e.g. [11] and [13] for an overview. In practice, splitting methods have shown good performance for

solving complex problems in many fields, e.g. signal processing, image processing and machine learning, see e.g. [11], [12] and [13]. We focus on using these methods to solve control problems. The alternating direction method of multipliers (ADMM), as one of the most well-known splitting methods, was shown to solve optimal control problems both rapidly and robustly in [39]. However, ADMM also has its drawbacks. Firstly, ADMM only provides the convergence rate $O(\frac{1}{k})$ under the assumption that both objectives are convex, see [40]. An accelerated variant of ADMM, the fast alternating direction method of multipliers (FADMM) presented in [11], improves the practical convergence speed, but maintains the same theoretical convergence rate as ADMM. Secondly, ADMM and FADMM do not have efficient step-size tuning rules, whereas the step-size has been observed to be critical for their performance. In practice, the step-size is usually tuned by trial and error. The third aspect is that for ADMM and FADMM no theoretical complexity bound on the number of iterations is known. The complexity bound plays an important role in the context of real-time MPC, since it allows one to derive a certificate on the number of iterations to achieve a given accuracy, and thereby offers a prediction of the worst-case sub-optimality of the solution after running the algorithm for a fixed number of iterations.

In this chapter, we propose the use of the fast (accelerated) alternating minimization algorithm (FAMA) in Algorithm 4 in Section 2.2.2, which was introduced in [29] and [11], for solving MPC problems, offering superior theoretical properties while providing similar or even better performance than the existent work, i.e., applying ADMM to MPC problems in [39]. Compared to ADMM, FAMA requires stronger assumptions on the objectives of the optimization problem for convergence, which can, however, be satisfied by standard MPC problem formulations with polytopic and second-order cone constraints. In return, FAMA offers a faster convergence rate of $O(\frac{1}{k^2})$ and provides theoretical complexity bounds on the required number of iterations. **The main contributions of this chapter are**:

- Second-order cone constraints: Compared to previous work, e.g. [39] and [41], we focus on MPC problems with polytopic and second-order cone constraints, covering a broad range of MPC problems, e.g. including ellipsoidal constraints and chance constraints.

- Splitting strategies: We propose two splitting strategies for MPC problems that satisfy the assumptions of FAMA and provide efficient implementation, by reducing each iteration of FAMA to simple operations.

- Complexity bound: In order to allow for a derivation of real-time guarantees on the online solution, we derive complexity upper-bounds on the number of iterations to achieve a certain solution accuracy both in the dual function value and for the primal iterates for both splitting strategies.

- Computation of the complexity bound: We further addresses the computation of the convexify bounds, requiring the solution of a non-convex minimization problem. We propose two methods to convexify the problem and compute approximate bounds.

- Preconditioning: For MPC problems with polytopic and ellipsoidal constraints, we propose an off-line preconditioning method to further improve the conver-

gence speed of FAMA. The method reduces the complexity bounds and enlarges the step-size of the algorithm by scaling the polytopic constraints and reshaping the ellipsoidal constraints.

All properties above are demonstrated for the simulation example of a quadroter.

4.2 Fast alternating minimization algorithm (FAMA) for MPC

In the following, we show how FAMA can be applied to MPC problems to achieve an efficient online implementation. We consider the MPC problem in Problem 3.1 with state and input constraints in the form of polytopic and/or second-order cone constraints and quadratic stage and terminal costs and present two splitting strategies.

Splitting Strategy 1

By eliminating all state variables and moving the constraints to the cost in the form of indicator functions, MPC problems of this class can be reformulated in the following form suitable for the application of FAMA, with one strongly convex and one convex objective.

Problem 4.1.

$$\begin{aligned} \min_{\mathbf{u},\sigma} \quad & \underbrace{\mathbf{u}^T H \mathbf{u} + h^T \mathbf{u}}_{f(\mathbf{u})} + \underbrace{\sum_{i=1}^{M} I_{\mathbb{C}_i}(\sigma_i)}_{g(\sigma)} \\ s.t. \quad & C_i \mathbf{u} - c_i = \sigma_i, \quad i = 1, \cdots, M \ , \end{aligned}$$

where $\mathbf{u} = [u_0^T, u_1^T, \cdots, u_{N-1}^T]^T \in \mathbb{R}^{N \cdot n_u}$ denotes the sequence of inputs over the control horizon N and $\sigma = [\sigma_1^T, \cdots, \sigma_M^T]^T \in \mathbb{R}^{N_\sigma}$ are auxiliary variables. C_i and c_i denote a constant matrix and a constant vector for the ith constraint. $\mathbb{C}_i$ denotes the constraint on the variable σ_i. The matrices in the quadratic cost are given by $H = \mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R}$ and $h = \mathcal{A}^T \mathcal{Q} \mathcal{B} \bar{x}$, where

$$\mathcal{Q} = \begin{bmatrix} I_N \otimes P & 0 \\ 0 & P^f \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix},$$

$\mathcal{A} = \begin{bmatrix} A^T & \cdots & A^{N^T} \end{bmatrix}$ and $\mathcal{R} = I_N \otimes R$. $\bar{x}$ is the initial state measurement in the MPC problem. I_N denotes the identity matrix in $\mathbb{R}^{N \times N}$, and $\otimes$ denotes the Kronecker product. We assume that all state and input constraints in the MPC problem in Problem 3.1 are polytopic or second-order cone constraints. Combining the fact that the state at each time-step t in Problem 3.1 can be expressed as an affine function of the control sequence $\mathbf{u}$ and the initial state measurement $\bar{x}$, i.e.,

$x_t = A^t\bar{x} + \mathcal{B}_t\mathcal{M}_t\mathbf{u}$, with $\mathcal{B}_t = \begin{bmatrix} A^{t-1}B & A^{t-2}B & \cdots & B \end{bmatrix}$ and $\mathcal{M}_t = \begin{bmatrix} I_t \otimes I_{n_u} & 0 \end{bmatrix} \in \mathbb{R}^{tn_u \times (N-1)n_u}$, all state and input constraints in the MPC problem in Problem 3.1 can be represented as $C_i\mathbf{u} - c_i \in \mathbb{C}_i$, $i = 1, \cdots, M$. The number M denotes the total number of polytopic and second-order cone constraints in the state, input and terminal state constraints in Problem 3.1. $\mathbb{C}_i$ is given either by the non-negative orthant, i.e., $\mathbb{C}_i := \{v \mid v \geq 0\}$, or simple second-order cone constraints, i.e., $\mathbb{C}_i := \{[v_1, v_2] \mid \|v_1\| \leq v_2\}$. Note that these definitions cover all polytopic and second-order cone constraints on $\mathbf{u}$ by involving the affine coupling $C_i\mathbf{u} - c_i = \sigma_i$. Both the non-negative orthant and the second-order cone are self-dual cones, a fact that will be used in the proof of Theorem 4.3.

Assumption 4.1. *A positive definite quadratic cost on the input sequence is chosen in the MPC problem and the linear dynamical system is controllable.*

Remark 4.1. *If Assumption 4.1 holds, the first objective function $f(\mathbf{u})$ is strongly convex and the convexity modulus σ_f is given by the minimum eigenvalue of the matrix H, i.e., $\sigma_f = \lambda_{min}(H)$. Since the second objective $g(\sigma)$ is an indicator function of a convex cone, which is a convex function, then Problem 4.1 satisfies Assumption 2.3 required by FAMA.*

Remark 4.2. *We denote the current measured state by $\bar{x}$. The matrix H is independent of $\bar{x}$ and the vector h is a linear function of $\bar{x}$.*

Algorithm 9 Fast alternating minimization algorithm (FAMA) for Problem 4.1

Require: Initialize $\alpha^0 = 1$, $\alpha^1 = (1+\sqrt{5})/2$, $\lambda_i^0 = \hat{\lambda}_i^1 \in \mathbb{R}^{N_\sigma}$, $\mathbf{u}^0 = \frac{1}{2}H^{-1}(\sum_{i=1}^M C_i^T\lambda_i^0 - h)$ and $\tau < \sigma_f/\rho(C) = \lambda_{min}(H)/\rho(C)$.

for $k = 1, 2, \cdots$ **do**

1: $\mathbf{u}^k = \operatorname{argmin}_{\mathbf{u}} \quad \mathbf{u}^T H\mathbf{u} + h^T\mathbf{u} - \sum_{i=1}^M \lambda_i^{k-1^T} C_i\mathbf{u}$

2: $\hat{\mathbf{u}}^k = \mathbf{u}^k(\alpha^{k-1} + \alpha^k - 1)/\alpha^k - \mathbf{u}^{k-1}(\alpha^{k-1} - 1)/\alpha^k$

3: $\alpha^{k+1} = (1 + \sqrt{4\alpha^{k^2} + 1})/2$

for $i = 1, \cdots, M$ **do**

4: $\sigma_i^k = \mathbf{Pr}_{\mathbb{C}_i}(C_i\hat{\mathbf{u}}^k - \frac{1}{\tau}\hat{\lambda}_i^k - c_i)$

5: $\lambda_i^k = \hat{\lambda}_i^k + \tau(c_i - C_i\hat{\mathbf{u}}^k + \sigma_i^k)$

6: $\hat{\lambda}_i^{k+1} = \lambda_i^k + (\alpha^k - 1)(\lambda_i^k - \lambda_i^{k-1})/\alpha^{k+1}$

end for

end for

We apply FAMA to the MPC Problem 4.1 resulting in Algorithm 9, where $C := [C_1^T, \cdots, C_M^T]^T$. The advantage of the splitting strategy in Problem 4.1 is that the two objectives $f(\mathbf{u})$ and $g(\sigma)$ are very easy to minimize separately. The solution to the unconstrained minimization problem in Step 1 can be obtained analytically, i.e., $\mathbf{u}^k = \frac{1}{2}H^{-1}(\sum_{i=1}^M C_i^T\lambda_i^{k-1} - h)$, where the inverse H^{-1}, or an appropriate factorization, can be computed off-line. Step 4 involves basic projections onto the non-negative orthant and simplified second-order cone. We denote the projection operator as $\mathbf{Pr}_{\mathbb{C}}(\cdot)$. For the non-negative orthant, the projection is defined as

$$\mathbf{Pr}_{\mathbb{C}}(v) = \max\{0, v\} \ . \tag{4.1}$$

For the second-order cone, the projection is defined as

$$\mathbf{Pr}_{\mathbb{C}}([v_1, v_2]) = \begin{cases} [v_1, v_2] & \text{if } \|v_1\| \leq v_2 \\ \frac{v_2+\|v_1\|}{2\|v_1\|}[v_1, \|v_1\|] & \text{if } \|v_1\| > v_2,\ v_1 \neq 0 \\ [0, 0] & \text{if } \|v_1\| \leq -v_2\ . \end{cases} \tag{4.2}$$

The projections in (4.1) and (4.2) are computationally cheap. They reduce to simply a clipping and a scaling operation.

Remark 4.3. *Step 1 and 2 in Algorithm 9 are equivalent to* $\hat{\mathbf{u}}^k = argmin \quad \mathbf{u}^T H \mathbf{u} + h^T \mathbf{u} - \sum_{i=1}^{M} \hat{\lambda}_i^{k-1^T} C_i \mathbf{u}$*, which is the standard first step of FAMA. By splitting this step into two steps, we can represent* $\mathbf{u}^k$ *as a function of* λ^k*, i.e.* $\mathbf{u}^k = \frac{1}{2} H^{-1}(C^T \lambda^k - h)$*. This allows us to derive the primal complexity bound on* $\mathbf{u}^k$ *based on the dual complexity bound on the Lagrange multipliers in Section 4.3.*

Splitting Strategy 2

Consider again the MPC problems in Problem 3.1 with state and input constraints in the form of polytopic and/or second-order cone constraints and quadratic stage and terminal costs. We propose the second splitting strategy in Problem 4.2, which maintains both the states and inputs as optimization variables and involves the dynamics of the system as a constraint on the first objective f.

Problem 4.2.

$$\begin{aligned} \min_{\mathbf{z}} \quad & \underbrace{\{\mathbf{z}^T Q \mathbf{z} + q^T \mathbf{z} \mid T\mathbf{z} = t\}}_{f(\mathbf{z})} + \underbrace{\sum_{i=1}^{M} I_{\mathbb{C}_i}(\sigma_i)}_{g(\sigma)} \\ s.t. \quad & D_i \mathbf{z} - d_i = \sigma_i, \quad i = 1, \cdots, M\ , \end{aligned}$$

where $\mathbf{z} = [x_0^T, x_1^T, \cdots, x_N^T, u_0^T, u_1^T, \cdots, u_{N-1}^T]$ denotes the state and input sequences over the control horizon, and $\sigma = [\sigma_1^T, \cdots, \sigma_M^T]^T \in \mathbb{R}^{N_\sigma}$ are auxiliary variables. The matrices in the quadratic cost are given by

$$Q = \begin{bmatrix} I_{N+1} \otimes P & 0 & 0 \\ 0 & P^f & 0 \\ 0 & 0 & I_N \otimes R \end{bmatrix}, \quad q = 0.$$

The matrices T and t represent the dynamical constraint, with

$$T = \begin{bmatrix} -I_{n_x} & 0 & \cdots & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ A & -I_{n_x} & 0 & \cdots & 0 & B & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots & 0 & B & \ddots & \vdots \\ \vdots & \ddots & A & -I_{n_x} & 0 & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & A & -I_{n_x} & 0 & \cdots & 0 & B \end{bmatrix}$$

and $t = [\bar{x}, 0, \cdots, 0] \in \mathbb{R}^{(N+1)n_x}$, where $\bar{x}$ is the initial state measurement in the MPC problem. All state and input constraints in the MPC Problem 3.1 can be

represented as $D_i\mathbf{z} - d_i \in \mathbb{C}_i$, $i = 1, \cdots, M$, where D_i and d_i denote a constant matrix and a constant vector for the ith constraint. The number M denotes the total number of polytopic and second-order cone constraints in the state, input and terminal state constraints in Problem 3.1. $\mathbb{C}_i$ are given either by the non-negative orthant, i.e., $\mathbb{C}_i := \{v \mid v \geq 0\}$, or simple second-order cone constraints, i.e., $\mathbb{C}_i := \{[v_1, v_2] | \, \|v_1\| \leq v_2\}$.

Assumption 4.2. *The cost on the states and the inputs in the MPC problem are chosen to be positive definite quadratic functions.*

Assumption 4.2 is a relatively strong assumption on an MPC problem, but possible to be satisfied by setting the weight matrices in the stage cost functions to be positive definite.

Remark 4.4. *The first objective $f(\mathbf{z})$ in Problem 4.2 consists of a quadratic function and a convex constraint. If Assumption 4.2 is satisfied, the matrix Q is positive definite. The convex constraint can be considered as an indicator function, which is convex. Due to the fact that the sum of a strongly convex and a convex function is strongly convex, the objective $f(\mathbf{z})$ is strongly convex and Problem 4.2 satisfies Assumption 2.3 with the convexity modulus $\sigma_f = \lambda_{min}(Q)$.*

We apply FAMA to MPC Problem 4.2 resulting in Algorithm 10, where $D := [D_1^T, \cdots, D_M^T]^T$.

Algorithm 10 Fast alternating minimization algorithm (FAMA) for Problem 4.2

Require: Initialize $\alpha^0 = 1$, $\alpha^1 = (1 + \sqrt{5})/2$, $\lambda_i^0 = \hat{\lambda}_i^1 \in \mathbb{R}^{N_\sigma}$, $\mathbf{z}^0 = \operatorname{argmin}_{T\mathbf{z}=t} \ \mathbf{z}^T Q\mathbf{z} + q^T\mathbf{z} - \sum_{i=1}^M \lambda_i^{0^T} D_i\mathbf{z}$ and $\tau < \sigma_f/\rho(D) = \lambda_{min}(Q)/\rho(D)$.
for $k = 1, 2, \cdots$ **do**
1: $\mathbf{z}^k = \operatorname{argmin}_{T\mathbf{z}=t} \ \mathbf{z}^T Q\mathbf{z} + q^T\mathbf{z} - \sum_{i=1}^M \lambda_i^{k-1^T} D_i\mathbf{z}$
2: $\hat{\mathbf{z}}^k = \mathbf{z}^k(\alpha^{k-1} + \alpha^k - 1)/\alpha^k - \mathbf{z}^{k-1}(\alpha^{k-1} - 1)/\alpha^k$
3: $\alpha^{k+1} = (1 + \sqrt{4\alpha^{k^2} + 1})/2$
for $i = 1, \cdots, M$ **do**
4: $\sigma_i^k = \mathbf{Pr}_{\mathbb{C}_i}(D_i\hat{\mathbf{z}}^k - \frac{1}{\tau}\hat{\lambda}_i^k - d_i)$
5: $\lambda_i^k = \hat{\lambda}_i^k + \tau(d_i - D_i\hat{\mathbf{z}}^k + \sigma_i^k)$
6: $\hat{\lambda}_i^{k+1} = \lambda_i^k + (\alpha^k - 1)(\lambda_i^k - \lambda_i^{k-1})/\alpha^{k+1}$
end for
end for

Remark 4.5. *The projection and dual update loop, i.e., Steps 4-6 in Algorithm 9 and Algorithm 10 can be computed in parallel.*

Remark 4.6. *Similar to Algorithm 9, Steps 2 and 3 in Algorithm 10 are equivalent to the first step of FAMA in Algorithm 4, i.e., $\hat{\mathbf{z}}^k = \operatorname{argmin}_{T\mathbf{z}=t} \ \mathbf{z}^T Q\mathbf{z} + q^T\mathbf{z} - \sum_{i=1}^M \hat{\lambda}_i^{k-1^T} D_i\mathbf{z}$. By splitting this step into two steps, we can represent $\mathbf{z}^k$ as a function of λ^k, which permits to derive the complexity bound on $\|\mathbf{z}^k - \mathbf{z}^\star\|$ in Section 4.3.*

Remark 4.7. *In Problems 4.1 and 4.2, we presented two splitting strategies for solving MPC problems. These two splitting strategies have different advantages and disadvantages. For a given MPC problem, Problem 4.1 has less optimization variables than Problems 4.2 and no equality constraints in the first objective, which reduces the size and the complexity of the optimization problem, and makes the problem easier to solve, in principle. However, due to the fact that the matrix in the quadratic function, as well as the matrices in the conic constraints, in Problem 4.1 are dense matrices compared to Problems 4.2, the real computation task for each iteration for Problem 4.1 is heavier than Problems 4.2.*

4.3 Complexity Bounds of FAMA for MPC

Complexity upper-bounds of optimization algorithms are important for real-time MPC, since they provide a certificate that a solution of pre-specified sub-optimality can be obtained within the available computation time. In this section, we will derive the complexity upper-bounds on the number of iterations to achieve a certain solution accuracy for the sequences generated by Algorithm 9 and Algorithm 10.

4.3.1 Complexity upper-bounds for both the primal and dual sequences $\{\mathbf{u}^k\}$ and $\{\lambda^k\}$ generated by Algorithm 9

Before we present the complexity upper-bounds on the primal and dual sequences in Theorem 4.3, we claim two new lemmas showing properties of convex cones, which will be used in the proof of Theorem 4.3.

Lemma 4.1. *Let $\mathbb{C}$ be a convex cone. The conjugate function of the indicator function of the set $\mathbb{S} := \{v | -v \in \mathbb{C}\}$ is equal to the indicator function of the dual cone of $\mathbb{C}$, i.e., $I^\star_{\mathbb{S}}(v) = I_{\mathbb{C}^\star}(v)$.*

Proof: By the definition of a convex cone, the set $\mathbb{S}$ is still a convex cone. Example 7.3.5 in [24] shows $I^\star_{\mathbb{S}}(v) = I_{\mathbb{S}^\circ}(v)$, where $\mathbb{S}^\circ$ denotes the polar cone of $\mathbb{S}$. By the definitions of the polar cone and the dual cone, we know $\mathbb{S}^\circ = \{w \mid w'v \leq 0,\ -v \in \mathbb{C}\} = \{w \mid w'v \geq 0,\ v \in \mathbb{C}\} = \mathbb{C}^\star$, and the result $I^\star_{\mathbb{S}}(v) = I_{\mathbb{C}^\star}(v)$ follows. ■

Lemma 4.2. *Let $\mathbb{C}$ be the non-negative orthant $\mathbb{C} := \{v \mid v \geq 0\}$ or a second order cone $\mathbb{C} := \{[v_1, v_2] | \ \|v_1\| \leq v_2\}$. For any $v \in \mathbb{R}^{N_\mathbb{C}}$, the point $z = \mathbf{Pr}_\mathbb{C}(v) - v$ satisfies $z \in \mathbb{C}$.*

Proof: For the non-negative orthant, it is easy to show that $z = \mathbf{Pr}_\mathbb{C}(v) - v = \max\{0, v\} - v \geq 0$. For a second-order cone, we denote $z = [z_1, z_2]$ and show that $\|z_1\| \leq z_2$ holds for the three cases in equation (4.2). For the first case, it can easily be verified that $\|z_1\| \leq z_2$. For the second case, we have

$$\|z_1\| = \left\| \frac{v_2 + \|v_1\|}{2\|v_1\|} v_1 - v_1 \right\| = \left\| \frac{v_2 - \|v_1\|}{2} \right\| \ ,$$

$$z_2 = \frac{v_2 + \|v_1\|}{2\|v_1\|} \|v_1\| - v_2 = \frac{\|v_1\| - v_2}{2} \ .$$

Since in this case it holds that $\|v_1\| > v_2$, we get $\|z_1\| \leq z_2$. For the third case, we have $\|z_1\| = \|v_1\|$ and $z_2 = -v_2$. Since $\|v_1\| \leq -v_2$, we prove $\|z_1\| \leq z_2$. ■

We are ready to present the complexity upper-bounds on the primal and dual sequences in Theorem 4.3.

Theorem 4.3. *Consider Problem 4.1. Let $\{\mathbf{u}^k\}$ and $\{\lambda^k\}$ be generated by Algorithm 9, where $\lambda^k = [\lambda_1^{k^T}, \cdots, \lambda_M^{k^T}]^T$ and λ_i are the Lagrange multipliers associated with the constraint $C_i\mathbf{u} - c_i = \sigma_i$ at iteration k. If Assumption 4.1 is satisfied, then for any $k \geq 1$*

$$\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^k) \leq \frac{2\rho(C)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{min}(H)(k+1)^2} \ , \tag{4.3}$$

where $\lambda^0 = [\lambda_1^{0^T}, \cdots, \lambda_M^{0^T}]^T$ and $\lambda^\star$ denote the starting point and the optimizer, respectively. If $\lambda_i^0 \in \mathbb{C}_i$ for all $i = 1, \cdots, M$, then $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 1$ and $i = 1, \cdots, M$ and

$$\|\mathbf{u}^k - \mathbf{u}^\star\|^2 \leq \frac{4\rho(C)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{min}^2(H)k^2} \ . \tag{4.4}$$

Proof: The first objective in Problem 4.1 is equal to $f(\mathbf{u}) = \mathbf{u}^T H\mathbf{u} + h^T\mathbf{u}$, and therefore $\sigma_f = \lambda_{min}(H)$. Since Assumption 4.1 holds, Problem 4.1 satisfies Assumption 2.3, and the complexity upper-bound in (2.18) holds by Theorem 2.5. This directly implies the dual bound in (4.3) by considering the matrix A in Problem 2.2 to be the matrix $C = [C_1^T, \cdots, C_M^T]^T$ in Problem 4.1.

The second step is to prove that if $\lambda_i^0 \in \mathbb{C}_i$ for all $i = 1, \cdots, M$, then $\lambda_i^k \in \mathbb{C}_i$ for all $k \in \mathbb{N}$ and $i = 1, \ldots, M$. From Steps 4 and 5 in Algorithm 9, we know that $\lambda_i^k = \hat{\lambda}_i^k + \tau(c_i - C_i\mathbf{u}^k + \sigma_i^k) = \tau(\mathbf{Pr}_{\mathbb{C}_i}(C_i\mathbf{u}^k - \frac{1}{\tau}\hat{\lambda}_i^k - c_i) - (C_i\mathbf{u}^k - \frac{1}{\tau}\hat{\lambda}_i^k - c_i))$. By the fact that $\tau > 0$ and Lemma 4.2, we can conclude $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 1$ and $i = 1, \cdots, M$. The last step is to prove inequality (4.4). From Step 1 in Algorithm 9 we have

$$\mathbf{u}^k = \frac{1}{2}H^{-1}(C^T\lambda^{k-1} - h) \ ,$$

which implies

$$\begin{aligned}
\|\mathbf{u}^k - \mathbf{u}^\star\|^2 &= \|\frac{1}{2}H^{-1}C^T(\lambda^{k-1} - \lambda^\star)\|^2 \\
&\leq \frac{1}{2\lambda_{min}(H)}(\lambda^{k-1} - \lambda^\star)^T CH^{-1}C^T(\lambda^{k-1} - \lambda^\star) \\
&= \frac{2}{\lambda_{min}(H)}\Big[\frac{1}{4}\lambda^{k-1^T}CH^{-1}C^T\lambda^{k-1} - (\frac{1}{2}h^TH^{-1}C^T + d^T)\lambda_{k-1} \\
&\quad - \frac{1}{4}\lambda^{\star^T}CH^{-1}C^T\lambda^\star + (\frac{1}{2}h^TH^{-1}C^T + d^T)\lambda^\star \\
&\quad + \frac{1}{2}((\lambda^\star)^TCH^{-1}C^T\lambda^\star - \lambda^{k^T}CH^{-1}C^T\lambda^\star) \\
&\quad + (\frac{1}{2}h^TH^{-1}C^T + d^T)(\lambda^\star - \lambda^{k-1})\Big] \ .
\end{aligned}$$

The dual function of Problem 4.1 is

$$\begin{aligned}\mathbf{D}(\lambda) &= -f^\star(C^T\lambda) + d^T\lambda - g^\star(-\lambda) \\ &= -\frac{1}{4}\lambda^T C H^{-1} C^T \lambda + \frac{1}{2} h^T H^{-1} C^T \lambda - \frac{1}{4} h^T H^{-1} h + d^T \lambda - \sum_{i=1}^{M} I^\star_{-\lambda_i \in \mathbb{C}_i} \ .\end{aligned}$$

Since we know that all $\mathbb{C}_i$ are self-dual cones, i.e. $\mathbb{C}_i^\star = \mathbb{C}_i$, Lemma 4.1 implies that the dual function can be simplified as

$$\mathbf{D}(\lambda) = -\frac{1}{4}\lambda^T C H^{-1} C^T \lambda + (\frac{1}{2} h^T H^{-1} C^T + d^T)\lambda - \frac{1}{4} h^T H^{-1} h - \sum_{i=1}^{M} I_{\lambda_i \in \mathbb{C}_i} \ ,$$

and the gradient of the dual function for $\lambda_i \in \mathbb{C}_i$ is

$$\nabla \mathbf{D}(\lambda) = -\frac{1}{2} C (H^{-1})^T C^T \lambda + (\frac{1}{2} C (H^{-1})^T h + d) \ . \tag{4.5}$$

Since we have shown that $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 0$ and $i = 1, \cdots, M$, we obtain

$$\|\mathbf{u}^k - \mathbf{u}^\star\|^2 \leq \frac{2}{\lambda_{min}(H)} (\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^{k-1}) - \nabla \mathbf{D}^T(\lambda^\star)(\lambda^\star - \lambda^{k-1})) \ .$$

Since the dual function $\mathbf{D}$ is concave, by optimality, we conclude

$$\begin{aligned}\frac{2}{\lambda_{min}(H)} (\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^{k-1}) - \nabla \mathbf{D}(\lambda^\star)(\lambda^{k-1} - \lambda^\star)) &\leq \frac{2}{\lambda_{min}(H)} (\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^{k-1})) \\ &\leq \frac{4\rho(C)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{min}^2(H) k^2} \ .\end{aligned}$$

■

Remark 4.8. *The proof of inequality (4.4) exists for the case of polytopic constraints in the context of distributed MPC problems [8]. We extend it to the case of general self-dual conic constraints in the context of MPC problems.*

4.3.2 Complexity upper-bounds for both the primal and dual sequences $\{\mathbf{z}^k\}$ and $\{\lambda^k\}$ generated by Algorithm 10

Theorem 4.4. *If Assumption 4.2 holds, the sequence $\{\lambda_k\}$ generated by applying FAMA to Problem 4.2 satisfies*

$$\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^k) \leq \frac{2\rho(D)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{min}(H)(k+1)^2} \ , \tag{4.6}$$

where λ^0 and $\lambda^\star$ denote the starting point and the optimizer, respectively. If $\lambda_i^0 \in \mathbb{C}_i$ for all $i = 1, \cdots, M$, then $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 1$ and $i = 1, \cdots, M$ and

$$\|\mathbf{z}^k - \mathbf{z}^\star\|^2 \leq \frac{2\lambda_{max}(M_1)\rho(D)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{max}(I - M_1^T Q)\lambda_{min}(Q) k^2} \ , \tag{4.7}$$

where M_1 is defined as in (4.8), and I is an identity matrix.

Proof: Since Assumption 4.2 is satisfied, Problem 4.2 satisfies Assumption 2.3. It follows from Theorem 2.5 that the sequence $\{\lambda^k\}$ generated by Algorithm 10 satisfies the complexity bound in (4.6). From Steps 4 and 5 in Algorithm 10, we know that $\lambda_i^k = \hat{\lambda}_i^k + \tau(d_i - D_i\mathbf{z}^k + \sigma_i^k) = \tau(\mathbf{Pr}_{\mathbb{C}_i}(D_i\mathbf{z}^k - \frac{1}{\tau}\hat{\lambda}_i^k - d_i) - (D_i\mathbf{z}^k - \frac{1}{\tau}\hat{\lambda}_i^k - d_i))$. Since $\tau > 0$ and $\lambda_i^0 \in \mathbb{C}_i$ for all $i = 1, \cdots, M$, from Lemma 4.2 we know $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 1$ and $i = 1, \cdots, M$. In the following, we prove the inequality in (4.7). From Step 1 in Algorithm 10, we know

$$\mathbf{z}^k = M_1(D^T\lambda^{k-1} - q) + M_2 t \ ,$$

where $M_1 \in \mathbb{R}^{n_z \times n_z}$, $M_2 \in \mathbb{R}^{n_z \times n_t}$ are defined as in (4.8).

$$\begin{aligned}\begin{bmatrix} M_1 & M_2 \\ M_3 & M_4 \end{bmatrix} &= \begin{bmatrix} 2Q & T^T \\ T & 0 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \frac{1}{2}(Q^{-1} - Q^{-1}T^T(TQ^{-1}T^T)^{-1}TQ^{-1}) & Q^{-1}T^T(TQ^{-1}TT)^{-1} \\ (TQ^{-1}T^T)^{-1}TQ^{-1} & -\frac{1}{2}(TQ^{-1}T^T)^{-1} \end{bmatrix}.\end{aligned} \tag{4.8}$$

$$\begin{aligned}&\|\mathbf{z}^k - \mathbf{z}^\star\|^2 \\ &= \|M_1D^T(\lambda^{k-1} - \lambda^\star)\|^2 \\ &\leq \frac{\lambda_{max}(M_1)}{\lambda_{max}(I - M_1^TQ)}(\lambda^{k-1} - \lambda^\star)^TD(I - M_1^TQ)M_1D^T(\lambda^{k-1} - \lambda^\star) \\ &= \frac{\lambda_{max}(M_1)}{\lambda_{max}(I - M_1^TQ)}\Big[- \lambda^{k-1^T}D(M_1^TQ - I)M_1D^T\lambda^{k-1} \\ &\quad - (2DM_1q - DM_2t + 2DM_1^TQM_2t - 2DM_1^TQM_1q + d)\lambda^{k-1} \\ &\quad + \lambda^{\star^T}D(M_1^TQ - I)M_1D^T\lambda^\star \\ &\quad + (2DM_1q - DM_2t + 2DM_1^TQM_2t - 2DM_1^TQM_1q + d)\lambda^\star \\ &\quad - 2\lambda^{\star^T}D(M_1^TQ - I)M_1D^T\lambda^\star + 2\lambda^{\star^T}D(M_1^TQ - I)M_1D^T\lambda^{k-1} \\ &\quad - (2DM_1q - DM_2t + 2DM_1^TQM_2t - 2DM_1^TQM_1q + d)(\lambda^\star - \lambda^{k-1})\Big] \ .\end{aligned} \tag{4.9}$$

Then we can derive an upper bound on $\|\mathbf{z}^k - \mathbf{z}^\star\|^2$ following the derivations in (4.9). The dual function of Problem 4.2 is equal to

$$\begin{aligned}\mathbf{D}(\lambda) =& - f^\star(D^T\lambda) + d^T\lambda - g^\star(-\lambda) \\ =& \lambda^TD(M_1^TQM_1 - M_1)D^T\lambda + (2DM_1q - DM_2t + 2DM_1^TQM_2t \\ &- 2DM_1^TQM_1q + d)^T\lambda + (M_1q - M_2t)^TQ(M_1q - M_2t) - q^T(M_1q - M_2t) \\ &- \sum_{i=1}^{M} I_{\lambda_i \in \mathbb{C}_i} \ .\end{aligned}$$

Note that the matrix $(I - M_1^TQ)M_1$ is positive semi-definite, which can be easily seen from the concavity of the dual function $\mathbf{D}(\lambda)$, which contains the quadratic function with weight matrix $(M_1^TQ - I)M_1$. Then, the function $-\mathbf{D}(\lambda)$ is convex, which implies that $(I - M_1^TQ)M_1$ is a positive semi-definite matrix. Since we have shown

that $\lambda_i^k \in \mathbb{C}_i$ for all $k \geq 0$ and $i = 1, \cdots, M$, the indicator functions $\sum_{i=1}^{M} I_{\lambda_i \in \mathbb{C}_i}$ in the dual function $\mathbf{D}$ can be omitted. Then from (4.9), we obtain

$$\|\mathbf{z}^k - \mathbf{z}^\star\|^2 \leq \frac{\lambda_{max}(M_1)}{\lambda_{max}(I - M_1^T Q)}(\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^{k-1}) - \nabla \mathbf{D}^T(\lambda^{k-1})(\lambda^\star - \lambda^{k-1})) \ .$$

By optimality, we conclude

$$\|\mathbf{z}^k - \mathbf{z}^\star\|^2 \leq \frac{\lambda_{max}(M_1)}{\lambda_{max}(I - M_1^T Q)}(\mathbf{D}(\lambda^\star) - \mathbf{D}(\lambda^{k-1})) \leq \frac{2\lambda_{max}(M_1)\rho(D)\|\lambda^0 - \lambda^\star\|^2}{\lambda_{max}(I - M_1^T Q)\lambda_{min}(Q)k^2} \ .$$

■

Remark 4.9. *Theorem 4.3 and 4.4 can be directly extended to an MPC problem with positive semi-definite cone constraints, since a positive semi-definite cone is also a self-dual cone.*

4.4 Computation of complexity bounds

Theorem 4.3 and Theorem 4.4 provide complexity bounds on the number of iterations for FAMA applied to Problem 4.1 and 4.2, to reach a given accuracy. To use these bounds to compute the number of iterations in a real-time MPC framework, we need to know all quantities in the complexity bounds. For a given problem, all quantities in the complexity bounds are known except for $\|\lambda^0 - \lambda^\star\|$. This section is devoted to computing or approximating the value of $\|\lambda^0 - \lambda^\star\|$. Related work includes [42] and [43], where the authors consider quadratic programming (QP) problems and estimate $\|\lambda^0 - \lambda^\star\|$ by solving an off-line mixed-integer linear programming problem, which does, however, not cover second-order cone constraints. In this section, we provide two methods for approximating $\|\lambda^0 - \lambda^\star\|$. We consider the cold-starting strategy, i.e., $\lambda^0 = 0$, in which case the problem reduces to computing the largest value of $\|\lambda^\star_{min}(\bar{x})\|$ for a given initial state set $\bar{x} \in \mathbb{X}_0$, i.e.,

$$\|\lambda^\star_{min}\| := \max_{\bar{x} \in \mathbb{X}_0} \quad |\lambda^\star_{min}(\bar{x})| \ , \tag{4.10}$$

where $\lambda^\star_{min}(\bar{x})$ denotes the minimal l_2-norm solution

$$\lambda^\star_{min}(\bar{x}) := \text{argmin}_{\lambda \in \Lambda^\star(\bar{x})} \quad |\lambda| \ , \tag{4.11}$$

and $\Lambda^\star(\bar{x})$ denotes the set of optimal Lagrange multipliers of Problem 4.1 or Problem 4.2, respectively, for a given initial state $\bar{x}$. The constraint $\lambda \in \Lambda^\star(\bar{x})$ represents the KKT conditions on the optimal Lagrange multipliers for Problem 4.1 or Problem 4.2. In the following, we focus on the computation of $\|\lambda^\star_{min}\|$ for a given $\mathbb{X}_0$ for Problem 4.1, and present two methods for its approximation. The methods can be easily extended to Problem 4.2.

4.4.1 Upper-bound on $\|\lambda^\star_{min}\|$ using sum of squares (SOS) relaxations

From the description above, we know that $\|\lambda^\star_{min}\|$ is the optimal solution of a three-level non-convex optimization problem, i.e., Problem 4.1 and the two problems in (4.10) and (4.11). In this section, we propose a method for constructing a

convex approximation and providing an upper bound of $\|\lambda^\star_{min}\|$. Note that when replacing $\|\lambda^\star_{min}\|$ by its upper bound, the complexity upper bounds in Theorem 4.3 still hold. The idea of the method is the following. We first rewrite the three-level problem as one optimization problem by involving the KKT conditions of the inner problems as constraints, i.e., we solve problem (4.10) subject to the KKT conditions of Problem 4.1 and those of problem (4.11). The KKT conditions include conic constraints originating from the primal and dual feasibility and polynomial constraints originating from the optimality and the complementary slackness conditions. Due to the fact that polynomial constraints are nonconvex, we use the sum of squares relaxations to approximate the solution of the problem, resulting in an SDP problem. See, e.g., [44] for previous work on using SOS relaxations for optimization problems with polynomial constraints.

The KKT conditions of Problem 4.1 are obtained from the KKT conditions derived for Problem 2.2 in Section 2.2.3 and Proposition 2.5. Since the constraint $\lambda \in \Lambda^\star(\bar{x})$ is equivalent to the KKT conditions of Problem 4.1, the problem (4.11) can be represented by (4.12), where $\mathbb{C}_i^\circ$ denotes the polar cone of $\mathbb{C}_i$.

$$\begin{aligned} \lambda^\star_{min}(\bar{x}) = \text{argmin}_{\lambda,\mathbf{u},\sigma_i} \quad & \|\lambda\| \\ s.t. \quad & 2H\mathbf{u} + h + \sum_{i=1}^{M} C_i^T \lambda_i = 0, \\ & C_i\mathbf{u} - d_i = \sigma_i,\ \sigma_i^T \lambda_i = 0, \\ & \sigma_i \in \mathbb{C}_i,\ \lambda_i \in \mathbb{C}_i^\circ,\ i = 1, \cdots, M. \end{aligned} \tag{4.12}$$

Note that problem (4.12) is defined for one initial state $\bar{x}$, since the vectors h and c_i are affine functions of $\bar{x}$. We derive the KKT conditions of problem (4.12), which can be easily obtained by following the standard rules in [24] and [25], and introduce these KKT conditions into problem (4.10). Due to the non-convexity of the polynomial constraints originating from the optimality and the complementary slackness conditions, we apply SOS relaxations [44] to the problem in order to compute an upper bound of $\|\lambda^\star_{min}\|$. It is important to notice that according to the results in [44], SOS relaxations always provide an upper-bound of the optimal solution $\|\bar{\lambda}^\star_{min}\| > \|\lambda^\star_{min}\|$, and there exists a sufficiently high-order SOS relaxation such that $\|\bar{\lambda}^\star_{min}\| = \|\lambda^\star_{min}\|$. However, this method is limited to small-scale problems.

4.4.2 Sample-based estimation of $\|\lambda^\star_{min}\|$

We present a sample-based approach to estimate $\|\lambda^\star_{min}\|$, which can be easily applied for medium- and large-scale problems. The optimization problem in (4.10) can be reformulated as

Problem 4.5.

$$\begin{aligned} \|\lambda^\star_{min}\| = \min \quad & \gamma \\ s.t. \quad & \|\lambda^\star_{min}(\bar{x})\| - \gamma \leq 0,\ \forall \bar{x} \in \mathbb{X}_0 \ , \end{aligned}$$

where $\lambda^\star_{min}(\bar{x})$ is defined in (4.11). We consider $\bar{x}$ as an uncertainty parameter, and apply the approach proposed in [45], called the scenario approach, in to Problem 4.5. We first introduce some required definitions.

Definition 4.1 (Definition 1 in [45]). *Let $\tilde{\gamma} \in \mathbb{R}$ be a candidate solution for Problem 4.5. The probability that a better solution exists is defined as*

$$V(\tilde{\gamma}) := \boldsymbol{P}\{\bar{x} \in \mathbb{X}_0 : \ \tilde{\gamma} \leq \|\lambda^\star(\bar{x})\|\} \ .$$

Definition 4.2 (Definition 2 in [45]). *Let $\epsilon \in [0,1]$. We say that $\tilde{\gamma} \in \mathbb{R}$ is an ϵ-level robustly feasible solution if $V(\tilde{\gamma}) \leq \epsilon$.*

We collect N_s random samples $\{\bar{x}_1, \cdots, \bar{x}_{N_s}\}$ in $\mathbb{X}_0$, and construct the sample-based optimization problem

Problem 4.6.

$$\begin{aligned} \|\lambda_{min}^{N_s}\| := \min \quad & \gamma \\ s.t. \quad & \|\lambda_{min}^\star(\bar{x}_i)\| - \gamma \leq 0, \forall\, \bar{x}_i, \quad i = 1, \cdots, N_s \ . \end{aligned}$$

The following corollary states the probabilistic meaning of the optimal solution $\|\lambda_{min}^{N_s}\|$ returned by Problem 4.6.

Corollary 4.1. *(Corollary 1 in [45]) Fix two real numbers $\epsilon \in [0,1]$ (level parameter) and $\beta \in [0,1]$ (confidence parameter) and let $N_s \geq \frac{1}{\epsilon\beta} - 1$. Then, with probability no smaller than $1-\beta$, $\|\lambda_{min}^{N_s}\|$ returned by Problem 4.6 is an optimal solution with ϵ-level robust feasibility for Problem 4.5.*

The remaining question is how to solve Problem 4.6, which is still non-convex, as it involves the KKT conditions of the problem in (4.12). In the following, we provide a method to compute an upper-bound of $\|\lambda_{min}^{N_s}\|$, by solving N_s convex problems. For a sample $\bar{x}_i$, an optimal solution $\lambda^\star(\bar{x}_i)$ can be computed by applying FAMA to Problem 4.1. Since the solution satisfies $\|\lambda^\star(\bar{x}_i)\| \geq \|\lambda_{min}^\star(\bar{x}_i)\|$, then, we can easily compute $\|\tilde{\lambda}_{min}^{N_s}\| := \max_{1 \leq i \leq N_s}\{\|\lambda^\star(\bar{x}_i)\|\}$, which satisfies $\|\tilde{\lambda}_{min}^{N_s}\| \geq \|\lambda_{min}^{N_s}\|$.

The procedure of using the scenario approach in [45] to estimate a solution for Problem 4.5 is summarized as follows: Choose ϵ and β, and take $N_s \geq \frac{1}{\epsilon\beta} - 1$. Randomly draw N_s samples $\{\bar{x}_1, \cdots, \bar{x}_{N_s}\}$ in $\mathbb{X}_0$ and run FAMA for Problem 4.1 to calculate the corresponding $\{\|\lambda^\star(\bar{x}_1)\|, \cdots, \|\lambda^\star(\bar{x}_{N_s})\|\}$. Then compute $\|\tilde{\lambda}_{min}^{N_s}\| = \max_{1 \leq i \leq N_s}\{\|\lambda^\star(\bar{x}_i)\|\}$, which is an upper-bound of an optimal solution with ϵ-level robust feasibility for Problem 4.5. Note that all computations can be done off-line, and therefore a large number of samples can be potentially considered to compute a good approximation of $\|\lambda_{min}^\star\|$ with high confidence.

Remark 4.10. *In this section, we proposed to use the SOS relaxation in [44] and the scenario approach in [45] to approximate the optimal solution of $\|\lambda^0 - \lambda^\star\|$ for a given MPC problem with a known initial state set. For small-scale problems, the SOS relaxation is suggested to use, since it always provides an upper-bound of the optimal solution $\|\bar{\lambda}_{min}^\star\| > \|\lambda_{min}^\star\|$, and there exists a sufficiently high-order SOS relaxation such that real optimal solution can be achieved. However, this approach is computationally heavy and is limited to small-scale problems. For medium- and large-scale problems, the scenario approach in [45] is suggested to use. This approach does not guarantee any strict upper-bound, but it is more applicable due its low computational requirement.*

4.5 Preconditioning

Preconditioning has been observed to offer significant compuational speedups in gradient based methods in [3] and [46]. In [3] and [46], preconditioning methods for solving linear MPC problem with polytopic constraints were presented. In this section, we present a preconditioning technique to improve the performance of FAMA, when applied to linear MPC problems with polytopic and ellipsoidal (a special case of second-order cone) constraints. The goal of the method is to enlarge the step-size and decrease the complexity bounds of the algorithms by minimizing the condition number of the constraint matrices.

We again focus on the first splitting in Problem 4.1. The preconditioning method can be easily extended to the second splitting in Problem 4.2 by replacing the matrices H, C and c in Problem 4.1 in the following design procedure by the matrices Q, D and d in Problem 4.2. Recall the condition on the step-size $\tau < \lambda_{min}(H)/\rho(C)$ and the complexity bound in Theorem 4.3. The value $\rho(C)$ affects them in the way that the smaller $\rho(C)$, the larger the step-size and the smaller the complexity bound. Therefore, we minimize the condition number of the matrix C by imposing preconditioning matrices to C to enlarge the step-size and decrease the complexity bounds. In order to simplify the notation, we assume that Problem 4.1 has only two constraints, a polytopic constraint $C_1\mathbf{u} - c_1 \geq 0$ and an ellipsoidal constraint $|C_2\mathbf{u} - c_2| \leq 1$. We introduce a positive-definite diagonal matrix P_1 and a positive-definite matrix P_2 to precondition the constraints: P_1 to scale the polytopic constraints $P_1C_1\mathbf{u} - P_1c_1 \geq 0$, and $P_2 \in \mathbb{R}^{n_{C_2} \times n_{C_2}}$ to reshape the ellipsoidal constraint $|P_2C_2\mathbf{u} - P_2c_2| \leq 1$, where n_{C_1} and n_{C_2} denote the number of rows of the matrices C_1 and C_2, respectively.

We first compute the optimal preconditioning matrices P_1 and P_2 minimizing the condition number of C^TC by means of the following optimization problem (see Chapter 3.1 in [47]). Let $W_1 = P_1^TP_1$ and $W_2 = P_2^TP_2$.

Problem 4.7.

$$\begin{aligned} \min_{\alpha, W_1, W_2} \quad & \alpha \\ s.t. \quad & \mu I \preceq \begin{bmatrix} C_1^T & C_2^T \end{bmatrix} \begin{bmatrix} W_1 & 0 \\ 0 & W_2 \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \preceq \alpha I \ , \\ & W_1 = blkdiag(w_1, \cdots, w_{n_{C_1}}) \ , \ w_i > 0 \ , i = 1, \cdots, n_{C_1} \ , \ W_2 \succ 0 \ , \end{aligned}$$

where μ is equal to the minimum eigenvalue of C^TC. It is important to point out that by setting the preconditioning matrix P_1 to be a diagonal matrix, the preconditioned polytopic constraint is equivalent to the original one. However, the preconditioning matrix P_2 changes the ellipsoidal constraint, which means that the preconditioning of the ellipsoidal constraint modifies the optimal solution. In order to guarantee that the solution given by the preconditioned problem is still a sub-optimal and feasible solution to Problem 4.1, one extra step is required to compute the maximal reshaped ellipsoidal constraint contained in the original ellipsoidal constraint. Problem 4.8 computes the maximal inner approximation.

Problem 4.8.

$$\begin{aligned} \min_{\beta,\omega} \quad & \beta \\ s.t. \quad & \begin{bmatrix} -\omega+1 & 0 & 0 \\ 0 & \omega I & \beta(P_2C_2)^{-1} \\ 0 & \beta(P_2C_2)^{-1} & (C_2^TC_2)^{-1} \end{bmatrix} \succeq 0 \ , \ \omega \geq 0 \ , \ \beta \geq 0 \ . \end{aligned}$$

The LMI constraint in Problem 4.8 guarantees that the new scaled ellipsoidal constraint $|P_2C_2\mathbf{u} - P_2c_2| \leq \beta$ is contained in the original constraint $|C_2\mathbf{u} - c_2| \leq 1$ (see Chapter 8.4.2 in [25]). Note that the matrix P_2 in Problem 4.8 is not an optimization variable but a constant computed by Problem 4.7. We summarize the properties of the proposed preconditioning method in Problems 4.7 and 4.8 in the following theorem.

Theorem 4.9. *Let $\mathbf{u}^\star$ be an optimal solution of the original problem: $\min_{\mathbf{u}} \ \mathbf{u}^T H\mathbf{u} + h^T\mathbf{u}$, s.t. $C_1\mathbf{u} - c_1 \geq 0$ and $|C_2\mathbf{u} - c_2| \leq 1$, and let $\mathbf{u}_p^\star$ be an optimal solution of the preconditioned problem: $\min_{\mathbf{u}_p} \ \mathbf{u}_p^T H\mathbf{u}_p + h^T\mathbf{u}_p$, s.t $P_1C_1\mathbf{u}_p - P_1c_1 \geq 0$ and $|P_2C_2\mathbf{u}_p - P_2c_2| \leq \beta$, where the matrices P_1 and P_2 and the parameter β are computed by Problem 4.7 and Problem 4.8. The following holds:*

- *The optimal solution $\mathbf{u}_p^\star$ is a feasible solution of the original problem, i.e. $C_1\mathbf{u}_p^\star - c_1 \geq 0$ and $|C_2\mathbf{u}_p^\star - c_2| \leq 1$.*
- *The preconditioned constraint matrix $C_p = [C_1^TP_1^T, C_2^TP_2^T]^T$ satisfies $\rho(C_p) \leq \rho(C)$, where $C = [C_1^T, C_2^T]^T$.*

Proof: Since the matrix W_1 is set to be a positive definite diagonal matrix, the preconditioned polytopic constraint $P_1C_1\mathbf{u}_p - P_1c_1 \geq 0$ is equivalent to the original one $C_1\mathbf{u} - c_1 \geq 0$. Problem 4.8 guarantees that the preconditioned ellipsoidal constraint $|P_2C_2\mathbf{u}_p - P_2c_2| \leq \beta$ is an inner-approximation of the original ellipsoidal constraint. Hence, the optimal solution $\mathbf{u}_p^\star$ is a feasible solution of the original problem. By optimality of P_1 and P_2 for Problem 4.7, it follows immediately that $\rho(C_p) \leq \rho(C)$. ■

In the following, we consider the special case that the ellipsoidal constraint $|C_2\mathbf{u} - c_2| \leq 1$ originates from a terminal state constraint, which is a common problem type in MPC. In this case, the preconditioning of the ellipsoidal constraint has not only to maintain feasibility, but also stability properties. To address this problem, we first present some notation and preliminaries about stability of an MPC controller. Consider the discrete-time linear time-invariant system $x_{t+1} = A_dx_t + B_du_t$, where x_t and u_t denote the state and input at time t, and A_d and B_d denote the dynamical matrices of a discrete-time linear system. Let $\mathbb{X}$ and $\mathbb{U}$ be the state and input constraints and K be a linear control law, such that $A_d + B_dK$ is stable.

Definition 4.3. *(Positive invariant (PI) set): A set $\mathbb{P} \subseteq \mathbb{R}^n$ is a positively invariant set of system $x_{t+1} = A_dx_t + B_dKx_t$, if $A_dx_t + B_dKx_t \in \mathbb{P}$ and $Kx_t \in \mathbb{U}$ for all $x_t \in \mathbb{P}$.*

Let $|Ex_N - e| \leq 1$ be the original ellipsoidal terminal constraint on the state, where $E \succ 0$ and N is the horizon of the MPC problem. Since x_N can be represented by a linear combination of the control sequence $\mathbf{u}$ and the initial state $\bar{x}$, i.e., $x_N = S_1\mathbf{u} + S_2\bar{x}$, it follows that $C_2 = ES_1$ and $c_2 = e - ES_2\bar{x}$.

According to the standard MPC theory, the original terminal constraint $|Ex_N - e| \leq 1$ has to be a positively invariant set to guarantee stability of the closed-loop system. In order to maintain this property for the preconditioned ellipsoidal constraint, an additional invariance condition is imposed on W_2 in Problem 4.7. We exemplify this procedure for an ellipsoidal terminal constraint of the form $\mathbb{X}_f = \{x_N | x_N^T \Gamma x_N < 1\}$, where $\Gamma \succ 0$, i.e. $\Gamma = E^T E$ and $e = 0$. Invariance of the preconditioned ellipsoid can be ensured by enforcing the constraint in (4.13) in Problem 4.7, which can be written as an LMI by using Schur complements.

$$(A_d + B_d K)^T \Gamma^{\frac{1}{2}^T} W_2 \Gamma^{\frac{1}{2}} (A_d + B_d K) - \Gamma^{\frac{1}{2}^T} W_2 \Gamma^{\frac{1}{2}} \preceq 0 \tag{4.13}$$

Again, by solving Problem 4.8, we can compute the maximal inner approximation in $|Ex_t - e| \leq 1$. If the state and input constraints $\mathbb{X}$ and $\mathbb{U}$ are polytopic sets, then the inner approximation in Problem 4.8 can be relaxed by only requiring the scaled new ellipsoid to be contained in $\mathbb{X} \cap K\mathbb{U}$.

The following theorem summarizes the properties of the preconditioned MPC controller.

Theorem 4.10. *Let $\mathbf{u}^\star$ be an optimal solution of the original problem: $\min_{\mathbf{u}} \ \mathbf{u}^T H \mathbf{u} + h^T \mathbf{u}, s.t\, C_1 \mathbf{u} - c_1 \geq 0$ and $|C_2 \mathbf{u} - c_2| \leq 1$, in which the ellipsoidal constraint originates from a quadratic invariant set, and let $\mathbf{u}_p^\star$ be an optimal solution of the preconditioned problem: $\min_{\mathbf{u}_p} \ \mathbf{u}_p^T H \mathbf{u}_p + h^T \mathbf{u}_p, s.t\ P_1 C_1 \mathbf{u}_p - P_1 c_1 \geq 0$ and $|P_2 C_2 \mathbf{u}_p - P_2 c_2| \leq \beta$. The matrices P_1 and P_2 are computed by Problem 4.7 with the extra constraint in (4.13). The parameter β is computed by Problem 4.8. The following holds:*

- *The optimal solution $\mathbf{u}_p^\star$ is a feasible solution of the original problem, i.e. $C_1 \mathbf{u}_p^\star - c_1 \geq 0$ and $|C_2 \mathbf{u}_p^\star - c_2| \leq 1$.*
- *The preconditioned constraint matrix $C_p = [C_1^T P_1^T, C_2^T P_2^T]^T$ satisfies $\rho(C_p) \leq \rho(C)$, where $C = [C_1^T, C_2^T]^T$.*
- *The new ellipsoidal constraint $|P_2 C_2 \mathbf{u}_p - P_2 c_2| \leq \beta$ is a positive invariant set.*

Proof: The first two statements follow from the same proof of Theorem 4.9. The condition $(A_d + B_d K)^T \Gamma^{\frac{1}{2}^T} W_2 \Gamma^{\frac{1}{2}} (A_d + B_d K) - \Gamma^{\frac{1}{2}^T} W_2 \Gamma^{\frac{1}{2}} \preceq 0$ guarantees the new ellipsoidal constraint is a positive invariant set. ■

Remark 4.11. *Since the matrices C_1, E and S_1 are independent of the initial state $\bar{x}$, the computation of the preconditioning matrices P_1 and P_2 and the parameter β can be computed off-line.*

Remark 4.12. *The preconditioning method introduced in this section can be easily extended to the case with more than two constraints and with ellipsoidal input constraints.*

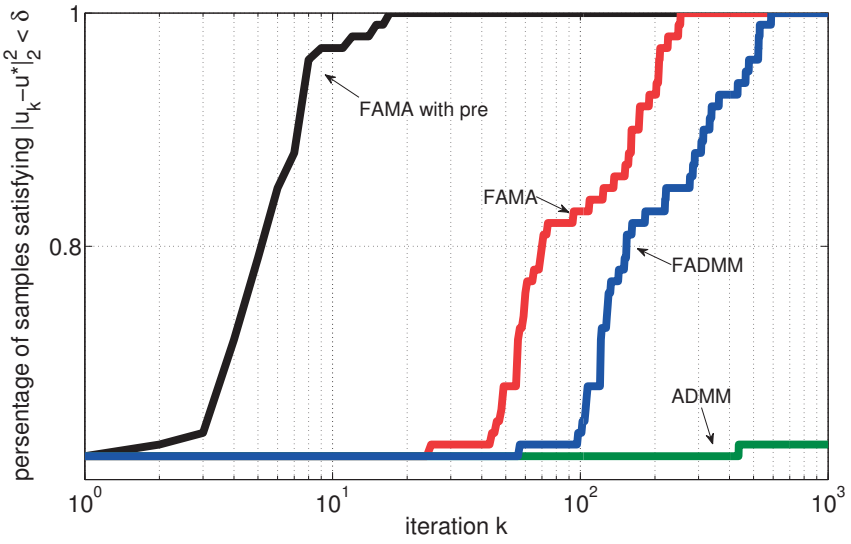
Remark 4.13. *The preconditioning is enabled by the existence of the step-size rule and the complexity bounds. Since ADMM and FADMM do not provide these properties, it is unclear how to provide a similar preconditioning method for them. The derived theoretical properties of FAMA therefore also have practical implications by allowing for tuning the algorithm and improving its performance for the particular problem at hand.*

4.6 Numerical example

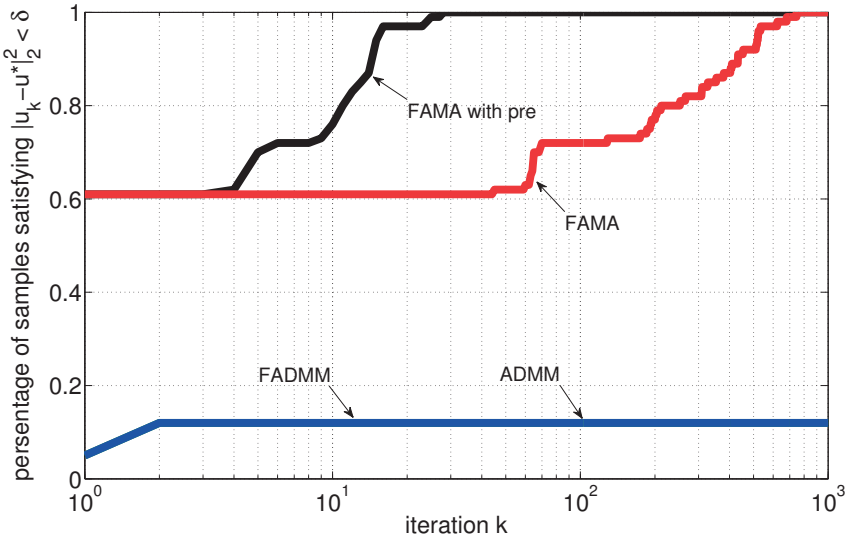
This section illustrates the theoretical findings of the chapter and demonstrates the performance of FAMA for solving MPC problems. We consider a quadroter model, see [48], which is driven by four independently controlled rotors. In this experiment, we use a cascaded control structure and design an MPC controller to control the inner-loop, which is in charge of the derivative of the height of the quadroter, the roll, pitch and yaw angles and the derivative of these angles, i.e $x = [\dot{z}, \alpha, \beta, \gamma, \dot{\alpha}, \dot{\beta}, \dot{\gamma}]^T$. The state of the system is subject to constraints on the maximum angle, maximum angle velocity as well as maximum velocity in the z direction - these constraints are mainly chosen to ensure validity of the linearized model and have been specified as: $|\dot{z}| \leq 1m/s$, $|\alpha| \leq 10°$, $|\beta| \leq 10°$, $|\dot{\alpha}| \leq 15°$, $|\dot{\beta}| \leq 15°$ and $|\dot{\gamma}| \leq 60°$. The input constraint is $0 \leq u \leq 1$. The horizon of the MPC controller is set to $N = 25$. The terminal state x_N is subject to a positively invariant ellipsoidal terminal constraint.

In the simulations shown in Fig. 4.1 and Fig. 4.2, we collect 1000 randomly sampled initial states in the set $\{\bar{x}|[-0.5m/s, -5°, -5°, -60°, -5°/s, -5°/s, -30°/s]^T \leq \bar{x} \leq [0.5m/s, 5°, 5°, 60°, 5°/s, 5°/s, 30°/s]^T\}$ and compare the proposed FAMA algorithms with ADMM and FADMM for the two splitting strategies in Problem 4.1 and Problem 4.2. For FAMA, the step-size is set to $0.99 * \lambda_{min}(H)/\rho(C)$ and $0.99 * \lambda_{min}(Q)/\rho(D)$, respectively, while for ADMM and FADMM the step-size is set to the best value obtained by manual tuning. Performance is measured by the percentage of samples, for which $\|\mathbf{u}^k - \mathbf{u}^\star\|/\|\mathbf{u}^\star\| < \delta$ or $\|\mathbf{z}^k - \mathbf{z}^\star\|/\|\mathbf{z}^\star\| < \delta$ after k iterations. For the first splitting strategy, FAMA with preconditioning shows the best performance, fastest convergence speed and good accuracy after few iterations. FAMA without preconditioning converges more slowly but still faster than ADMM and FADMM. Fig. 4.1b shows that the solution accuracy given by ADMM and FADMM is inferior to FAMA. They achieve a solution accuracy of $\delta = 10^{-6}$ in 1000 iterations for only 10% of the samples. In Fig. 4.2, it is shown that for the second splitting strategy, FAMA similarly performs better than ADMM and FADMM, fast convergence speed and good accuracy. The preconditioning is not shown for the second splitting strategy, as it will not provide significant improvements due to the fact that for this example the constraint matrix D is a block-diagonal matrix. Fig. 4.1 and Fig. 4.2 also show that for this example FAMA on the first splitting strategy in Problem 4.1 requires less iterations and provides better accuracy than FAMA on the second splitting in Problem 4.2.

Fig. 4.3 illustrates the primal sequences generated by Algorithm 9 and Algorithm 10 and the corresponding complexity bounds in Theorem 4.3 and Theorem 4.4, respectively, for the initial state $\bar{x} = [0.5m/s, 5°, 5°, 60°, 5°/s, 5°/s, 20°/s]^T$. The complexity bounds are computed by setting $\lambda^0 = 0$ and assessing $\|\lambda^*\|$ by the sample-based method in Section 4.4.2 using 3000 samples. According to Corollary 4.1, β and ϵ are set to be 1.6×10^{-2}. In Fig. 4.3a, it can be clearly seen that the preconditioning method improves the convergence speed of the algorithm and reduces the complexity upper-bound. As the number of iterations k increases, the complexity upper-bound of Algorithm 9 with preconditioning appears to be less tight, and the practical convergence is faster than the bound. The complexity bound for the second splitting in Fig. 4.3b shows similar behavior. We don't apply the preconditioning method to Problem 4.2, since the optimization problem for this example has well conditioning already and the condition number of the optimization problem is very



(a) $\delta = 10^{-4}$



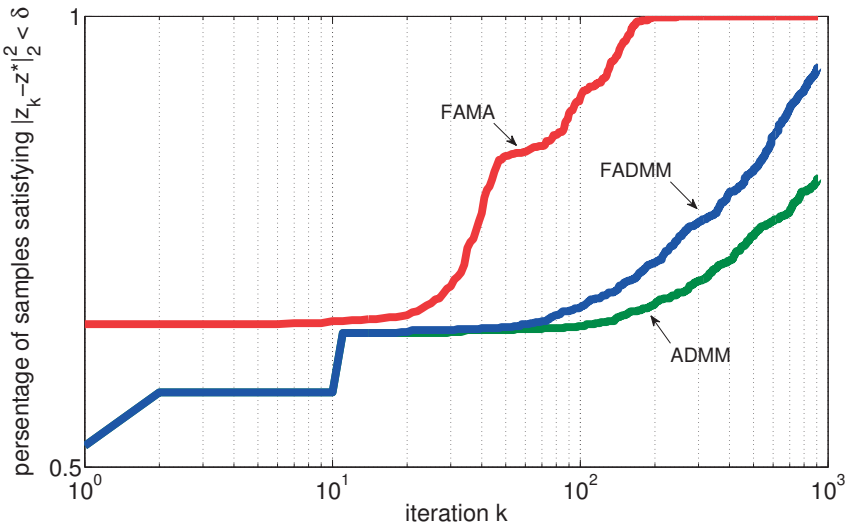
(b) $\delta = 10^{-6}$

Figure 4.1 – Performance of ADMM, FADMM, FAMA and FAMA with preconditioning applied to Problem 4.1 for the quadroter example.

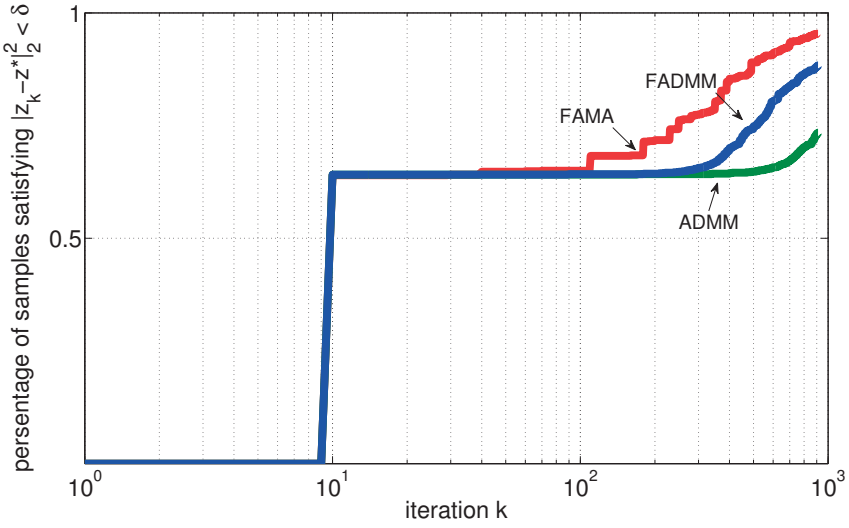
close to one.

4.7 Conclusion

In Chapter 4, we studied the fast alternating minimization algorithm and proposed efficient implementations for solving MPC problems with polytopic and second-order cone constraints. We derived complexity bounds on the number of iterations for both dual and primal variables, which are of particular relevance in the context of real-time MPC to bound the required online computation time, and further discussed the computation of the complexity bounds. For MPC problems with polyhedral and

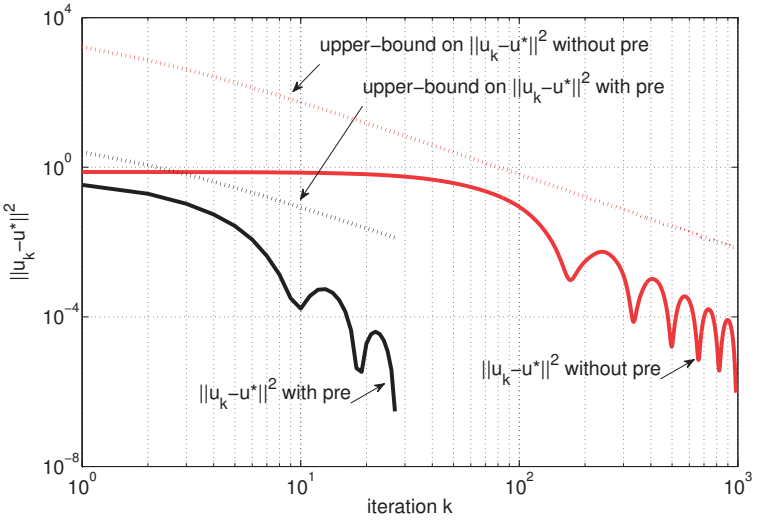


(a) $\delta = 10^{-4}$

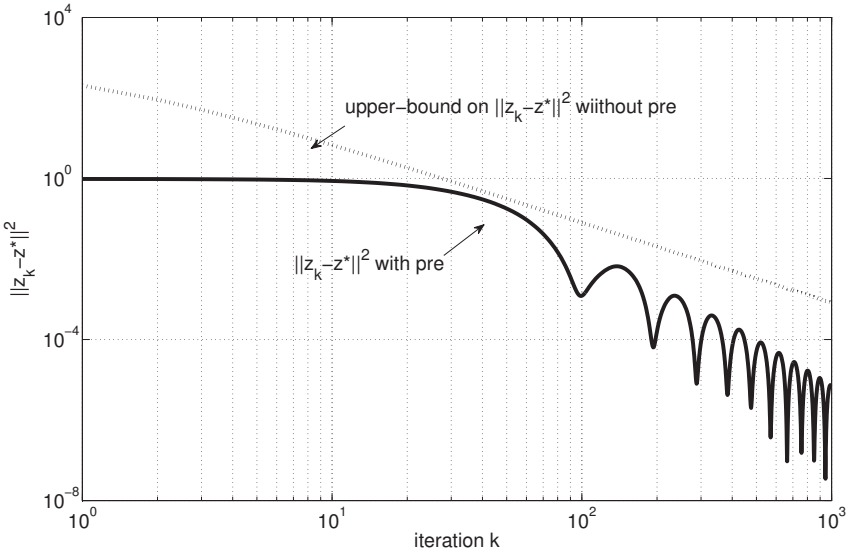


(b) $\delta = 10^{-6}$

Figure 4.2 – Performance of ADMM, FADMM and FAMA applied to Problem 4.2 for the quadroter example



(a)



(b)

Figure 4.3 – (a) Illustration of $\|\mathbf{u}^k - \mathbf{u}^\star\|^2$ by Algorithm 9 (with and without preconditioning) and the complexity upper-boun in Theorem 4.3 (with and without preconditioning), (b) Illustration of $\|\mathbf{z}^k - \mathbf{z}^\star\|^2$ generated by Algorithm 10 without preconditioning and the complexity upper-bounds in Theorem 4.4 for without preconditioning.

ellipsoidal constraints, we provided an off-line pre-conditioning method to further improve the convergence speed of FAMA by decreasing the complexity upper-bounds and enlarging the step-size of the algorithm.

Inexact Alternating Minimization Algorithm for Distributed Optimization

5

The majority of the text and content in Chapter 5 has appeared in [49] and [50].

5.1 Introduction

Splitting methods offer the ability to split the objective into multiple parts and minimize them in an alternating way, which provides an efficient technique for solving distributed optimization problems arising in many engineering fields. See in [12]. By considering the local cost functions, as well as local constraints, as the multiple objectives of a distributed optimization problem, splitting methods allow us to split a global constrained optimization problem into sub-problems according to the structure of the network, and solve them in a distributed manner. The advantages of using distributed optimization algorithms include the following three points: in contrast to centralized methods, they do not require full communication, but only local communication, i.e., neighbour-to-neighbour communication; secondly, they parallelize the computation tasks of solving the global optimization problem into small sub-problems, which reduces the required computational power for each sub-system; thirdly, distributed optimization algorithms preserve the privacy of each-subsystem in the sense that each sub-system computes an optimal solution without sharing its local cost function and local constraint with all the entities in the network.

In this chapter, we will consider a distributed Model Predictive Control problem in Section 5.3.2 as the application for the distributed optimization to demonstrate the proposed algorithms, as well as the theoretical findings. For distributed systems, implementing an centralized MPC controller becomes challenging, since solving an centralized MPC problem in a centralized way requires full communication to collect information from each sub-system, and the computational power to solve the global problem in one central entity. Distributed model predictive control [35] is a promising tool to overcome the limiting computational complexity and communication requirements associated with centralized control of large-scale networked systems. The research on distributed MPC has mainly focused on the impact of distributed optimization on system properties such as stability and feasibility, and on the development of efficient distributed optimization algorithms. See in [14], [8]

and [36].

However, a key challenge in practice is that distributed optimization algorithms, see e.g. [19], [12] and [8], may suffer from inexact local solutions and unreliable communications. The resulting inexact updates in the distributed optimization algorithms affect the convergence properties, and can even cause divergence of the algorithm. The local inexact solution can be due to many causes, for instance limited local computation power and computation time, as well as inexact information of the local cost functions and constraints. For example, in distributed MPC problems the inexactness may originate from noisy state measurements in each local systems.

In this chapter, we study inexact splitting methods and aim at answering the following questions: how these errors affect the algorithms and under which conditions convergence can still be guaranteed. Seminal work on inexact optimization algorithms includes [20], [21] and [22]. In [21], the authors propose an inexact decomposition algorithm for solving distributed optimization problems by employing smoothing techniques and an excessive gap condition. In [22], an inexact proximal-gradient method, as well as its accelerated version, are introduced. The proximal gradient method, also known as the iterative shrinkage-thresholding algorithm (ISTA) [28], has two main steps: the first one is to compute the gradient of the smooth objective and the second one is to solve the proximal minimization. The conceptual idea of the inexact proximal-gradient method is to allow errors in these two steps, i.e. the error in the calculation of the gradient and the error in the proximal minimization. The results in [22] show convergence properties of the inexact proximal-gradient method and provide conditions on the errors, under which convergence of the algorithm can be guaranteed.

Building on the results in [22], we propose two new inexact splitting algorithms, the inexact Alternating Minimization Algorithm (inexact AMA) and its accelerated variant, inexact Fast Alternating Minimization Algorithm (inexact FAMA). The contributions of this work are the following:

- We propose the inexact AMA and inexact FAMA algorithms, which are inexact variants of the splitting methods, AMA and FAMA in [29] and [11]. We show that applying inexact AMA and inexact FAMA to the primal problem is equivalent to applying the inexact proximal-gradient method (inexact PGM) and the inexact accelerated proximal-gradient method (inexact APGM) in [22] to the dual problem. Based on this fact, we extend the results in [22], and show the convergence properties of inexact AMA and inexact FAMA. We derive complexity upper bounds on the number of iterations to achieve a certain accuracy for the algorithms. By exploiting these complexity upper-bounds, we present sufficient conditions on the errors for convergence of the algorithms.

- We study the special case where the error sequences do not satisfy the sufficient conditions for convergence, but are bounded by constants. We show the complexity upper bounds on the number of iterations for this special case.

- We apply inexact AMA and inexact FAMA for solving distributed optimization problems with local computational errors. We present the complexity upper bounds of the algorithms for this special case, and show sufficient conditions on the local computational errors for convergence.

- We study the special case of quadratic local objective functions, relating to a standard form of distributed MPC problems. We show that if the local quadratic functions are positive definite, then the algorithms converge to the optimal solution with a linear rate. We propose to use gradient-based method to solve the local problems. By exploiting the sufficient condition on the local computational errors for the convergence together with a warm-starting strategy, we provide an approach to certify the number of iterations for the proximal gradient method to solve the local problems to the accuracy required for convergence of the distributed algorithm. The proposed on-line certification method only requires on-line local information.
- We demonstrate the performance and the theoretical results for inexact algorithms by solving a randomly generated example of a distributed MPC problem with 40 subsystems.

5.2 Inexact alternating minimization algorithm and its accelerated variant

The inexact proximal gradient method, as well as its accelerated version, introduced in Section 2.3, is limited to the case where both objectives are functions of the same variable. However, many optimization problems from engineering fields, e.g., optimal control and machine learning [12], are not of this problem type. In order to generalize the problem formulation, we employ the alternating minimization algorithm (AMA) and its accelerated variant in [29] and [11], which cover optimization problems of the form of Problem 5.1. In this section, we extend AMA and its accelerated variant to the inexact case and present the theoretical convergence properties.

Problem 5.1.

$$\begin{aligned} \min \quad & f(v) + g(w) \\ s.t. \quad & Av + Bw = c \end{aligned}$$

with variables $v \in \mathbb{R}^{n_v}$ and $w \in \mathbb{R}^{n_w}$, where $A \in \mathbb{R}^{n_c \times n_v}$, $B \in \mathbb{R}^{n_c \times n_w}$ and $c \in \mathbb{R}^{n_c}$. $f : \mathbb{R}^{n_v} \to \mathbb{R}$ and $g : \mathbb{R}^{n_w} \to \mathbb{R}$ are convex functions. The Lagrangian of Problem 5.1 is:

$$L(v, w, \lambda) = f(v) + g(w) - \lambda^T (Av + Bw - c) \ , \tag{5.1}$$

and the dual function is:

$$\begin{aligned} D(\lambda) = & \inf_{v,w} L(v, w, \lambda) && \text{(5.2a)} \\ = & - \sup_{v} \left\{ \lambda^T A v - f(v) \right\} \\ & - \sup_{w} \left\{ \lambda^T B w - g(w) \right\} + \lambda^T c && \text{(5.2b)} \\ = & - f^\star(A^T \lambda) - g^\star(B^T \lambda) + \lambda^T c, && \text{(5.2c)} \end{aligned}$$

where $f^\star$ and $g^\star$ are the conjugate functions of f and g. The dual problem of Problem 5.1 is:

Problem 5.2.

$$\min \quad -D(\lambda) = \underbrace{f^\star(A^T\lambda)}_{\phi(\lambda)} + \underbrace{g^\star(B^T\lambda) - c^T\lambda}_{\psi(\lambda)}.$$

5.2.1 Inexact alternating minimization algorithm (inexact AMA)

We propose the inexact alternating minimization algorithm (inexact AMA) presented in Algorithm 11 for solving Problem 5.1. The algorithm allows errors in Step 1 and Step 2, i.e., both minimization problems are solved inexactly with errors δ^k and θ^k, respectively.

Algorithm 11 Inexact alternating minimization algorithm (Inexact AMA)

Require: Initialize $\lambda^0 \in \mathbb{R}^{N_b}$, and $\tau < \sigma_f/\rho(A)$
for $k = 1, 2, \cdots$ **do**
1: $\tilde{v}^k = \operatorname{argmin}_v \{f(v) + \langle \lambda^{k-1}, -Av\rangle\} + \delta^k$.
2: $\tilde{w}^k = \operatorname{argmin}_w \{g(w) + \langle \lambda^{k-1}, -Bw\rangle + \frac{\tau}{2}\|c - A\tilde{v}^k - Bw\|^2\} + \theta^k$
3: $\lambda^k = \lambda^{k-1} + \tau(c - A\tilde{v}^k - B\tilde{w}^k)$
end for

In this section, we study the theoretical properties of inexact AMA under Assumption 5.1. If Assumption 5.1 holds, we show that inexact AMA in Algorithm 11 is equivalent to applying inexact PGM to the dual problem in Problem 5.2 with the following correspondence: the gradient computation error in Algorithm 7 is equal to $e^k = A\delta^k$ and the error of solving the proximal minimization is equal to $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$. See in Lemma 5.1. With this equivalence, the complexity bound in Proposition 2.6 can be extended for the inexact AMA algorithm in Theorem 5.3.

Assumption 5.1. *We assume that*

- *f is a strongly convex function with convexity modulus σ_f,*
- *g is a convex function, not necessarily smooth.*

Lemma 5.1. *If Assumption 5.1 is satisfied and inexact AMA and inexact PGM are initialized with the same dual and primal starting point, then applying the inexact AMA in Algorithm 11 to Problem 5.1 is equivalent to applying inexact PGM in Algorithm 7 to the dual problem defined in Problem 5.2 with the errors $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$, where $L(\psi)$ denotes the Lipschitz constant of the function ψ.*

Proof: In order to show the equivalence, we prove that Step 1, 2 and 3 in Algorithm 11 are equivalent to Step 1 in Algorithm 7, i.e. the following equality holds:

$$\lambda^k = \operatorname{prox}_{\tau\psi,\epsilon^k}(\lambda^{k-1} - \tau(\nabla\phi(\lambda^{k-1}) + e^k)) \tag{5.3}$$

with $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$. Step 2 in Algorithm 11 implies:

$$B^T\lambda^{k-1} + \tau B^T(c - A\tilde{v}^k - Bw^k) \in \partial g(w^k),$$

where $w^k = \text{argmin}_w \{g(w) + \langle \lambda^{k-1}, -Bw\rangle + \frac{\tau}{2}\|c - A\tilde{v}^{k+1} - Bw\|^2\} = \tilde{w}^k - \theta^k$. From the property of the conjugate function $p \in \partial f(q) \Leftrightarrow q \in \partial f^\star(p)$, it follows:

$$w^k \in \partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - A\tilde{v}^k - Bw^k)).$$

By left-multiplying with B and subtracting c on both sides, we obtain:

$$Bw^k - c \in B\partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - A\tilde{v}^k - Bw^k)) - c.$$

By multiplying with τ and adding $\lambda^{k-1} + \tau(c - A\tilde{v}^k - Bw^k)$ on both sides, we get:

$$\begin{aligned}\lambda^{k-1} - \tau A\tilde{v}^k \in \tau B\partial g^\star(B^T\lambda^{k-1} + \tau B^T(c - A\tilde{v}^k - Bw^k)) \\ - \tau c + \lambda^{k-1} + \tau(c - A\tilde{v}^k - Bw^k).\end{aligned}$$

Since $\psi(\lambda) = g^\star(B^T\lambda) - c^T\lambda$, we have $\partial\psi(\lambda) = B\partial g^\star(B^T\lambda) - c$, which implies:

$$\begin{aligned}\lambda^{k-1} - \tau A\tilde{v}^k \in \tau\partial\psi(\lambda^{k-1} + \tau(c - A\tilde{v}^k - Bw^k)) \\ + \lambda^{k-1} + \tau(c - A\tilde{v}^k - Bw^k).\end{aligned}$$

Since $w^k = \tilde{w}^k - \theta^k$, it follows that:

$$\begin{aligned}\lambda^{k-1} - \tau A\tilde{v}^k \in \tau\partial\psi(\lambda^{k-1} + \tau(c - A\tilde{v}^k - B\tilde{w}^k + B\theta^k)) \\ + \lambda^{k-1} + \tau(c - A\tilde{v}^k - B\tilde{w}^k + B\theta^k).\end{aligned}$$

By Step 3 in Algorithm 11, the above equation results in:

$$\lambda^{k-1} - \tau A\tilde{v}^k \in \tau\partial\psi(\lambda^k + \tau B\theta^k) + \lambda^k + \tau B\theta^k.$$

From Step 1 in Algorithm 11 and the property of the conjugate function $p \in \partial f(q) \Leftrightarrow q \in \partial f^\star(p)$, we obtain:

$$\lambda^{k-1} - \tau A(\nabla f^\star(A^T\lambda^k) + \delta^k) \in \tau\partial\psi(\lambda^k + \tau B\theta^k) + \lambda^k + \tau B\theta^k.$$

By definition of the function ϕ, we get:

$$\lambda^{k-1} - \tau(\nabla\phi(\lambda^{k-1}) + A\delta^k) \in \tau\partial\psi(\lambda^k + \tau B\theta^k) + \lambda^k + \tau B\theta^k,$$

which is equivalent to:

$$\lambda^k = \text{prox}_{\tau\psi}(\lambda^{k-1} - \tau(\nabla\phi(\lambda^{k-1}) + e^k)) - \tau B\theta^k,$$

with $e^k = A\delta^k$. In order to complete the proof of equation (5.3), we need to show that λ^k is an inexact solution of the proximal operator as defined in equation (2.23) with the error $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$, i.e., to prove:

$$\tau\psi(\lambda^k) + \frac{1}{2}\|\lambda^k - v\|^2 \;\leq\; \epsilon^k + \min_\lambda\left\{\tau\psi(\lambda) + \frac{1}{2}\|\lambda - v\|^2\right\},$$

where $\nu = \lambda^{k-1} - \tau(\nabla\phi(\lambda^{k-1}) + A\delta^k)$. Finally, using

$$\begin{aligned}&\tau\psi(\lambda^k + \tau B\theta^k) + \frac{1}{2}\|\lambda^k + \tau B\theta^k - \nu\|^2 - \tau\psi(\lambda^k) - \frac{1}{2}\|\lambda^k - \nu\|^2 \\ &\leq \tau(\psi(\lambda^k + \tau B\theta^k) - \psi(\lambda^k)) + \frac{1}{2}\|\tau B\theta^k\|^2 \\ &\leq \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2 = \epsilon^k,\end{aligned}$$

equation (5.3) is proved. ■

The proof of Lemma 5.1 is an extension of the proof of Theorem 2 in [11] and the proof in Section 3 in [29]. Based on the equivalence shown in Lemma 5.1 , we can now derive an upper-bound on the difference of the dual function value of the sequence $\{\lambda^k\}$ in Theorem 5.3.

Theorem 5.3. *Let $\{\lambda^k\}$ be generated by the inexact AMA in Algorithm 11. If Assumption 5.1 holds, then for any $k \geq 1$*

$$D(\lambda^\star) - D\left(\frac{1}{k}\sum_{p=1}^{k}\lambda^p\right) \leq \frac{L(\nabla\phi)}{2k}\left(\|\lambda^0 - \lambda^\star\| + 2\Gamma^k + \sqrt{2\Lambda^k}\right)^2 \tag{5.4}$$

where $L(\nabla\phi) = \sigma_f^{-1}\cdot\rho(A)$,

$$\Gamma^k = \sum_{p=1}^{k}\left(\frac{\|A\delta^p\|}{L(\nabla\phi)} + \tau\sqrt{\frac{2L(\psi)\|B\theta^p\| + \|B\theta^p\|^2}{L(\nabla\phi)}}\right), \tag{5.5}$$

$$\Lambda^k = \sum_{p=1}^{k}\frac{\tau^2(2L(\psi)\|B\theta^p\| + \|B\theta^p\|^2)}{2L(\nabla\phi)} \tag{5.6}$$

and λ^0 and $\lambda^\star$ denote the initial points of Algorithm 11 and the optimal solution of Problem 5.1, respectively.

Proof: Lemma 5.1 shows the equivalence between Algorithm 11 and Algorithm 7 with $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$. Then we need to show that the dual defined in Problem 5.2 satisfies Assumption 2.1. $\phi(\lambda)$ and $\psi(\lambda)$ are both convex, since the conjugate functions and linear functions as well as their weighted sum are always convex (the conjugate function is the point-wise supremum of a set of affine functions). Furthermore, since f is strongly convex with σ_f by Assumption 5.1, then we know $f^\star$ has Lipschitz-continuous gradient with Lipschitz constant:

$$L(\nabla f^\star) = \sigma_f^{-1}.$$

It follows that the function ϕ has Lipschitz-continuous gradient $\nabla\phi$ with a Lipschitz constant:

$$L(\nabla\phi) = \sigma_f^{-1}\cdot\rho(A).$$

Hence, the functions ϕ and ψ satisfy Assumption 2.1. Proposition 2.6 completes the proof of the upper-bound in inequality (5.4). ■

Using the complexity upper-bound in Theorem 5.3, we derive sufficient conditions on the error sequences for the convergence of inexact AMA in Corollary 5.1.

Corollary 5.1. *Let $\{\lambda^k\}$ be generated by the inexact AMA in Algorithm 11. If Assumption 5.1 holds, and the constant $L(\psi) < \infty$, the following sufficient conditions on the error sequences $\{\delta^k\}$ and $\{\theta^k\}$ guarantee the convergence of Algorithm 11:*

- *The sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ are finitely summable, i.e., $\sum_{k=1}^{\infty}\|\delta^k\| < \infty$ and $\sum_{k=0}^{\infty}\|\theta^k\| < \infty$.*
- *The sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ decrease at the rate $O(\frac{1}{k^{1+\kappa}})$ for any $\kappa > 0$.*

Proof: By Assumption 5.1, the dual Problem 5.2 satisfies Assumption 7 and the complexity upper-bound in Proposition 2.6 holds. By extending the sufficient conditions on the error sequences for the convergence of inexact proximal-gradient method discussed after Proposition 2.6, we can derive sufficient conditions on the error sequences for inexact AMA with the errors defined in Lemma 5.1 $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$. Since $L(\psi) < \infty$, we have that if the error sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ satisfy the conditions in Corollary 5.1, the complexity upper-bound in Theorem 5.3 converges to zero, as the number of iterations k goes to infinity, which further implies that the inexact AMA algorithm converges to the optimal solution. ■

Remark 5.1. *If the function ψ is an indicator function on a convex set, then the constant $L(\psi)$ is equal to infinity, if for any iteration the inexact solution is infeasible with respect to the convex set. However, if it is guaranteed that for every iteration k the solutions are feasible with respect to the convex set, then the constant $L(\psi)$ is equal to zero.*

Linear convergence of inexact AMA for a quadratic cost

In this section, we study the convergence properties of inexact AMA with a stronger assumption, i.e., the first objective f is a quadratic function and coupling matrix A has full-row rank. We show that with this stronger assumption, the convergence rate of inexact AMA is improved to be a linear rate. We study this special case, since the quadratic functions are very common cost functions in distributed MPC problems,

Assumption 5.2. *We assume that*

- *f is a quadratic function $f = v^T H v + h^T v$ with $H \succ 0$,*
- *A has full-row rank.*

Remark 5.2. *If Assumption 5.2 holds, we know that the first objective $\phi(\lambda)$ in the dual problem in Problem 5.2 is equal to $\phi(\lambda) = \frac{1}{4}(A^T\lambda - h)^T H^{-1}(A^T\lambda - h)$. Then, the Lipschitz constant $L(\nabla\phi)$ is equal to the largest eigenvalue of the matrix $\frac{1}{4}AH^{-1}A^T$, i.e., $L(\nabla\phi) = \lambda_{max}(\frac{1}{4}AH^{-1}A^T)$. In addition, the convexity modulus of $\phi(\lambda)$ is equal to the smallest eigenvalue, i.e., $\sigma_\phi = \lambda_{min}(\frac{1}{4}AH^{-1}A^T)$.*

Theorem 5.4. *Let $\{\lambda^k\}$ be generated by inexact AMA in Algorithm 11. If Assumption 5.1 and 5.2 hold, then for any $k \geq 1$*

$$\|\lambda^k - \lambda^\star\| \leq (1-\gamma)^k \cdot (\|\lambda^0 - \lambda^\star\| + \Gamma^k) \ , \tag{5.7}$$

with

$$\gamma = \frac{\lambda_{min}(AH^{-1}A^T)}{\lambda_{max}(AH^{-1}A^T)},$$

$$\Gamma^k = \sum_{p=1}^{k}(1-\gamma)^{-p} \cdot \left(\frac{\|A\delta^p\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\theta^p\| + \|B\theta^p\|^2}{L(\nabla\phi)}}\right).$$

and λ^0 and $\lambda^\star$ denote the initial point of Algorithm 11 and the optimal solution of Problem 5.1, respectively.

By using the complexity upper-bounds in Theorem 5.4, we derive sufficient conditions on the error sequences, which guarantee the convergence of the inexact AMA algorithm.

Corollary 5.2. *Let $\{\lambda^k\}$ be generated by the inexact AMA in Algorithm 11. If Assumptions 5.1 and 5.2 hold, and the constant $L(\psi) < \infty$, the following sufficient conditions on the error sequences $\{\delta^k\}$ and $\{\theta^k\}$ guarantee the convergence of Algorithm 11:*

- *The sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ are finitely summable, i.e., $\sum_{k=1}^{\infty} \|\delta^k\| < \infty$ and $\sum_{k=0}^{\infty} \|\theta^k\| < \infty$.*
- *The sequences $\{\|\delta^k\|\}$ and $\{\theta^k\}$ decrease at $O(\frac{1}{k^{1+\kappa}})$ for any $\kappa \in \mathbb{Z}_+$. For this case the complexity upper-bound in (5.7) reduces to the same rate as the error sequences.*
- *The sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ decrease at a linear rate.*

Proof: By Assumption 5.1 and 5.2, the dual problem in Problem 5.2 satisfies Assumption 2.2 and the complexity upper-bound in Proposition 2.7 holds for the dual problem. By extending the sufficient conditions on the error sequences discussed after Proposition 2.7, we can derive sufficient conditions on the error sequences for inexact AMA with the errors defined in Lemma 5.1 $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$. Since $L(\psi) < \infty$, we have that if the error sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ satisfy the first and third conditions in Corollary 5.2, the complexity upper-bound in Theorem 5.4 converges to zero, as the number of iterations k goes to infinity, which further implies that the inexact AMA algorithm converges to the optimal solution. For the second sufficient condition in Corollary 5.2, we provide Lemma 5.2 to prove that the second sufficient condition guarantees the convergence of the algorithm. ■

Lemma 5.2. *Let α be a positive number $0 < \alpha < 1$. The following series S^k converges to zero, as the index k goes to infinity*

$$\lim_{k\to\infty} S^k := \lim_{k\to\infty} \alpha^k \cdot \sum_{p=1}^{k} \frac{\alpha^{-p}}{p} = 0 \ .$$

Furthermore, the series S^k converges at the rate $O(\frac{1}{k})$.

The proof of Lemma 5.2 is provided in the appendix in Section 5.6.1. With Lemma 5.2, we can easily show that if the sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ decrease at $O(\frac{1}{k})$, the complexity upper-bound in (5.7) converges at the rate $O(\frac{1}{k})$. Note that this result can be extended to the case where $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ decrease at $O(\frac{1}{k^{1+\kappa}})$ for any $\kappa \in \mathbb{Z}_+$, by following a similar proof as for Lemma 5.2.

5.2.2 Inexact fast alternating minimization algorithm (inexact FAMA)

In this section, we present an accelerated variant of inexact AMA, named the inexact fast alternating minimization Algorithm (inexact FAMA), which is presented in Algorithm 12. It addresses the same problem class as Problem 5.1 and requires

the same assumption as Assumption 5.1 for convergence. Similar to inexact AMA, inexact FAMA allows computation errors in the two minimization steps in the algorithm. Differing from inexact AMA, inexact FAMA involves one extra linear update in Step 4 in Algorithm 12, which improves the optimal convergence rate of the complexity upper-bound of the algorithm from $O(\frac{1}{k})$ to $O(\frac{1}{k^2})$. This is similar to the relationship between the inexact PGM and inexact APGM. By extending the result in Lemma 5.1, we show that inexact FAMA is equivalent to applying inexact APGM to the dual problem. With this equivalence, we further show a complexity upper bound for inexact FAMA by using the result in Proposition 2.8 for inexact APGM.

Algorithm 12 Inexact Fast alternating minimization algorithm (Inexact FAMA)

Require: Initialize $\hat{\lambda}^0 = \lambda^0 \in \mathbb{R}^{N_b}$, and $\tau < \sigma_f/\rho(A)$

for $k = 1, 2, \cdots$ **do**

1: $\tilde{v}^k = \operatorname{argmin}_v \{f(v) + \langle \hat{\lambda}^{k-1}, -Av\rangle\} + \delta^k$.

2: $\tilde{w}^k = \operatorname{argmin}_w \{g(w) + \langle \hat{\lambda}^{k-1}, -Bw\rangle + \frac{\tau}{2}\|c - A\tilde{v}^k - Bw\|^2\} + \theta^k$

3: $\lambda^k = \hat{\lambda}^{k-1} + \tau(c - A\tilde{v}^k - B\tilde{w}^k)$

4: $\hat{\lambda}^k = \lambda^k + \frac{k-1}{k+2}(\lambda^k - \lambda^{k-1})$

end for

Lemma 5.3. *If Assumption 5.1 is satisfied and inexact FAMA and inexact APGM are initialized with the same dual and primal starting sequence, respectively, then applying the inexact FAMA in Algorithm 12 to Problem 5.1 is equivalent to applying inexact APGM in Algorithm 8 to the dual problem defined in Problem 5.2 with the errors $e^k = A\delta^k$ and $\epsilon^k = \tau^2 L(\psi)\|B\theta^k\| + \frac{\tau^2}{2}\|B\theta^k\|^2$, where $L(\psi)$ denotes the Lipschitz constant of the function ψ.*

Proof: The proof follows the same flow of the proof of Lemma 5.1 by replacing λ^{k-1} by $\hat{\lambda}^{k-1}$ computed in Step 4 in Algorithm 12 and showing the following equality

$$\lambda^k = \operatorname{prox}_{\tau\psi,\epsilon^k}(\hat{\lambda}^{k-1} - \tau(\nabla\phi(\hat{\lambda}^{k-1}) + e^k)) \tag{5.8}$$

■

Based on the equivalence shown in Lemma 5.3 , we can now derive an upper-bound on the difference of the dual function value of the sequence $\{\lambda^k\}$ for inexact FAMA in Theorem 5.5.

Theorem 5.5. *Let $\{\lambda^k\}$ be generated by the inexact FAMA in Algorithm 12. If Assumption 5.1 holds, then for any $k \geq 1$*

$$D(\lambda^\star) - D(\lambda^k) \leq \frac{2L(\nabla\phi)}{(k+1)^2}\left(\|\lambda^0 - \lambda^\star\| + 2\Gamma^k + \sqrt{2\Lambda^k}\right)^2 \tag{5.9}$$

where

$$\Gamma^k = \sum_{p=1}^{k} p\left(\frac{\|A\delta^p\|}{L(\nabla\phi)} + \tau\sqrt{\frac{2L(\psi)\|B\theta^p\| + \|B\theta^p\|^2}{L(\nabla\phi)}}\right), \tag{5.10}$$

$$\Lambda^k = \sum_{p=1}^{k} \frac{p^2\tau^2(2L(\psi)\|B\theta^p\| + \|B\theta^p\|^2)}{2L(\nabla\phi)} \tag{5.11}$$

and $L(\nabla\phi) = \sigma_f^{-1} \cdot \rho(A)$.

Proof: The proof is a similar to the proof of Theorem 5.3. Lemma 5.3 shows the equivalence between Algorithm 12 and Algorithm 8. Proposition 2.8 completes the proof of the upper-bound in inequality (5.9). ■

With the results in Theorem 5.5, the sufficient conditions on the errors for the convergence of inexact APGM presented in Section 2.3.1 can be extended to inexact FAMA with the errors defined in Lemma 5.3.

Corollary 5.3. *Let* $\{\lambda^k\}$ *be generated by the inexact FAMA in Algorithm 12. If Assumption 5.1 holds, and the constant* $L(\psi) < \infty$*, the following sufficient conditions on the error sequences* $\{\delta^k\}$ *and* $\{\theta^k\}$ *guarantee the convergence of Algorithm 12:*

- *The series* $\{\|\delta^k\|\}$ *and* $\{\|\theta^k\|\}$ *are finitely summable, i.e.,* $\sum_{k=1}^{\infty} \|\delta^k\| < \infty$ *and* $\sum_{k=0}^{\infty} \|\theta^k\| < \infty$.
- *The sequences* $\{\|\delta^k\|\}$ *and* $\{\|\theta^k\|\}$ *decrease at the rate* $O(\frac{1}{k^{2+\kappa}})$ *for any* $\kappa > 0$.

Proof: By Assumption 5.1, the dual Problem 5.2 satisfies Assumption 7 and the complexity upper-bound in Proposition 2.8 holds. By extending the sufficient conditions on the error sequences discussed after Proposition 2.8, we obtain sufficient conditions on the error sequences for inexact FAMA with the errors defined in Lemma 5.3. Since $L(\psi) < \infty$, we have that if the error sequences $\{\|\delta^k\|\}$ and $\{\|\theta^k\|\}$ satisfy the conditions in Corollary 5.3, the complexity upper-bound in Theorem 5.5 converges to zero, as the number of iterations k goes to infinity, which further implies that the inexact FAMA algorithm converges to the optimal solution. ■

5.2.3 Discussion: inexact AMA and inexact FAMA with bounded errors

In this section, we study a special case where the error sequences $\{\delta^k\}$ and $\{\theta^k\}$ are bounded by constants. This special case attracts a lot of attention, since it appears in many engineering problems in practice, e.g. quantized distributed computation and distributed optimization with constant local computation errors. Previous work includes [20], where the authors studied the complexity upper-bounds for a distributed optimization algorithm with bounded noise on the solutions of local problems. In this section, we will study the errors satisfying Assumption 5.3 and derive the corresponding complexity upper-bounds for inexact AMA, as well as for inexact FAMA, with different assumptions. We show that if the stronger assumption in Assumption 5.2, i.e. the cost function f is a quadratic function, holds, then the complexity bounds for the inexact algorithms with bounded errors converge to a neighbourhood of the origin, as k increases. We show the detailed derivation of the complexity bound for inexact AMA, and this result can be extended to inexact FAMA. If only the basic conditions in Assumption 5.1 are satisfied, the complexity upper-bounds for the inexact algorithms do not converge. We present the complexity upper-bound of inexact FAMA in details for this case, and the result can be easily extended to inexact AMA.

Assumption 5.3. *We assume that the error sequences* δ^k *and* θ^k *are bounded by* $\|\delta^k\| \leq \bar{\delta}$ *and* $\|\theta^k\| \leq \bar{\theta}$ *for all* $k \geq 0$*, where* $\bar{\delta}$ *and* $\bar{\theta}$ *are positive constants.*

Corollary 5.4. *Let $\{\lambda^k\}$ be generated by the inexact AMA in Algorithm 11. If Assumption 5.1, 5.2 and 5.3 hold, then for any $k \geq 1$*

$$\|\lambda^k - \lambda^\star\| \leq (1-\gamma)^k \cdot \|\lambda^0 - \lambda^\star\| + \Delta \ , \tag{5.12}$$

where $\Delta = \frac{1}{\gamma}\left(\frac{\|A\bar{\delta}\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\bar{\theta}\|+\|B\bar{\theta}\|^2}{L(\nabla\phi)}}\right)$, $\gamma = \frac{\lambda_{min}(AH^{-1}A^T)}{\lambda_{max}(AH^{-1}A^T)}$ and λ^0 and $\lambda^\star$ denote the initial point of Algorithm 11 and the optimal solution of Problem 5.1, respectively.

Proof: Since Assumption 5.1 and 5.2 are satisfied, then the results in Theorem 5.4 hold. By Assumption 5.3, we know that the error sequences satisfy $\|\delta^k\| \leq \bar{\delta}$ and $\|\theta^k\| \leq \bar{\theta}$ for all $k \geq 0$. Then the error function Γ^k in Theorem 5.4 is upper-bounded by

$$\Gamma^k \leq \sum_{p=1}^{k}(1-\gamma)^{-p} \cdot \left(\frac{\|A\bar{\delta}\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\bar{\theta}\| + \|B\bar{\theta}\|^2}{L(\nabla\phi)}}\right) \ .$$

Due to the fact that $0 < \gamma < 1$ and the property of geometric series, we get

$$\begin{aligned}(1-\gamma)^k \cdot \Gamma^k &\leq \sum_{p=1}^{k}(1-\gamma)^{k-p} \cdot \left(\frac{\|A\bar{\delta}\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\bar{\theta}\| + \|B\bar{\theta}\|^2}{L(\nabla\phi)}}\right) \\ &\leq \frac{1-(1-\gamma)^k}{\gamma} \cdot \left(\frac{\|A\bar{\delta}\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\bar{\theta}\| + \|B\bar{\theta}\|^2}{L(\nabla\phi)}}\right) \\ &\leq \frac{1}{\gamma} \cdot \left(\frac{\|A\bar{\delta}\|}{L(\nabla\phi)} + \tau\sqrt{\frac{L(\psi)\|B\bar{\theta}\| + \|B\bar{\theta}\|^2}{L(\nabla\phi)}}\right) \ .\end{aligned}$$

Then the upper-bound in Theorem 5.4 implies the upper-bound in (5.12). ■

Remark 5.3. *The inexact AMA algorithm with bounded errors satisfying Assumption 5.1 and 5.2 has one extra constant Δ in the complexity upper-bound in (5.12). Note that due to the stronger assumption in Assumption 5.2, the term Δ remains constant as k increases, the complexity bound in (5.12) converges to a neighbourhood of the origin with the size of Δ, as k goes to infinity.*

Remark 5.4. *For the inexact FAMA in Algorithm 12, if Assumption 5.1 and Assumption 5.3 hold, i.e. the cost is not necessarily quadratic, we can also derive a complexity upper bound as follows*

$$D(\lambda^\star) - D(\lambda^k) \leq \left(\frac{2L(\nabla\phi)\|\lambda^0 - \lambda^\star\|}{(k+1)} + k \cdot \Delta\right)^2 \tag{5.13}$$

with

$$\Delta = \frac{\|A\| \cdot \bar{\delta}}{L(\nabla\phi)} + \frac{3\tau}{2} \cdot \sqrt{\frac{(2L(\psi)\|B\| \cdot \bar{\theta} + \|B\| \cdot \bar{\theta}^2)}{L(\nabla\phi)}}$$

and $L(\nabla\phi) = \sigma_f^{-1} \cdot \rho(A)$. The proof follows the same flow of the proof for Corollary 5.4 by replacing Theorem 5.4 with Theorem 5.5. Compared to the FAMA algorithm without errors, we see that the inexact FAMA with bounded errors has one extra term $k \cdot \Delta$ in the complexity upper-bound in (5.13). Unfortunately, the term $k \cdot \Delta$ increases as k increases. Hence, the complexity bound for the inexact FAMA with bounded errors does not converge to zero, as k goes to infinity.

5.3 Inexact AMA for distributed optimization with an application to distributed MPC

5.3.1 Distributed optimization problem

In this section, we consider a distributed optimization problem on a network of M sub-systems (nodes). The sub-systems communicate according to a fixed undirected graph $G = (\mathcal{V}, \mathcal{E})$. We denote by $\mathcal{N}_i = \{j|(i,j) \in \mathcal{E}\}$ the set of the neighbours of sub-system i. The global optimization variable is denoted by z. The local variable of sub-system i, namely the ith element of z and $z = [z_1^T, \cdots, z_M^T]^T$, is denoted by $[z]_i$. The concatenation of the variable of sub-system i and the variables of its neighbours is denoted by z_i. We define the matrices $E_i \in \mathbb{R}^{\sum_{j\in\mathcal{N}_i} m_j \times \sum_{1\leq i\leq M} m_i}$ and $F_{ji} \in \mathbb{R}^{m_i \times \sum_{j\in\mathcal{N}_i} m_j}$ to be selection matrices with elements in $\{0,1\}$. With these matrices, the variables have the following relationship: $z_i = E_i z$ and $[z]_i = F_{ji} z_j$, $j \in \mathcal{N}_i$, which implies the relation between the local variable $[z]_i$ and the global variable z, i.e.. $[z]_i = F_{ji} E_j z$, $j \in \mathcal{N}_i$. We consider the following distributed optimization problem:

Problem 5.6.

$$\begin{aligned} \min_{z,v} \quad & \sum_{i=1}^{M} f_i(z_i) \\ s.t. \quad & z_i \in \mathbb{C}_i, \quad z_i = E_i v, \quad i = 1, 2, \cdots, M. \end{aligned}$$

where f_i is the local cost function for sub-system i, and the constraint $\mathbb{C}_i$ represents a convex local constraint on the local variables z_i.

Assumption 5.4. *Each local cost function f_i in Problem 5.6 is a strongly convex function with a convexity modulus σ_{f_i} and has a Lipschitz continuous gradient with Lipschitz constant $L(\nabla f_i)$. The set $\mathbb{C}_i$ is a convex set, for all $i = 1, \cdots, M$.*

Remark 5.5. *Recall the problem formulation of inexact AMA and FAMA defined in Problem 5.1. For Problem 5.6, the two objectives are considered to be $f(z) = \sum_{i=1}^{M} f_i(z_i)$ subject to $z_i \in \mathbb{C}_i$ for all $i = 1, \cdots, M$ and $g = 0$. The matrices are $A = I$, $B = -[E_1^T, E_2^T, \cdots, E_M^T]^T$ and $c = 0$. The first objective f consists of a strongly convex function on z and convex constraints. The convex constraints can be considered as indicator functions, which are convex functions. Due to the fact that the sum of a strongly convex and a convex function is strongly convex, the objective f is strongly convex with the modulus σ_f and Problem 5.6 satisfies Assumption 5.1.*

5.3.2 Application: distributed model predictive control

In this section, we consider a distributed linear MPC problem with M sub-systems given in Problem 3.2. We show that it can be written in the form of Problem 5.6. We denote the concatenations of the state sequences and input sequences of agent i and its neighbours by $x_{\mathcal{N}_i}$ and $u_{\mathcal{N}_i}$. The corresponding constraints are $x_{\mathcal{N}_i} \in \mathbb{X}_{\mathcal{N}_i}$ and $u_{\mathcal{N}_i} \in \mathbb{U}_{\mathcal{N}_i}$. We define $v = [x_1^T, x_2^T, \cdots, x_M^T, u_1^T, u_2^T, \cdots, u_M^T]^T$ to be the global variable and $z_i = [x_{\mathcal{N}_i}, u_{\mathcal{N}_i}]$ to be the local variables. $\mathbb{Z}_{\mathcal{N}_i} = \mathbb{X}_{\mathcal{N}_i} \times \mathbb{U}_{\mathcal{N}_i}$ denotes the local constraints on z_i. We use $H_i z_i = h_i$ to represent the dynamical constraint of sub-system i in the distributed MPC problem. Then considering the distributed problem in Problem 5.6, we see that the local cost function f_i for agent i contains all the stage cost functions of the state and input sequences of agent i and its neighbours. The constraint $\mathbb{C}_i$ includes the constraint $\mathbb{Z}_{\mathcal{N}_i}$ and the dynamical constraint $H_i z_i = h_i$. E_i are the matrices selecting the local variables from the global variable. The ith component of v is equal to $[v]_i = [x_i, u_i]$.

Remark 5.6. *If the stage cost functions $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ are strictly convex functions, and the state and input constraints $\mathbb{X}_i$ and $\mathbb{U}_i$ are convex sets, then the conditions in Assumption 5.4 are all satisfied. Furthermore, if the state cost functions $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ are set to be strictly positive quadratic functions, then the distributed optimization problem originating from the distributed MPC problem further satisfies Assumption 5.2.*

Remark 5.7. *For the case where the distributed MPC problem has only input constraints and the state coupling matrices in the linear dynamics are $A_{ij} = 0$ for any $i \neq j$, we can eliminate all the state variables in the distributed MPC problem and only have the input variables as the optimization variables. For this case, if the stage cost functions $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ are strictly convex functions with respect to the input variables and the local linear dynamical system $x_i(t+1) = A_{ii}x_i(t) + \sum_{j\in\mathcal{N}_i} B_{ij}u_j(t)$ is controllable, then the resulting distributed optimization problem satisfies Assumption 5.2. The details of this formulation can be found in Problem 6.4.*

5.3.3 Inexact AMA and Inexact FAMA for distributed optimization

In this section, we apply inexact AMA and inexact FAMA to the distributed optimization problem in Problem 5.6, originating from the distributed MPC problem in Problem 3.2. The concept is to split the distributed optimization into small and local problems according to the physical couplings of the sub-systems. Algorithm 13 and Algorithm 14 represent the algorithms. Note that Step 2 in inexact AMA and inexact FAMA, i.e., Algorithm 11 and Algorithm 12, are simplified to be a consensus step in Step 3 in Algorithm 13 and Algorithm 14, which requires only local communication. In the algorithms, δ_i^k represents the computation error of the local problems.

Remark 5.8. *Note that for every iteration k, Algorithms 13 and 14 only need local communication and the computations can be performed in parallel for every subsystem. For Algorithm 13 and 14, we assume that all the communication steps are done in a synchronous way.*

Algorithm 13 Inexact Alternating Minimization Algorithm for DMPC

Require: Initialize $\lambda_i^0 = 0 \in \mathbb{R}^{z_i}$, and $\tau < \min_{1\leq i\leq M}\{\sigma_{f_i}\}$
for $k = 1, 2, \cdots$ **do**
1: $\tilde{z}_i^k = \text{argmin}_{z_i\in\mathbb{C}_i}\{f_i(z_i) + \langle\lambda_i^{k-1}, -z_i\rangle\} + \delta_i^k$
2: Send $\tilde{z}_i^k$ to all the neighbours of agent i.
3: $[\tilde{v}^k]_i = \frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}^{M}[\tilde{z}_j^k]_i$.
4: Send $[\tilde{v}^k]_i$ to all the neighbours of agent i.
5: $\lambda_i^k = \lambda_i^{k-1} + \tau(E_i\tilde{v}^k - \tilde{z}_i^k)$
end for

Algorithm 14 Inexact Fast alternating minimization algorithm for DMPC

Require: Initialize $\lambda_i^0 = \hat{\lambda}_i^0 \in \mathbb{R}^{z_i}$, and $\tau < \min_{1\leq i\leq M}\{\sigma_{f_i}\}$
for $k = 1, 2, \cdots$ **do**
1: $\tilde{z}_i^k = \text{argmin}_{z_i\in\mathbb{C}_i}\{f_i(z_i) + \langle\hat{\lambda}_i^{k-1}, -z_i\rangle\} + \delta_i^k$
2: Send $\tilde{z}_i^k$ to all the neighbours of agent i.
3: $[\tilde{v}^k]_i = \frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}^{M}[\tilde{z}_j^k]_i$.
4: Send $[\tilde{v}^k]_i$ to all the neighbours of agent i.
5: $\lambda_i^k = \hat{\lambda}_i^{k-1} + \tau(E_i\tilde{v}^k - \tilde{z}_i^k)$
6: $\hat{\lambda}_i^k = \lambda_i^k + \frac{k-1}{k+2}(\lambda_i^k - \lambda_i^{k-1})$
end for

We provide a lemma showing that considering Algorithm 13 there exists a Lipschitz constant $L(\psi)$ equal to zero. The results can be easily extended to Algorithm 14. This result is required by the proofs of the complexity upper-bounds in Theorem 5.3, 5.4 and 5.5.

Lemma 5.4. *Let the sequence $\{\lambda^k\}$ be generated by Algorithm 13. For all $k \geq 0$ it holds that $E^T\lambda^k = 0$ and the Lipschitz constant of the second objective in the dual problem of Problem 5.6 $L(\psi)$ is equal to zero.*

Proof: We first prove that for all $k \geq 0$, the sequence $\{\lambda^k\}$ satisfies $E^T\lambda^k = 0$ for all $k \geq 0$. We know that Step 3 in Algorithm 13 is equivalent to the following update

$$\tilde{v}^k = \mathcal{M}\cdot\sum_{i=1}^{M} E_i^T\cdot\tilde{z}_i^k = \mathcal{M}\cdot E^T\cdot\tilde{z}^k \ ,$$

with $\mathcal{M} = blkdiag(\frac{1}{|\mathcal{N}_1|}\cdot I_1, \cdots, \frac{1}{|\mathcal{N}_i|}\cdot I_i, \cdots, \frac{1}{|\mathcal{N}_M|}\cdot I_M) = (E^TE)^{-1}$, where $|\mathcal{N}_i|$ denotes the number of the elements in the set $\mathcal{N}_i$, and I_i denotes an identity matrix with the dimension of the ith component of v, denoted as $[v]_i$. From Step 5 in Algorithm 13, for all $k \geq 1$ we have that

$$\lambda^k = \lambda^{k-1} + \tau(E\tilde{v}^k - \tilde{z}^k) \ .$$

By multiplying the matrix E^T to both sides, we have

$$\begin{aligned} E^T\lambda^k &= E^T\lambda^{k-1} + \tau(E^TE\tilde{v}^k - E^T\tilde{z}^k) \\ &= E^T\lambda^{k-1} + \tau(E^TE\mathcal{M}E^T\tilde{z}^k - E^T\tilde{z}^k) \ . \end{aligned}$$

Since $\mathcal{M} = (E^T E)^{-1}$, the above equality becomes

$$E^T \lambda^k = E^T \lambda^{k-1} + \tau(E^T \tilde{z}^k - E^T \tilde{z}^k) = E^T \lambda^{k-1} \ .$$

From the initialization in Algorithm 13, we know $E^T \lambda^0 = E^T \cdot 0 = 0$. Then by induction, we can immediately prove that for all $k \geq 0$ it holds that $E^T \lambda^k = 0$. We can now show that for all $E^T \lambda = 0$, a Lipschitz constant of the second objective in the dual problem in Problem 5.2 $L(\psi)$ is equal to zero. Since $g = 0$, $B = -E$ and $c = 0$, then the second objective in the dual problem is equal to

$$\begin{aligned} \psi(\lambda) &= g^\star(B^T \lambda) - c^T \lambda = g^\star(E^T \lambda) \\ &= \sup_v (v^T E^T \lambda - 0) = \begin{cases} 0 & \text{if } E^T \lambda = 0 \\ \infty & \text{if } E^T \lambda \neq 0. \end{cases} \ . \end{aligned}$$

The function $\psi(\cdot)$ is an indicator function on the nullspace of matrix E^T. For all λ satisfying $E^T \lambda = 0$, the function $\psi(\cdot)$ is equal to zero. Hence, zero is a Lipschitz constant of the function $\psi(\cdot)$ for all $E^T \lambda = 0$. ■

After proving Lemma 5.4, we are ready to show the main theoretical properties of Algorithm 13 and 14.

Corollary 5.5. *Let $\{\lambda^k = [\lambda_1^{k^T}, \cdots, \lambda_M^{k^T}]^T\}$ be generated by Algorithm 13. If Assumption 5.4 is satisfied and the inexact solutions $\tilde{z}_i^k$ for all $k \geq 1$ are feasible, i.e., $\tilde{z}_i^k \in \mathbb{C}_i$, then for any $k \geq 1$*

$$D(\lambda^\star) - D\left(\frac{1}{k}\sum_{p=1}^{k} \lambda^p\right) \leq \frac{L(\nabla\phi)}{2k}\left(\|\lambda^0 - \lambda^\star\| + 2\sum_{p=1}^{k} \frac{\|\delta^p\|}{L(\nabla\phi)}\right)^2 \tag{5.14}$$

where $D(\cdot)$ is the dual function of Problem 5.6, $\lambda^0 = [\lambda_1^{0^T}, \cdots, \lambda_M^{0^T}]^T$ and $\lambda^\star$ are the starting sequence and the optimal sequence of the Lagrangian multiplier, respectively, and $\delta^p = [\delta_1^{p^T}, \cdots, \delta_M^{p^T}]^T$ denotes the global error sequence. The Lipschitz constant $L(\nabla\phi)$ is equal to σ_f^{-1}, with $\sigma_f = \min\{\sigma_{f_1}, \cdots, \sigma_{f_M}\}$.

Proof: As we presented in Remark 5.5, Problem 5.6 is split as follows: $f = \sum_{i=1}^{M} f_i(z_i)$ with the constraints $z_i \in \mathbb{C}_i$ for all $i = 1, \cdots, M$ and $g = 0$. The matrices are $A = I$, $B = -E$ and $c = 0$. If Assumption 5.4 holds, then this splitting problem satisfies Assumption 5.1 with the convexity modulus σ_f. From Theorem 5.3, we know that the sequence $\{\lambda^k\}$ generated by inexact AMA in Algorithm 13, satisfies the complexity upper bound in (5.4) with Γ^k and Λ^k in (5.5) and (5.6) with $\delta^k = [\delta_1^{k^T}, \cdots, \delta_M^{k^T}]^T$ and $\theta^k = 0$. By Lemma 5.4, it follows that the constant $L(\psi)$ in Λ^k is equal to zero. The Lipschitz constant of the gradient of the dual objective is equal to $L(\nabla\phi) = \sigma_f^{-1} \cdot \rho(A) = \sigma_f^{-1}$ with $\sigma_f = \min\{\sigma_{f_1}, \cdots, \sigma_{f_M}\}$. Hence, we can simplify the complexity upper bound in (5.4) for Algorithm 13 to be inequality (5.14). ■

As we discussed in Remark 5.6, if the state cost functions $l_i(\cdot,\cdot)$ and $l_i^f(\cdot)$ in the distributed MPC problem are strictly positive quadratic functions, then the distributed optimization problem originating from the distributed MPC problem satisfies Assumption 5.2, which according to Theorem 5.3 implies a linearly decreasing upper-bound given in Corollary 5.6.

Corollary 5.6. *Let $\{\lambda^k = [\lambda_1^{k^T}, \cdots, \lambda_M^{k^T}]^T\}$ be generated by Algorithm 13. If Assumption 5.4 is satisfied, the local cost function f_i is a strictly positive quadratic function, and the inexact solutions $\tilde{z}_i^k$ for all $k \geq 1$ are feasible, i.e., $\tilde{z}_i^k \in \mathbb{C}_i$, then for any $k \geq 1$*

$$\|\lambda^k - \lambda^\star\| \leq (1-\gamma)^{k+1} \cdot \left(\|\lambda^0 - \lambda^\star\| + \sum_{p=0}^{k} (1-\gamma)^{-p-1} \cdot \frac{\|\delta^p\|}{L(\nabla\phi)} \right) , \tag{5.15}$$

where $\gamma = \frac{\lambda_{min}(H)}{\lambda_{max}(H)}$, and λ^0 and $\lambda^\star$ are the starting point and the optimal solution of the Lagrangian multiplier, respectively. The Lipschitz constant $L(\nabla\phi)$ is equal to σ_f^{-1}, where $\sigma_f = \min\{\sigma_{f_1}, \cdots, \sigma_{f_M}\}$.

Proof: In Algorithm 14, the variable $\hat{\lambda}_i^k$ is a linear function of λ_i^k and λ_i^{k-1}. This preserves all properties shown in Lemma 5.4 for Algorithm 14. Then, Corollary 5.6 can be easily proven by following the same steps as in the proof of Corollary 5.5 by replacing Theorem 5.3 by Theorem 5.4. ■

Corollary 5.7. *Let $\{\lambda^k = [\lambda_1^{k^T}, \cdots, \lambda_M^{k^T}]^T\}$ be generated by Algorithm 14. If Assumption 5.4 is satisfied and the inexact solutions $\tilde{z}_i^k$ for all $k \geq 1$ are feasible, i.e. $\tilde{z}_i^k \in \mathbb{C}_i$, then for any $k \geq 1$*

$$D(\lambda^\star) - D(\lambda^k) \leq \frac{2L(\nabla\phi)}{(k+1)^2} \left(\|\lambda^0 - \lambda^\star\| + 2M \sum_{p=1}^{k} p \frac{\delta^p}{L(\nabla\phi)} \right)^2 . \tag{5.16}$$

where $D(\cdot)$ is the dual function of Problem 5.6, λ^0 and $\lambda^\star$ are the starting point and the optimal solution of the Lagrangian multiplier, respectively. The Lipschitz constant $L(\nabla\phi)$ is equal to σ_f^{-1}, where $\sigma_f = \min\{\sigma_{f_1}, \cdots, \sigma_{f_M}\}$.

Proof: It follows from the same proof as Corollary 5.5 by replacing Theorem 5.3 by Theorem 5.5. ■

Remark 5.9. *For the case that all the local problems are solved exactly, i.e., $\delta_i^k = 0$ for all $k \geq 1$, Algorithm 13 and Algorithm 14 reduce to standard AMA and FAMA, and converge to the optimal point at the rate of the complexity upper-bounds.*

Remark 5.10. *The sufficient conditions on the errors for convergence given in Corollary 5.1, 5.2 and 5.3 can be directly extended to the error sequence $\{\delta^k\}$.*

5.3.4 An approach to certify the number of iterations for solving local problems satisfying the global error conditions

We have shown that the inexact distributed optimization algorithms in Algorithm 13 and 14 allow one to solve the local problems, i.e. Step 1 in Algorithm 13 and 14, inexactly. In this section, we will address two questions: which algorithms are suitable for solving the local problems; and what termination conditions for the local algorithms guarantee that the computational error of the local solution satisfies the sufficient conditions on the errors for the global distributed optimization algorithms.

We apply the proximal gradient method for solving the local problems in Step 1 in Algorithm 13 and 14, and propose an approach to certify the number of iterations for their solution, by employing a warm-start strategy and the complexity upper-bounds of the proximal gradient method. The approach guarantees that the local computational errors δ_i^k decrease with a given rate, that satisfies the sufficient conditions discussed in Remark 5.10, ensuring convergence of the inexact distributed optimization algorithm to the optimal solution. We define a decrease function α^k satisfying the sufficient conditions, for example $\alpha^k = \alpha^0 \cdot \frac{1}{k^2}$, where α^0 is a positive number.

Gradient-based method

The local problems in Step 1 in Algorithm 13 and 14 are optimization problems with strongly convex cost functions and convex constraints. From Corollary 5.5, 5.6 and 5.7, we know that the inexact solution $\tilde{z}_i^k$ needs to be a feasible solution subject to the local constraint $\mathbb{C}_i$, i.e., $\tilde{z}_i^k \in \mathbb{C}_i$ for all $k \geq 1$. Therefore, a good candidate algorithm for solving the local problems should have the following three properties: the algorithm can solve convex optimization problems efficiently; if the algorithm is stopped early, i.e., only a few number of iterations are implemented, the sub-optimal solution is feasible with respect to the local constraint $\mathbb{C}_i$; and there exists a certificate on the number of iterations to achieve a given accuracy of the sub-optimal solution. Gradient-based methods satisfy these requirements, have simple and efficient implementations, and offer complexity upper-bounds on the number of iterations [28]. These methods have been studied in the context of MPC in [3], [8] and [37].

We apply the proximal gradient method in Algorithm 15 for solving the local problems in Step 1 in Algorithm 13 and 14. The local optimization problems at iteration k are parametric optimization problems with the parameter λ_i^{k-1}. We denote the optimal function as

$$\mathbf{z}_i^\star(\lambda_i) := \operatorname{argmin}_{z_i \in \mathbb{C}_i} \{ f_i(z_i) + \langle \lambda_i, -z_i \rangle \} \ . \tag{5.17}$$

The solution of the optimal function at λ_i^{k-1} is denoted as $z_i^{k,\star} := \mathbf{z}_i^\star(\lambda_i^{k-1})$. The function $\mathbf{z}_i^\star(\cdot)$ has a Lipschitz constant $L(\mathbf{z}_i^\star)$ satisfying as $\|\mathbf{z}_i^\star(\lambda_{i_1}) - \mathbf{z}_i^\star(\lambda_{i_1})\| \leq L(\mathbf{z}_i^\star) \cdot \|\lambda_{i_1} - \lambda_{i_2}\|$ for any λ_{i_1} and λ_{i_2}. Motivated by the fact that the difference between the parameters λ_i^{k-1} and λ_i^k is limited and measurable for each k, i.e. $\beta_i^k = \|\lambda_i^{k-1} - \lambda_i^k\| = \tau(E_i \tilde{v}^{k-1} - \tilde{z}_i^{k-1})$, we use a warm-starting strategy to initialize the local problems, i.e., we use the solution $\tilde{z}_i^{k-1}$ from the previous step $k-1$ as the initial solution for Algorithm 15 for step k.

Remark 5.11. *Note that we initialize the vectors $\tilde{v}^{k-1}$, $\tilde{z}_i^{k-1}$ and $\tilde{z}_i^{k-1}$ for $k = 1$ in Algorithm 13 to be zero vectors.*

Proposition 5.1 (Proposition 3 in [22])**.** *Let $z_i^{k,j}$ be generated by Algorithm 15. If Assumption 5.4 holds, then for any $j \geq 0$ we have:*

$$\|z_i^{k,j} - z_i^{k,\star}\| \leq \|z_i^{k,0} - z_i^{k,\star}\| \cdot (1-\gamma)^j \ , \tag{5.18}$$

where $\gamma = \frac{\sigma_{f_i}}{L(\nabla f_i)}$ and $z_i^{k,0}$ and $z_i^{k,\star}$ denote the initial sequence of Algorithm 7 and the optimal solution of the problem in Step 6 in Algorithm 13 at iteration k, respectively.

Algorithm 15 Gradient-based Method for solving Step 1 in Algorithm 13 at iteration k

Require: Initialize $\alpha^k = \alpha^0 \cdot \frac{1}{k^2}$, $\beta^k = \|\tau(E_i\tilde{v}^{k-1} - \tilde{z}_i^{k-1})\|$, λ_i^{k-1}, $z_i^{k,0} = \tilde{z}_i^{k-1}$ and $\tau_i < \frac{1}{L(\nabla f_i)}$
Compute J_k satisfying (5.19)
for $j = 1, 2, \cdots, J_k$ **do**
$z_i^{k,j} = \mathrm{Proj}_{\mathbb{C}_i}(z_i^{k,j-1} - \tau(\nabla f_i(z_i^{k,j-1}) - \lambda_i^{k-1}))$
end for
$\tilde{z}_i^k \leftarrow z_i^{k,J_k}$

Termination condition on the number of iterations for solving local problems

The topic, i.e. to develop methods to on-line terminate the number of iterations until a given accuracy is reached, has been studied in previous work, e.g. [51], [43] and [21]. In [51] and [43], the authors proposed dual decomposition based optimization methods for solving quadratic programming problems and presented termination conditions to guarantee a prespecified accuracy. However, for these methods, it is quite difficult to guarantee the feasibility of a sub-optimal solution on-line, or it requires to tighten the constraints on-line to guarantee the feasibility, which is quite conservative in practice. Hence, though these methods provide interesting termination conditions to guarantee a prespecified accuracy, they are not good candidates for solving our problem. In [21], the authors propose an inexact decomposition algorithm for solving distributed optimization problems by employing smoothing techniques and an excessive gap condition as the termination condition on the number of iterations to achieve a given accuracy. To certify the termination condition, this method requires to measure the values of the global primal and dual functions on-line, which requires full communication on the network and is not satisfied in our distributed framework. In addition, this method does not provide any algorithms for solving the local problems.

By employing the complexity upper-bounds in Proposition 5.1 for Algorithm 15, derived from [22], we propose a termination condition in (5.19) to find the number of iterations J_k for Algorithm 15, which guarantees that the local computational error is upper-bounded by the predefined decrease function α^k, i.e., $\|\delta_i^k\| \leq \alpha^k$, shown in Lemma 5.5.

Lemma 5.5. *If the number of iterations J_k in Algorithm 15 satisfies*

$$J_k \geq \lceil \log_{(1-\gamma)} \frac{\alpha^k}{\alpha^{k-1} + L(\boldsymbol{z}_i^\star)\beta^k} \rceil \tag{5.19}$$

for all $k \geq 1$, then the computational error for solving the local problem in Step 6 in Algorithm 13 δ_i^k satisfies $\|\delta_i^k\| \leq \alpha^k$.

Proof: We will prove Lemma 5.5 by induction.

- Base case: When $k = 1$, the vectors $\tilde{v}^{k-1}$, $\tilde{z}_i^{k-1}$ and $\tilde{z}_i^{k-1}$ are initialized as zero vectors. By Proposition 2.6 and the fact $z_i^{1,0} = \tilde{z}_i^0 = 0$, we know

$$\begin{aligned} \|z_i^{1,J_1} - z_i^{1,\star}\| &\leq \|z_i^{1,0} - z_i^{1,\star}\| \cdot (1-\gamma)^{J_1} \\ &= \|0 - z_i^{1,\star}\| \cdot (1-\gamma)^{J_1} \quad . \end{aligned}$$

Due to the definition of the function α^k, it follows that the term above is upper-bounded by $\alpha^0 \cdot (1-\gamma)^{J_1}$. By the fact that $\beta^0 = \|\tau(E_i\tilde{v}^0 - \tilde{z}_i^0)\| = 0$ and J_1 satisfies (5.19), it is further upper-bounded by α^1. Then we get that

$$\|\delta^1\| = \|\tilde{z}_i^1 - z_i^{1,\star}\| = \|z_i^{1,J_1} - z_i^{1,\star}\| \leq \alpha^1 \ .$$

- Induction step: Let $g \geq 1$ be given and suppose that $\|\delta^g\| \leq \alpha^g$. We will prove that $\|\delta^{g+1}\| \leq \alpha^{g+1}$ By Proposition 2.6 and the warm-starting strategy, i.e. $z_i^{k,0} = \tilde{z}_i^{k-1} = z_i^{k-1,J_{k-1}}$, we know

$$\begin{aligned}
&\|\delta^{g+1}\| = \|z_i^{g+1,J_{g+1}} - z_i^{g+1,\star}\| \\
&\leq \|z_i^{g+1,0} - z_i^{g+1,\star}\| \cdot (1-\gamma)^{J_{g+1}} \\
&= \|z_i^{g,J_g} - z_i^{g+1,\star}\| \cdot (1-\gamma)^{J_{g+1}} \\
&\leq (\|z_i^{g,J_g} - z_i^{g,\star}\| + \|z_i^{g,\star} - z_i^{g+1,\star}\|) \cdot (1-\gamma)^{J_{g+1}} \\
&\leq (\delta^g + L(\mathbf{z}_i^\star) \cdot \beta^{g+1}) \cdot (1-\gamma)^{J_{g+1}} \ .
\end{aligned}$$

Due to the induction assumption and the fact that J_g satisfies (5.19), it follows that $\|\delta^{g+1}\| \leq \alpha^{g+1}$.

We conclude that by the principle of induction, it holds that $\|\delta^k\| \leq \alpha^k$ for all $k \geq 1$. ■

Corollary 5.8. *If Assumption 5.4 holds and the decrease rate of the function α^k satisfies the corresponding sufficient conditions presented in Corollary 5.1 and 5.3, then Algorithm 13 and 14 converge to the optimal solution, with Algorithm 15 solving the local problem in Step 1. Furthermore, If the local cost function f_i is a strictly positive quadratic function, and the decrease rate of the function α^k satisfies the sufficient conditions presented in Corollary 5.2, then Algorithm 13 converges to the optimal solution, with Algorithm 15 solving the local problem in Step 1.*

Proof: Lemma 5.5 shows that the computation error $\delta^k = [\delta_1^{k^T}, \cdots, \delta_M^{k^T}]^T$ is upper-bounded by the function α^k. Since α^k decrease at a rate satisfying the corresponding sufficient conditions discussed in Corollary 5.1, 5.2 and 5.3, then Algorithms 13 and 14 converges to the optimal solution. ■

Remark 5.12. *All the information required by the proposed on-line certification method, i.e.,by Algorithm 15, as well as the condition for J_k in (5.19), can be obtained on-line and locally.*

Computation of the Lipschitz constant $L(\mathbf{z}_i^\star)$

In the above proposed on-line certification method, the Lipschitz constant of the optimal solution function $\mathbf{z}_i^\star(\lambda_i^{k-1})$, $L(\mathbf{z}_i^\star)$, plays an important role. While it is generally difficult to compute this Lipschitz constant, it can be computed for special cases, such as positive quadratic functions.

Lemma 5.6. *Let the local cost function be a quadratic function, i.e.* $f_i(z_i) = \frac{1}{2} z_i^T H_i z_i + h_i^T z_i$ *with* $H_i \succ 0$. *A Lipschitz constant of the function* $\boldsymbol{z}_i^\star(\lambda_i)$ *defined in (5.17) is equal to* $\frac{1}{\rho_{min}(H_i)}$, *i.e.*

$$\|\boldsymbol{z}_i^\star(\lambda_{i_1}) - \boldsymbol{z}_i^\star(\lambda_{i_2})\| \leq \frac{1}{\rho_{min}(H_i)} \cdot \|\lambda_{i_1} - \lambda_{i_2}\|. \tag{5.20}$$

Proof: Since $H_i \succ 0$, *we get* $H_i = D \cdot D^T$ *with* D *invertible, which implies*

$$\begin{aligned} \boldsymbol{z}_i^\star(\lambda_i) &= \text{argmin}_{z_i \in \mathbb{C}_i} \ \frac{1}{2} z_i^T H_i z_i + (h_i - \lambda_i)^T z_i \\ &= \text{argmin}_{z_i \in \mathbb{C}_i} \ \frac{1}{2} \|D^T z_i + D^{-1}(h_i - \lambda_i)\|^2 \end{aligned}$$

Let $v = D^T z_i$. *The optimization problem above becomes*

$$\boldsymbol{v}^\star(\lambda_i) = \text{argmin}_{D^{T^{-1}} v \in \mathbb{C}_i} \ \ \frac{1}{2} \|v + D^{-1}(h_i - \lambda_i)\|^2 \ ,$$

which can be seen as the projection of the point $D^{-1}(h_i - \lambda_i)$ *onto the set* $\bar{\mathbb{C}}_i := \{v \mid D^{-1} v_i \in \mathbb{C}_i\}$. *Since* $\mathbb{C}_i$ *is convex, then* $\bar{\mathbb{C}}_i$ *is convex as well. It follows directly from Proposition 2.2.1 in [24] that*

$$\|\boldsymbol{v}^\star(\lambda_{i_1}) - \boldsymbol{v}^\star(\lambda_{i_2})\| \leq \|D^{-1} \cdot (\lambda_{i_1} - \lambda_{i_2})\| \ .$$

By $z_i = D^{T^{-1}} v$, *we get*

$$\begin{aligned} \|\boldsymbol{z}_i^\star(\lambda_{i_1}) - \boldsymbol{z}_i^\star(\lambda_{i_2})\| &\leq \|D^{-1}\| \cdot \|D^{-1} \cdot (\lambda_{i_1} - \lambda_{i_2})\| \\ &\leq \|D^{-1}\|^2 \cdot \|\lambda_{i_1} - \lambda_{i_2}\| \\ &\leq \frac{1}{\rho_{min}(H_i)} \cdot \|\lambda_{i_1} - \lambda_{i_2}\| \ . \end{aligned}$$

■

5.4 Numerical example

This section illustrates the theoretical findings of the chapter and demonstrates the performance of inexact AMA by solving a randomly generated distributed MPC problem with 40 sub-systems. For this example, we assume that the sub-systems are coupled only in the control input:

$$x_i(t+1) = A_{ii} x_j(t) + \sum_{j \in \mathcal{N}_i} B_{ij} u_j(t) \quad i = 1, 2, \cdots, M,$$

The input-coupled dynamics allow us to eliminate the states of the distributed MPC problem, such that the optimization variable in the distributed optimization problems is the control sequence $u = [u_1^T, \cdots, u_M^T]^T$, with $u_i = [u_i^T(0), u_i^T(1), \cdots, u_i^T(N)]^T$. The input-coupled dynamics also allow us to simplify the projection step in Algorithm 15. Examples with this structure include systems sharing one resource, e.g. a water-tank system or an energy storage system.

We randomly generate a connected network with 40 agents. Each sub-system has three states and two inputs. The dynamical matrices A_{ii} and B_{ij} are randomly generated, i.e. generally dense, and the local systems are controllable. The input constraint $\mathbb{U}_i$ for sub-system i is set to be $\mathbb{U}_i = \{u_i(t)| -0.4 \leq u_i(t) \leq 0.3\}$. The horizon of the MPC problem is set to be $N = 11$. The local cost functions are set to be quadratic functions, i.e. $l_i(x_i(t), u_i(t)) = x_i^T(t)Qx_i(t) + u_i^T(t)Ru_i(t)$ and $l_i^f(x_i(N)) = x_i^T(N)Px_i(N)$, where Q, R and P are identity matrices. Therefore, the distributed optimization problem resulting from the distributed MPC satisfies Assumption 5.4, and the local cost functions f_i are strictly positive quadratic functions. Hence, the results in Corollary 5.6 hold. The initial states $\bar{x}_i$ are chosen, such that more than 70% of the elements of the vector $u^\star$ are at the constraints.

In Fig. 5.1, we demonstrate the convergence performance of inexact AMA for solving the distributed optimization problem in Problem 5.6, originating from the randomly generated distributed MPC problem, applying Algorithm 13. In this simulation, we compare the performance of inexact AMA with three different kinds of errors for δ^k with exact AMA, for which the errors are equal to zero. Note that these errors are synthetically constructed to specify different error properties. We solve the local problems to optimality and then add the feasible errors with predefined decreasing rates to the local optimal solution. The black line shows the performance of exact AMA. The blue, red and green lines show the performance of inexact AMA, where the errors δ^k are set to be decreasing at the rates of $O(\frac{1}{k})$, $O(\frac{1}{k^2})$ and $O(\frac{1}{k^3})$, respectively. Note that all three errors satisfy the sufficient condition for convergence in Corollary 5.2. We can observe that as the number of iterations k increases, the differences $\|u^k - u^\star\|$ decrease for all the cases. However, the convergence speed is quite different. For the exact AMA algorithm (black line), it decreases linearly, which supports the results in Corollary 5.6. For the three cases for inexact AMA (blue, red and green lines), we can see that the differences $\|u^k - u^\star\|$ decrease more slowly than for exact AMA, and the decrease rates are same as the decrease rate of the errors, which supports the theoretical findings in Corollary 5.2.

The second simulation illustrates the convergence properties of inexact AMA, where the proximal gradient method in Algorithm 15 is applied to solve the local problems in Step 2 in Algorithm 13. Note that we initialize the vectors $\tilde{v}^{k-1}$, $\tilde{z}_i^{k-1}$ and $\tilde{z}_i^{k-1}$ for $k = 1$ in Algorithm 13 to be zero vectors. In this experiment Algorithm 15 is stopped after the number of iterations providing that the local computation error δ_i^k decreases at a certain rate. The error decrease rate is selected to be $O(\frac{1}{k})$, i.e., the decrease function α^k is set to be $\alpha^k = \alpha^0 \cdot \frac{1}{k}$. This decrease rate satisfies the second sufficient condition in Corollary 5.2. Hence, theoretically the convergence of the inexact AMA is guaranteed. In order to satisfy $\|\delta_i^k\| \leq \alpha^k$, the number of iterations for the proximal gradient method J_K in Algorithm 15 is chosen according to the certification method presented in Section 5.3.4 such that condition (5.19) is satisfied. Note that we use a warm-starting strategy for the initialization of Algorithm 15.

Fig. 5.2 shows the comparison of the performance of exact AMA and inexact AMA. We can observe that the black (exact AMA) and red lines basically overlap (inexact AMA with Algorithm 15 solving local problems with the numbers of iterations J_k satisfying (5.19)). Inexact AMA converges to the optimal solution as the iterations increase, and shows almost the same performance as exact AMA.

Fig. 5.3 shows the local error sequence δ_i^k for the case that the number of iterations J_k for Algorithm 15 satisfies the condition in (5.19). We can observe that the global error sequence $\delta^k = [\delta_1^k, \cdots, \delta_M^k]$ is upper-bounded by the decrease function α^k. As k is small, the upper-bound α^k is tight to the error sequence. As k increases, the error decreases faster and the bound becomes loose.

Fig. 5.4 shows the comparison of the numbers of iterations for Algorithm 15, computed using two different approaches. **Approach 1** uses the termination condition proposed in Section 5.3.4. In **Approach 2**, we first compute the optimal solution of the local problem $z_i^{k,\star}$ and then run the proximal gradient method to find the smallest number of iterations providing that the difference of the local sub-optimal solution satisfies the decrease function α^k, i.e. $\|z_i^{k,j} - z_i^{k,\star}\| \leq \alpha^k$. **Approach 2** is therefore the exact minimal number, whereas **Approach 1** uses a bound on the minimal number. Note that the second approach guarantees $\|\delta_i^k\| \leq \alpha^k$ for all k, however, this method is not practically applicable, since the optimal solution $z_i^{k,\star}$ is unknown. Its purpose is merely to compare with the proposed certification method and to show how tight the theoretical bound in (5.19) is. For both techniques, we use a warm-starting strategy for initialization of the proximal gradient method to solve the local problems for each k in Algorithm 13. In Fig. 5.4, the green line and region result from the termination condition proposed in Section 5.3.4, and the pink line and region result from the second approach. The solid green and red lines show the average value of the numbers of iterations for the proximal gradient method for solving the local problems over the 40 sub-systems. The upper and lower boundaries of the regions show the maximal and minimal number of iterations, respectively. We can observe that the result given by the proposed certification method is worse than the second method. The maximal number of iterations for the proposed certification method(green region) is equal to 7, while for the second method (the red region) it is equal to 4. However, Fig. 5.4 shows that the certification approach in (5.19), which can be performed locally, is reasonably tight and the provided number of iterations is close to the actual iterations required to satisfy the desired error.

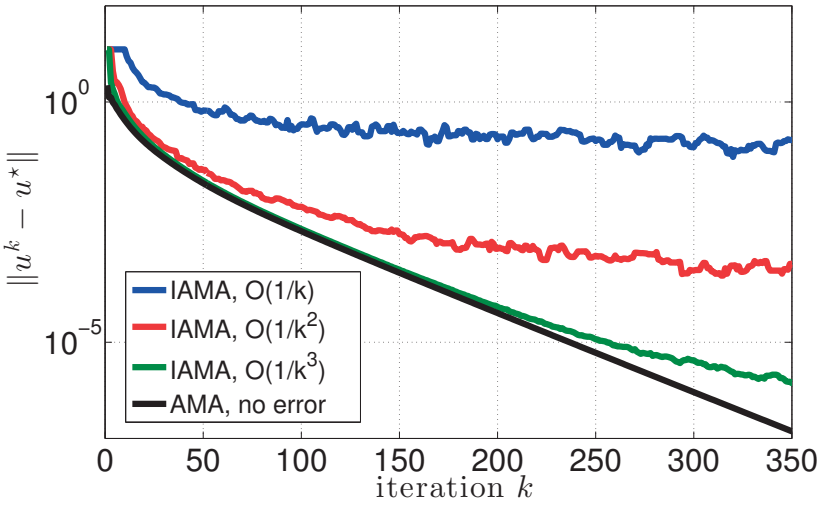


Figure 5.1 – The comparison of the performance of AMA and inexact AMA (IAMA) with the errors decreasing at pre-defined different rates.

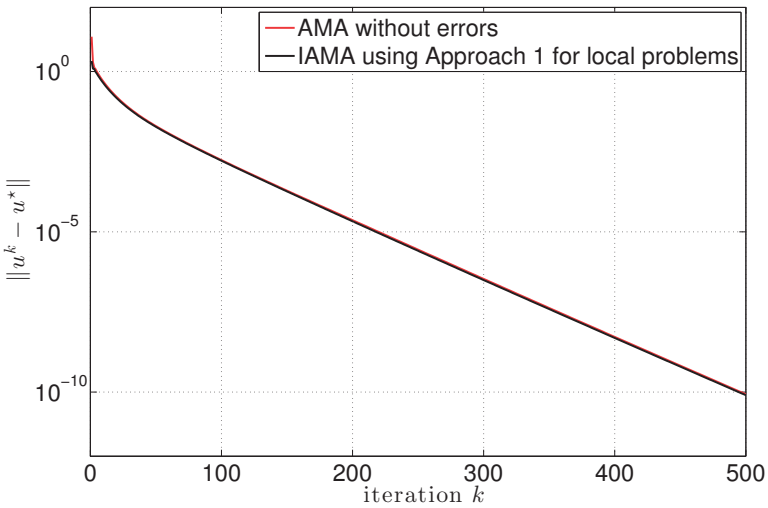


Figure 5.2 – The comparison of the performance of AMA and inexact AMA (IAMA) with the proximal-gradient method to solve local problems, where the number of iterations is chosen according to two approaches: **Approach 1** uses a bound on the minimal number, i.e., the termination condition proposed in (5.19); and **Approach 2** computes the exact minimal number, which requires the optimal solution of the local problem $z_i^{k,\star}$ at each iteration.

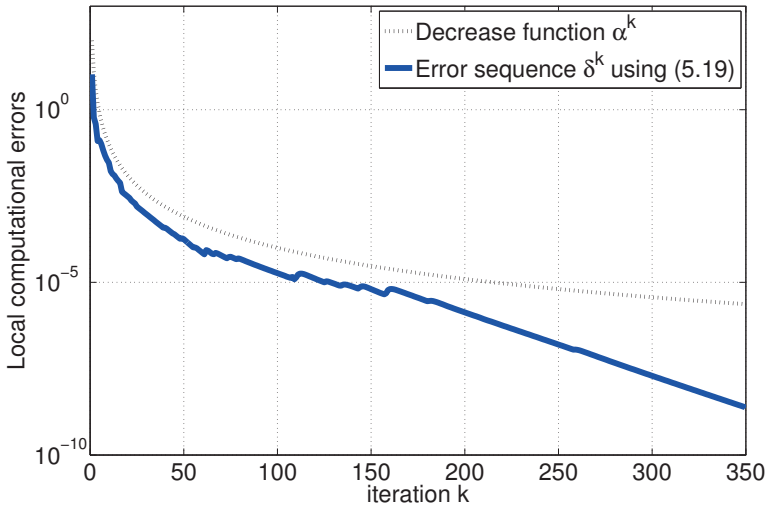


Figure 5.3 – The error sequence δ^k in inexact AMA using the proximal-gradient method for solving the local problems with the numbers of iterations satisfying (5.19).

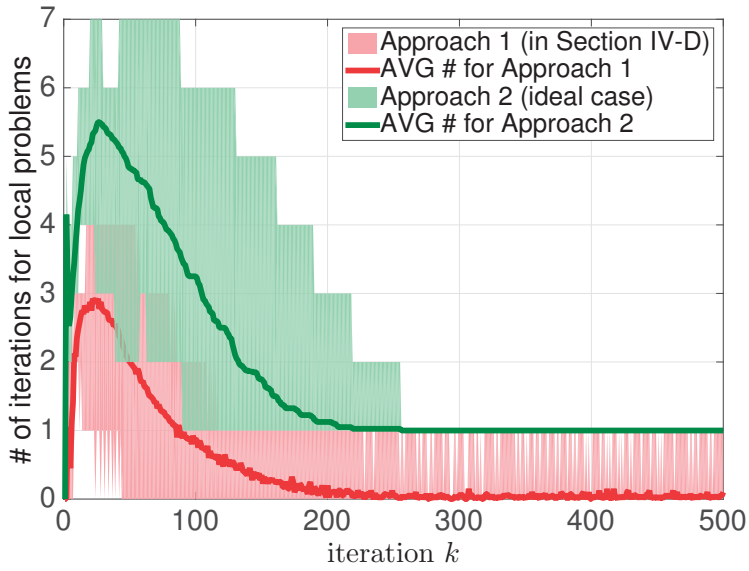


Figure 5.4 – The comparison of the numbers of iterations for Algorithm 15, using two approaches: **Approach 1** uses a bound on the minimal number, i.e. the termination condition proposed in (5.19); and **Approach 2** computes the exact minimal number, which requires the optimal solution of the local problem $z_i^{k,\star}$ at each iteration.

5.5 Conclusion

In this chapter, we proposed the inexact alternating minimization algorithm (inexact AMA), which allows inexact iterations in the algorithm, and its accelerated variant, called the inexact fast alternating minimization algorithm (inexact FAMA). We showed that inexact AMA and inexact FAMA are equivalent to the inexact proximal-gradient method and its accelerated variant applied to the dual problem. Based on this equivalence, we derived complexity upper-bounds on the number of iterations for the inexact algorithms. We apply inexact AMA and inexact FAMA to distributed optimization problems, with an emphasis on distributed MPC applications, and showed the convergence properties for this special case. By employing the complexity upper-bounds on the number of iterations, we provided sufficient conditions on the inexact iterations for the convergence of the algorithms. We further studied the special case of quadratic local objectives in the distributed optimization problems, which is a standard form of distributed MPC problem. For this special case, the algorithms allow local computational errors at each iteration. By exploiting a warm-starting strategy and the sufficient conditions on the errors for convergence, we proposed an approach to certify the number of iterations for solving local problems, which guarantees that the local computational errors satisfy the sufficient condition and the inexact distributed optimization algorithm converges to the optimal solution.

5.6 Appendix

5.6.1 Proof of Lemma 5.2

Proof: We first prove that there exists an upper bound on the series $b^k = \sum_{p=1}^{k} \frac{\alpha^{-p}}{p}$. Since $0 < \alpha < 1$, there always exists a positive integer k' such that $0 < \frac{\alpha^{-k'}}{k'} < \frac{\alpha^{-(k'+1)}}{k'+1}$. We can write the series b^k as

$$b^k = \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \sum_{p=k'}^{k} \frac{\alpha^{-p}}{p} \ .$$

Since k' satisfies $0 < \frac{\alpha^{-k'}}{k'} < \frac{\alpha^{-(k'+1)}}{k'+1}$ and $0 < \alpha < 1$, then we know that for any $t \geq k'$ the function $\frac{\alpha^{-t}}{t}$ is a non-decreasing function with respect to t. Due to the fact that for any non-decreasing function $f(t)$, the following inequality holds.

$$\sum_{p \in \mathbb{Z}: y \leq p \leq x} f(p) = \int_y^x f(\lfloor t \rfloor) dt + f(x) \leq \int_y^x f(t) dt + f(x)$$

where $\lfloor \cdot \rfloor$ denotes the floor operator, the series b^k can be upper-bounded by

$$b^k \leq \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \int_{k'}^{k} \frac{\alpha^{-t}}{t} dt + \frac{\alpha^{-k}}{k} \ .$$

We know that the integral of the function $\frac{\alpha^{-t}}{t}$ is equal to $\mathbf{E_i}(-x \log(\alpha))$, where $\mathbf{E_i}(\cdot)$ denotes the Exponential Integral Function, defined as $\mathbf{E_i}(x) := \int_{-x}^{\infty} \frac{e^{-t}}{t} dt$. By using the fact that $-\mathbf{E_i}(-x) = \mathbf{E_1}(x)$, where $\mathbf{E_1}(x) := \int_x^{\infty} \frac{e^{-t}}{t} dt$, and inequality (5.1.20) in [52], it follows that the Exponential Integral Function $\mathbf{E_i}(x)$ satisfies

$$-\log(1 + \frac{1}{x}) < e^x \cdot \mathbf{E_i}(-x) < -\frac{1}{2} \cdot \log(1 + \frac{2}{x}) \ .$$

Since $e^x > 0$, we can rewrite the inequality as

$$-e^{-x} \log(1 + \frac{1}{x}) < \mathbf{E_i}(-x) < -\frac{1}{2} e^{-x} \log(1 + \frac{2}{x}) \ .$$

Hence, the series b^k can be further upper-bounded by

$$\begin{aligned}
b^k &\leq \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \frac{\alpha^{-k}}{k} + \mathbf{E_i}(-k \log(\alpha)) - \mathbf{E_i}(-k' \log(\alpha)) \\
&< \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \frac{\alpha^{-k}}{k} - \frac{1}{2} e^{-k \log(\alpha)} \log(1 + \frac{2}{k \log(\alpha)}) \\
&\quad + e^{-k' \log(\alpha)} \log(1 + \frac{1}{k' \log(\alpha)}) \\
&= \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \frac{\alpha^{-k}}{k} - \frac{1}{2} \alpha^{-k} \log(1 + \frac{2}{k \log(\alpha)}) \\
&\quad + \alpha^{-k'} \log(1 + \frac{1}{k' \log(\alpha)}) \ .
\end{aligned}$$

We can now find the upper-bound for the series S^k as

$$s^k = \alpha^k \cdot b^k < \alpha^k \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p} + \frac{1}{k} - \frac{1}{2}\log(1 + \frac{2}{k\log(\alpha)}) \\ + \alpha^{k-k'} \log(1 + \frac{1}{k'\log(\alpha)}) \ .$$

Since $0 < \alpha < 1$ and the integer k' is a constant for a given α, the upper bound above converges to zero, as k goes to infinity. In addition, we know that the two terms $\alpha^k \sum_{p=1}^{k'} \frac{\alpha^{-p}}{p}$ and $\alpha^{k-k'} \log(1 + \frac{1}{k'\log(\alpha)})$ converge to zero linearly with the constant α. From Taylor series expansion, we know that the term $\frac{1}{2}\log(1 + \frac{2}{k\log(\alpha)})$ converges to zeros at the rate $O(\frac{1}{k})$. Note that since $0 < \alpha < 1$, the term $\frac{1}{2}\log(1 + \frac{2}{k\log(\alpha)})$ is always negative for all $k > 0$. To summarize, we know that the upper bound above converges to zero with the rate $O(\frac{1}{k})$. Therefore, we conclude that the series s^k converges to zero, as k goes to infinity. In addition, the convergence rate is $O(\frac{1}{k})$. ■

Quantization Design for Distributed Optimization

6

The majority of the text and content in Chapter 6 has appeared in [53] and [54].

6.1 Introduction

Distributed optimization methods for networked systems that have many coupled sub-systems and must act based on local information, are critical in many engineering problems, e.g. resource allocation, distributed estimation and distributed control problems. The algorithms are required to solve a global optimization problem in a distributed fashion subject to communication constraints.

In the framework of real-time distributed MPC, both local computation and communication resources can be quite limited, due to the short sampling time. In Chapter 5, we have developed distributed optimization algorithms with limited local computation power for solving distributed MPC problems. In this chapter, we consider a distributed optimization problem with communication constraints. In the distributed framework, each sub-problem in a network has a local cost function that involves both local and neighbouring variables, and local constraints on local variables. The first communication limitation is that the optimization problem needs to be solved in a distributed manner with only local communication, i.e. between neighbouring sub-systems. Related work proposing distributed optimization algorithms for different applications include, e.g. [55], [56] and [36]. The second communication limitation is that the communication data-rate between neighbouring sub-systems is limited. In order to meet the limited communication data-rate, the information exchanged between the neighbouring sub-systems needs to be quantized. The quantization process results in inexact iterations throughout the distributed optimization algorithm, which affects its convergence. Related work includes [57], [58], [59], [60] [61], [62] and [63], which study the effects of quantization on the performance of averaging or distributed optimization algorithms. In practice, many applications may suffer from these communication limitations, e.g. the implementation of predictive control algorithms for large-scale systems by utilizing distributed optimization methods. Systems suffering from a significant limitation in communication bandwidth include under-water vehicles and low-cost unmanned

aerial vehicles.

We propose two distributed optimization algorithms with progressive quantization design building on the work in [22] and [59]. The main challenge is that at each iteration only a fixed number of bits can be transmitted between neighbouring sub-systems. With a normal quantizer e.g. [64], the quantization process induces quantization errors that do not decrease with an increasing number of iterations. As a result, the distributed optimization algorithm may not converge or may only converge to a neighbourhood of the optimal solution. The main idea of the proposed techniques is to apply the inexact proximal gradient method to the distributed optimization problem and to employ the error conditions, which guarantee convergence to the global optimum, to design a progressive quantizer. Motivated by the linear convergence upper-bound of the inexact optimization algorithm, the range of the quantizer is set to reduce linearly at a rate smaller than one and larger than the rate of the algorithm, in order to refine the information exchanged in the network with each iteration and to achieve overall converge to the exact global optimum. The proposed quantization method is computationally cheap and consistent throughout the iterations as every node implements the same quantization procedure. In particular, the work makes the following main contributions:

- Constrained optimization problems: We consider distributed optimization problems with convex local constraints. To handle the constraints, two projection steps are required. One is applied before the information exchange, and the other afterwards. The reason for the second projection is that after the information exchange, the quantized value received by each agent can be an infeasible solution subject to the local constraints. The second projection step therefore guarantees that at each iteration every agent has a feasible solution for the computation of the gradient. We present conditions on the number of bits and the initial quantization intervals, which guarantee convergence of the algorithms. We show that after imposing the quantization scheme including the two projections, the algorithms preserve the linear convergence rate, and furthermore derive complexity upper-bounds on the number of iterations to achieve a given accuracy. In addition, we provide conditions on the minimum number of bits and the corresponding minimum initial quantization intervals that can be computed.

- Accelerated algorithm: We propose an accelerated variant of the distributed optimization algorithm with quantization refinement based on the inexact accelerated proximal-gradient method. With the acceleration step, the algorithm preserves the linear convergence rate, but the constant of the rate will be improved.

- Distributed optimal control example: We demonstrate the performance of the proposed method and the theoretical results for solving a distributed optimal control example.

6.2 Uniform quantizer

Let x be a real number. A uniform quantizer with a fixed number of bits n is defined as

$$Q(x) = \begin{cases} \bar{x} - \frac{l}{2} & \text{if } x \in \left(-\infty, \bar{x} - \frac{l}{2}\right) \\ \bar{x} + \text{sgn}(x - \bar{x}) \cdot \Delta \cdot \left\lfloor \frac{\|x - \bar{x}\|}{\Delta} \right\rfloor + \frac{\Delta}{2} & \text{if } x \in \left[\bar{x} - \frac{l}{2}, \bar{x} + \frac{l}{2}\right] , \\ \bar{x} + \frac{l}{2} & \text{if } x \in \left(\bar{x} - \frac{l}{2}, \infty\right) \end{cases} \tag{6.1}$$

where $\text{sgn}(\cdot)$ is the sign function. The parameter $\bar{x}$ denotes the mid-value of the uniform quantizer. The quantization step-size Δ is equal to $\Delta = \frac{l}{2^n}$, where l represents the size of the quantization interval. In this chapter, we assume that n is a fixed number, which means that the quantization interval is set to be $[\bar{x} - \frac{l}{2}, \bar{x} + \frac{l}{2}]$. From the definition in (6.1), we know that since the value x falls inside the quantization interval $x \in \left[\bar{x} - \frac{l}{2}, \bar{x} + \frac{l}{2}\right]$, the quantization error is bounded as

$$x - Q(x) \leq \frac{\Delta}{2} = \frac{l}{2^{n+1}} \ . \tag{6.2}$$

For the case that the input of the quantizer and the mid-value are not real numbers, but vectors with the same dimension n_x, the quantizer Q is composed of n_x independent scalar quantizers in (6.1) with the same quantization interval l and corresponding mid-value. In this chapter, we design a uniform quantizer denoted as $Q^k(\cdot)$ with changing quantization interval l^k and mid-value $\bar{x}^k$ at every iteration k of the optimization algorithm. A similar definition for a uniform quantizer can be found in [60] and [59].

The key challenge addressed in this work is to show that when applying a uniform quantizer with a fixed number of bits n to a distributed optimization algorithm, we can guarantee that at each iteration the value x^k falls inside the quantization interval and the quantization error is therefore bounded.

6.3 Distributed optimization with limited communication

In this section, we propose two distributed optimization algorithms with progressive quantization design based on the inexact PGM algorithm and its accelerated variant. The main challenge is that the communication in the distributed optimization algorithms is limited and the information exchanged in the network needs to be quantized. We propose a progressive quantizer with changing parameters, which satisfies the communication limitations, while ensuring that the errors induced by quantization satisfy the conditions for convergence.

6.3.1 Distributed optimization problem

In this chapter, we consider a distributed optimization problem on a network of M sub-systems (nodes). The sub-systems communicate according to a fixed undirected graph $G = (\mathcal{V}, \mathcal{E})$. We denote by $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}\}$ the set of the neighbours of sub-system i. The optimization variable of sub-system i and the global variable are denoted by x_i and $x = [x_1^T, \cdots, x_M^T]^T$, respectively. For each sub-system i, the

local variable has a convex local constraint $x_i \in \mathbb{C}_i \subseteq \mathbb{R}^{n_{m_i}}$. The constraint on the global variable x is denoted by $\mathbb{C} = \prod_{1\leq i\leq M} \mathbb{C}_i$. The dimension of the local variable x_i is denoted by m_i and the maximum dimension of the local variables is denoted by $\bar{m}$, i.e. $\bar{m} := \max_{1\leq i\leq M} m_i$. The concatenation of the variable of sub-system i and the variables of its neighbours is denoted by $x_{\mathcal{N}_i}$, and the corresponding constraint on $x_{\mathcal{N}_i}$ is denoted by $\mathbb{C}_{\mathcal{N}_i} = \prod_{j\in\mathcal{N}_i} \mathbb{C}_j$. We define the matrices $E_i \in \mathbb{R}^{\sum_{j\in\mathcal{N}_i} m_j \times \sum_{1\leq i\leq M} m_i}$ and $F_{ji} \in \mathbb{R}^{m_i \times \sum_{j\in\mathcal{N}_i} m_j}$ to be selection matrices with elements in $\{0, 1\}$. With these matrices, the variables can be represented as $x_{\mathcal{N}_i} = E_i x$ and $x_i = F_{ji} x_{\mathcal{N}_j}$, $j \in \mathcal{N}_i$, which implies the relation between the local variable x_i and the global variable x, i.e. $x_i = F_{ji} E_j x$, $j \in \mathcal{N}_i$. Note that E_i and F_{ji} are selection matrices, and therefore $\|E_i\| = \|F_{ji}\| = 1$. We solve a distributed optimization problem of the formulation in Problem 6.1:

Problem 6.1.

$$\begin{aligned} \min_{x} \quad & f(x) = \sum_{i=1}^{M} f_i(x_{\mathcal{N}_i}) \\ s.t. \quad & x_i \in \mathbb{C}_i \ , \ x_i = F_{ji} x_{\mathcal{N}_j} \ , \ j \in \mathcal{N}_i \ , \\ & x_{\mathcal{N}_i} = E_i x \ , \ i = 1, 2, \cdots, M \ . \end{aligned}$$

Assumption 6.1. *We assume that the global cost function $f(\cdot)$ is a twice differentiable and strongly convex function with respect to x with a convexity modulus σ_f.*

Assumption 6.2. *The local constraint set $\mathbb{C}_i$ is a closed, non-empty and convex set, for $i = 1, \cdots, M$.*

Assumption 6.3. *We assume that every local cost function $f_i(\cdot)$ has Lipschitz continuous gradient with Lipschitz constant L_i, and denote L_{max} as the maximum Lipschitz constant of the local functions, i.e. $L_{max} := \max_{1\leq i\leq M} L_i$.*

Remark 6.1. *If Assumption 6.3 holds, then the global cost function has a Lipschitz continuous gradient with a Lipschitz constant $L := \sum_{1\leq i\leq M} L_i$.*

6.3.2 Qualitative description of the algorithm

In this section, we provide a qualitative description of the distributed optimization algorithm with quantization refinement to introduce the main idea of the approach. We apply the inexact PGM algorithm to the distributed optimization problem in Problem 6.1, where the two objectives in Problem 2.1 are chosen as $\phi = \sum_{i=1}^{M} f_i(x_{\mathcal{N}_i})$ and $\psi = \sum_{i=1}^{M} I_{\mathbb{C}_i}(x_i)$, where $I_{\mathbb{C}_i}$ denotes the indicator function on the set $\mathbb{C}_i$, defined in (2.2). According to Assumption 6.1, ϕ should be a differentiable and strongly convex function with a convexity modulus σ_f, and Lipschitz continuous gradient with a Lipschitz constant L. Due to the fact that the indicator function defined in (2.2) is a convex function, ψ is a convex function. Hence, Assumption 2.1 for the inexact PGM algorithm is satisfied for the optimization problem with the two objectives ϕ and ψ. The complexity upper-bound in Proposition 2.6 holds. The parameter γ is equal to

$$\gamma = \frac{\sigma_f}{L} \ . \tag{6.3}$$

The communication in the network is limited: each sub-system in the network can only communicate with its neighbours, and at each iteration, only a fixed number of bits can be transmitted. Only considering the first limitation, the distributed optimization algorithm resulting from applying the inexact PGM algorithm to Problem 6.1 is represented by the blue boxes in Fig. 6.1. At iteration k, sub-system i carries out four main steps:

1. Send the local variable to the neighbours;
2. Compute the local gradient;
3. Send the local gradient to the neighbours;
4. Update the local variable and compute the projection of the updated local variable on the local constraint.

To handle the second limitation, we design two uniform quantizers (the pink boxes) for the two communication steps for each sub-system $Q^k_{\alpha,i}$ and $Q^k_{\beta,i}$ using a varying quantization interval and mid-value to refine the exchanged information at each iteration. Motivated by the second sufficient condition on the error sequences $\{e^k\}$ and $\{\epsilon^k\}$ for the convergence of the inexact PGM algorithm discussed in Section 2.3.1 (if the sequences $\{\|e^k\|\}$ and $\{\sqrt{\epsilon^k}\}$ decrease at a linear rate with the constant $(1-\gamma) < \rho < 1$, then $\|x^k - x^\star\|$ converges with the same rate), the quantization intervals are set to be two linearly decreasing functions in the number of iterations k, with a rate constant $(1-\gamma) < \rho < 1$. We know that if for every k, the values x^k_i and ∇f_i fall inside the quantization intervals, the quantization errors converge at the same linear rate with the constant ρ. In Section 6.3.3, we will show that by properly choosing the number of bits n and the initial intervals, it can be guaranteed that x^k_i and ∇f_i fall inside the quantization intervals at every iteration and the quantization errors decrease linearly.

We want to highlight the re-projection step (green box in Fig. 6.1), because it is the key step that allows us to solve distributed optimization problems with constraints. We set an extra re-projection step into the algorithm, because the quantized value $\hat{x}^k_{\mathcal{N}_i}$ can be an infeasible solution subject to the constraints $\mathbb{C}_{\mathcal{N}_i}$. Recall that we split Problem 6.1 as $\phi = \sum_{i=1}^M f_i(x_{\mathcal{N}_i})$ and $\psi = \sum_{i=1}^M I_{\mathbb{C}_i}(x_i)$. If the quantized value $\hat{x}^k_{\mathcal{N}_i}$ is an infeasible solution subject to $\mathbb{C}_{\mathcal{N}_i}$, then the computation error of the proximal operator with respect to ψ, defined in (2.23) is equal to infinity and it thereby violates the sufficient conditions on the error sequences for convergence. The re-projection step guarantees that at each iteration the gradient is computed based on a feasible solution, and the computation error of the proximal operator is finite. Using the convexity of the constraints, we can further show that the error caused by the re-projected point $\tilde{x}^k_{\mathcal{N}_i} = \mathrm{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}^k_{\mathcal{N}_i})$ is upper-bounded by the quantization error. To summarize, all the errors induced by the limited communication in the distributed algorithm are upper bounded by a linearly decreasing function with the constant ρ, which implies that the distributed algorithm with quantization converges to the global optimum and the linear convergence rate is preserved. These results will be shown in detail in Section 6.3.3.

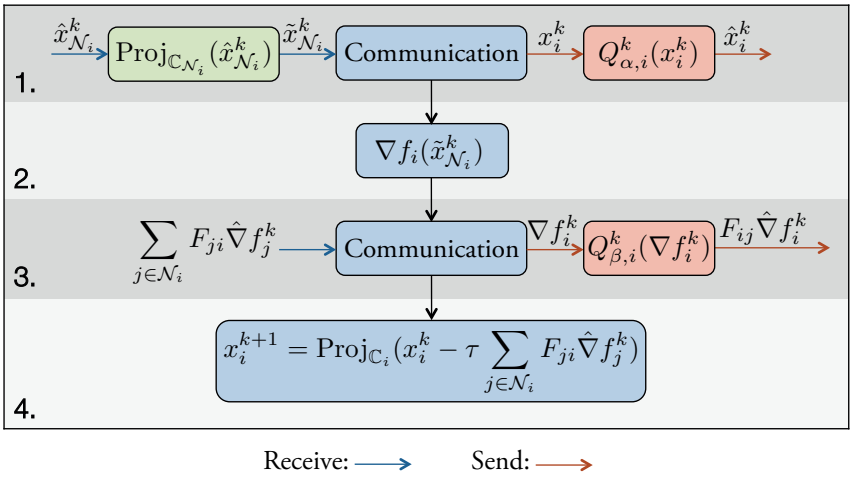


Figure 6.1 – Distributed algorithm with quantization refinement for subsystem i at iteration k.

6.3.3 Distributed algorithm with quantization refinement

In this section, we propose a distributed algorithm with a progressive quantization design in Algorithm 16. For every sub-system i, we define two uniform quantizers $Q^k_{\alpha,i}$ and $Q^k_{\beta,i}$ for transmitting x^k_i and ∇f^k_i at every iteration k. According to the definition introduced in Section 6.2, the quantizers are defined by a fixed number of bits n, changing quantization intervals $l^k_{\alpha,i}$ and $l^k_{\beta,i}$ and changing mid-values $\bar{x}^k_{\alpha,i}$ and $\bar{\nabla} f^k_{\beta,i}$. At iteration k, the quantization intervals are set to be $l^k_{\alpha,i} = C_\alpha \rho^k$ and $l^k_{\beta,i} = C_\beta \rho^k$, and the mid-values are set to be the previous quantized values $\bar{x}^k_{\alpha,i} = \hat{x}^{k-1}_i$ and $\bar{\nabla} f^k_{\beta,i} = \hat{\nabla} f^{k-1}_i$. The two parameters $C_\alpha = l^0_{\alpha,i}$ and $C_\beta = l^0_{\beta,i}$ denote the initial quantization intervals.

In the following, $\hat{\cdot}$ is used to denote a quantized value, e.g. $\hat{x}^k_i = Q^k_{\alpha,i}(x^k_i)$ and $\tilde{\cdot}$ is used to denote a re-projected value, e.g. $\tilde{x}^k_{\mathcal{N}_i} = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}^k_{\mathcal{N}_i})$. The quantization errors are denoted by $\alpha^k_i = \hat{x}^k_i - x^k_i$ and $\beta^k_i = \hat{\nabla} f^k_i - \nabla f^k_i$.

Remark 6.2. *Compared to the inexact PGM method in Algorithm 7, we modify the index of the sequence from k to $k+1$, such that in Section 7.3 the quantization errors have the same index as the quantized sequences at each iteration.*

In the following, we present four lemmas that link Algorithm 16 to the inexact PGM algorithm and prove that Algorithm 16 converges linearly to the global optimum despite the quantization errors. Lemma 6.1 states that due to the fact that the constraints are convex, the error between the re-projected point and the original point $\|\tilde{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\| \leq \|\hat{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\|$ is upper-bounded by the quantization error. Lemma 6.2 shows that the inexactness resulting from quantization in Algorithm 16 can be considered as the error in the gradient calculation $\{e^k\}$ and the error in the computation of the proximal minimization $\{\epsilon^k\}$ in Algorithm 7. Lemma 6.3 states that if at each iteration the values x^k_i and ∇f^k_i fall inside the quantization intervals, then the errors caused by quantization decrease linearly and the algorithm converges to the global optimum at the same rate. Lemma 6.4 provides conditions on the number of bits and the initial quantization intervals, which guarantee that

Algorithm 16 Distributed algorithm with quantization refinement

Require: Initialize $\hat{x}_i^{-1} = x_i^0 = 0$, $\hat{\nabla} f_i^{-1} = \nabla f_i(\text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(0))$, $(1-\gamma) < \rho < 1$ and $\tau < \frac{1}{L}$.

for $k = 0, 1, 2, \cdots$ **do**

For sub-system i, $i = 1, 2, \cdots, M$ do in parallel:

1: Update the parameters of quantizer $Q_{\alpha,i}^k$: $l_{\alpha,i}^k = C_\alpha \rho^k$ and $\bar{x}_{\alpha,i}^k = \hat{x}_i^{k-1}$

2: Quantize the local variable: $\hat{x}_i^k = Q_{\alpha,i}^k(x_i^k) = x_i^k + \alpha_i^k$

3: Send $\hat{x}_i^k$ to all the neighbours of sub-system i

4: Compute the projection of $\hat{x}_{\mathcal{N}_i}^k$: $\tilde{x}_{\mathcal{N}_i}^k = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}_{\mathcal{N}_i}^k)$

5: Compute $\nabla f_i^k = \nabla f_i(\tilde{x}_{\mathcal{N}_i}^k)$

6: Update the parameters of quantizer $Q_{\beta,i}^k$: $l_{\beta,i}^k = C_\beta \rho^k$ and $\bar{\nabla} f_{\beta,i}^k = \hat{\nabla} f_i^{k-1}$

7: Quantize the gradient: $\hat{\nabla} f_i^k = Q_{\beta,i}^k(\nabla f_i^k) = \nabla f_i^k + \beta_i^k$

8: Send $\hat{\nabla} f_i^k$ to all the neighbours of sub-system i

9: Update the local variable: $x_i^{k+1} = \text{Proj}_{\mathbb{C}_i}(x_i^k - \tau \sum_{j \in \mathcal{N}_i} F_{ji} \hat{\nabla} f_j^k)$

end for

x_i^k and ∇f_i^k fall inside the quantization intervals for each iteration. Once we prove the three lemmas, we are ready to present the main result in Theorem 6.2.

Lemma 6.1. *Let $\mathbb{C}$ be a convex subset of $\mathbb{R}^{n_v}$ and $\mu \in \mathbb{C}$. For any point $v \in \mathbb{R}^{n_v}$, the following holds:*

$$\|\mu - Proj_{\mathbb{C}}(v)\| \leq \|\mu - v\| \ . \tag{6.4}$$

Proof: Since $\mu \in \mathbb{C}$, we have $\text{Proj}_{\mathbb{C}}(\mu) = \mu$. Lemma 6.1 follows directly from Proposition 2.2.1 in [24]. ■

Lemma 6.2. *Consider the optimization problem with the two objectives $\phi = \sum_{i=1}^M f_i(x_{\mathcal{N}_i})$ and $\psi = \sum_{i=1}^M I_{\mathbb{C}_i}(x_i)$. Applying Algorithm 7 to the optimization problem results in Algorithm 16 with the error sequences defined as*

$$e^k = \sum_{i=1}^M E_i^T \nabla f_i(\tilde{x}_{\mathcal{N}_i}^k) + \sum_{i=1}^M E_i^T \beta_i^k - \sum_{i=1}^M E_i^T \nabla f_i(x_{\mathcal{N}_i}^k) \ ,$$

and $\epsilon^k = \frac{1}{2}\|x^k - \tilde{x}^k\|^2$. Furthermore, $\|e^k\|$ and $\sqrt{\epsilon^k}$ are upper-bounded by

$$\|e^k\| \leq \sum_{i=1}^M L_i \cdot \sum_{j \in \mathcal{N}_i} \|\alpha_j^k\| + \sum_{i=1}^M \|\beta_i^k\| \ , \tag{6.5}$$

and

$$\sqrt{\epsilon^k} \leq \frac{\sqrt{2}}{2} \sum_{i=1}^M \|\alpha_i^k\| \ . \tag{6.6}$$

Proof: By definition, the gradient computation error e^k in Algorithm 7 is equal to

$$\begin{aligned} e^k &= \hat{\nabla} f(\tilde{x}^k) - \nabla f(x^k) \\ &= \sum_{i=1}^{M} E_i^T \hat{\nabla} f_i(\tilde{x}^k_{\mathcal{N}_i}) - \sum_{i=1}^{M} E_i^T \nabla f_i(x^k_{\mathcal{N}_i}) \\ &= \sum_{i=1}^{M} E_i^T \nabla f_i(\tilde{x}^k_{\mathcal{N}_i}) + \sum_{i=1}^{M} E_i^T \beta_i^k - \sum_{i=1}^{M} E_i^T \nabla f_i(x^k_{\mathcal{N}_i}). \end{aligned}$$

Then,

$$\|e^k\| \leq \sum_{i=1}^{M} \|E_i^T\| \cdot L_i \cdot \|\tilde{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\| + \sum_{i=1}^{M} \|E_i^T\| \|\beta_i^k\| \ .$$

Since the matrix E_i is a selection matrix, $\|E_i^T\| = 1$. Since $x^k_{\mathcal{N}_i} \in \mathbb{C}_{\mathcal{N}_i}$ and $\tilde{x}^k_{\mathcal{N}_i} = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}^k_{\mathcal{N}_i})$, Lemma 6.1 implies $\|\tilde{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\| \leq \|\hat{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\|$. Hence, we have

$$\|e^k\| \leq \sum_{i=1}^{M} L_i \cdot \|\hat{x}^k_{\mathcal{N}_i} - x^k_{\mathcal{N}_i}\| + \sum_{i=1}^{M} \|\beta_i^k\| \leq \sum_{i=1}^{M} L_i \cdot \sum_{j \in \mathcal{N}_i} \|\alpha_j^k\| + \sum_{i=1}^{M} \|\beta_i^k\| \ .$$

By definition in (2.23) and the fact that $x^k \in \mathbb{C}$ and $\tilde{x}^k = \text{Proj}_{\mathbb{C}}(\hat{x}^k)$, we know $\epsilon^k = \frac{1}{2}\|x^k - \tilde{x}^k\|^2$. Lemma 6.1 again implies $\|x^k - \tilde{x}^k\| \leq \|x^k - \hat{x}^k\|$. Hence, we have

$$\sqrt{\epsilon^k} = \frac{\sqrt{2}}{2}\|x^k - \tilde{x}^k\| \leq \frac{\sqrt{2}}{2}\|x^k - \hat{x}^k\| \leq \frac{\sqrt{2}}{2} \sum_{i=1}^{M} \|\alpha_i^k\| \ .$$

■

Remark 6.3. *Lemma 6.2 shows that the errors $\|e^k\|$ and $\sqrt{\epsilon^k}$ are upper-bounded by functions of the quantization errors $\|\alpha_i^k\|$ and $\|\beta_i^k\|$. We want to emphasize that the quantization errors $\|\alpha_i^k\|$ and $\|\beta_i^k\|$ are not necessarily bounded by a linear function with the rate ρ. They are bounded only if the values x_i^k and ∇f_i fall inside the quantization intervals that are decreasing at a linear rate. Otherwise, the quantization errors $\|\alpha_i^k\|$ and $\|\beta_i^k\|$ can be arbitrarily large.*

From the discussion in Section 2.3.1, we know that if $\|e^k\|$ and $\sqrt{\epsilon^k}$ decrease linearly at a rate larger than $(1-\gamma)$, then $\|x^k - x^\star\|$ converges linearly at the same rate as $\|e^k\|$. Lemma 6.3 provides the first step towards achieving this goal. It shows that if the values of x_i^k and ∇f_i^k always fall inside the quantization interval, then the computational error of the gradient $\|e^k\|$ and the computational error of the proximal operator $\sqrt{\epsilon^k}$ as well as $\|x^k - x^\star\|$ decrease linearly with the constant ρ.

Lemma 6.3. *For any parameter ρ satisfying $(1-\gamma) < \rho < 1$ and a $k \geq 0$, if for all $0 \leq p \leq k$ the values of x_i^p and ∇f_i^p generated by Algorithm 16 fall inside of the quantization intervals of $Q^p_{\alpha,i}$ and $Q^p_{\beta,i}$, i.e. $\|x_i^p - \bar{x}^p_{\alpha,i}\|_\infty \leq \frac{l^p_{\alpha,i}}{2}$ and $\|\nabla f_i^p - \bar{\nabla} f^p_{\beta,i}\|_\infty \leq \frac{l^p_{\beta,i}}{2}$, then the error sequences $\|e^p\|$ and $\sqrt{\epsilon^p}$ satisfy*

$$\|e^p\| \leq C_1 \rho^p \ , \quad \sqrt{\epsilon^p} \leq C_2 \rho^p \ , \tag{6.7}$$

where $C_1 = \frac{M\sqrt{\bar{m}}(L_{max}dC_\alpha+\sqrt{d}C_\beta)}{2^{n+1}}$ *and* $C_2 = \frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$, *and* $\|x^{p+1}-x^\star\|$ *satisfies*

$$\|x^{p+1}-x^\star\| \leq \rho^{p+1}\left[\|x^0-x^\star\| + \frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)(1-\gamma)}\right] \ , \tag{6.8}$$

where $\bar{m} := \max_{1\leq i\leq M} m_i$, $L_{max} := \max_{1\leq i\leq M} L_i$ *and* d *denotes the degree of the graph of the distributed optimization problem.*

Proof: From the property of the uniform quantizer, we know that if x_i^p and ∇f_i^p fall inside of the quantization intervals of $Q_{\alpha,i}^p$ and $Q_{\beta,i}^p$, then the quantization errors α_i^p and β_i^p are upper-bounded by

$$\|\alpha_i^p\| \leq \sqrt{m_i}\cdot\|\alpha_i^p\|_\infty \leq \sqrt{m_i}\cdot\frac{l_{\alpha,i}^p}{2^{n+1}} \leq \sqrt{\bar{m}}\cdot\frac{l_{\alpha,i}^p}{2^{n+1}} \ ,$$
$$\|\beta_i^p\| \leq \sqrt{\sum_{j\in\mathcal{N}_i} m_i}\cdot\|\beta_i^p\|_\infty \leq \sqrt{\sum_{j\in\mathcal{N}_i} m_i}\cdot\frac{l_{\beta,i}^p}{2^{n+1}} \leq \sqrt{d\bar{m}}\cdot\frac{l_{\beta,i}^p}{2^{n+1}} \ .$$

From Lemma 6.2, we have

$$\|e^p\| \leq \sum_{i=1}^M L_i\cdot\sum_{j\in\mathcal{N}_i}\frac{\sqrt{\bar{m}}\cdot l_{\alpha,j}^p}{2^{n+1}} + \sum_{i=1}^M\frac{\sqrt{d\bar{m}}\cdot l_{\beta,i}^p}{2^{n+1}} \ ,$$

and

$$\sqrt{\epsilon^k} \leq \frac{\sqrt{2}}{2}\sum_{i=1}^M\frac{\sqrt{\bar{m}}l_{\alpha,i}^p}{2^{n+1}} \ .$$

Since the quantization intervals are set to $l_{\alpha,i}^p = C_\alpha\rho^p$ and $l_{\beta,i}^p = C_\beta\rho^p$, it implies that

$$\|e^p\| \leq \frac{ML_{max}d\sqrt{\bar{m}}\cdot C_\alpha\rho^p}{2^{n+1}} + \frac{M\sqrt{d\bar{m}}\cdot C_\beta\rho^p}{2^{n+1}} = C_1\rho^p \ ,$$

and

$$\sqrt{\epsilon^k} \leq \frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha\rho^p}{2^{n+1}} = C_2\rho^p \ ,$$

with $C_1 = \frac{M\sqrt{\bar{m}}(L_{max}dC_\alpha+\sqrt{d}C_\beta)}{2^{n+1}}$ and $C_2 = \frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$. Since $(1-\gamma) < \rho < 1$, Lemma 6.2 and Proposition 2.6 imply that for $0 \leq p \leq k$

$$\|x^{p+1}-x^\star\| \leq (1-\gamma)^{p+1}\|x^0-x^\star\| + \frac{(C_1+\sqrt{2L}C_2)}{L}\sum_{q=0}^p\rho^q(1-\gamma)^{p+1-q-1}$$
$$\leq \rho^{p+1}\left[\|x^0-x^\star\| + \frac{(C_1+\sqrt{2L}C_2)}{L(1-\gamma)}\sum_{q=0}^p(\frac{1-\gamma}{\rho})^{p+1-q}\right].$$

Since $0 < (1-\gamma) < \rho < 1$, by using the property of geometric series, we get that the expression above is equal to

$$= \rho^{p+1}\left[\|x^0-x^\star\| + \frac{(C_1+\sqrt{2L}C_2)}{L(1-\gamma)}\cdot\frac{1-(\frac{1-\gamma}{\rho})^{p+1}}{1-\frac{1-\gamma}{\rho}}\right]$$
$$\leq \rho^{p+1}\left[\|x^0-x^\star\| + \frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)(1-\gamma)}\right] \ .$$

Hence, inequality (6.8) is proven. ■

From Lemma 6.3, we know that the last missing piece is to show that the values x_i^k and ∇f_i^k fall inside the quantization interval at every iteration k. The following assumption presents conditions on the number of bits n and the initial quantization intervals C_α and C_β, which guarantee that for each iteration x_i^k and ∇f_i^k in Algorithm 16 fall inside the changing quantization intervals and the quantization errors decrease linearly with the constant ρ, which further implies that the Algorithm 16 converges to the global optimum linearly with the same rate ρ.

Assumption 6.4. *Consider the quantizers $Q_{\alpha,i}^k$ and $Q_{\beta,i}^k$ in Algorithm 16. We assume that the parameters of the quantizers, i.e., the number of bits n and the initial quantization intervals C_α and C_β satisfy*

$$a_1 + a_2\frac{C_\alpha}{2^{n+1}} + a_3\frac{C_\beta}{2^{n+1}} \leq \frac{C_\alpha}{2} \tag{6.9}$$

$$b_1 + b_2\frac{C_\alpha}{2^{n+1}} + b_3\frac{C_\beta}{2^{n+1}} \leq \frac{C_\beta}{2} \ , \tag{6.10}$$

with

$$\begin{aligned}
a_1 &= \frac{(\rho+1)\|x^0 - x^\star\|}{\rho} \ , \\
a_2 &= \frac{M\sqrt{\bar{m}}\rho(\rho+1)(dL_{\max}+\sqrt{L})}{L\rho(\rho+\gamma-1)(1-\gamma)} + \frac{M\sqrt{\bar{m}}L}{L\rho} \ , \\
a_3 &= \frac{M\sqrt{d\bar{m}}(\rho+1)}{L(\rho+\gamma-1)(1-\gamma)} \ , \ b_1 = \frac{L_{\max}(\rho+1)\|x^0 - x^\star\|}{\rho} \ , \\
b_2 &= \frac{L_{\max}M\sqrt{\bar{m}}\rho(\rho+1)(dL_{max}+\sqrt{L})}{L\rho(\rho+\gamma-1)(1-\gamma)} \\
&\quad + \frac{L_{max}d\sqrt{\bar{m}}L(\rho+1)}{L\rho} \ , \\
b_3 &= \frac{L_{\max}M\sqrt{d\bar{m}}\rho(\rho+1) + L\sqrt{d\bar{m}}(\rho+\gamma-1)(1-\gamma)}{L\rho(\rho+\gamma-1)(1-\gamma)} \ .
\end{aligned}$$

Remark 6.4. *The parameters of the quantizers n, C_α and C_β are all positive constants. Assumption 6.4 can always be satisfied by increasing n, C_α and C_β.*

Remark 6.5. *For a fixed n, inequalities (6.9) and (6.10) represent two polyhedral constraints on C_α and C_β. Therefore, the minimal C_α and C_β can be computed by solving a simple LP problem, i.e. minimizing $C_\alpha + C_\beta$ subject to $C_\alpha \geq 0$, $C_\beta \geq 0$, and inequalities (6.9) and (6.10). Since the minimal n is actually the minimal one guaranteeing that the LP problem has a feasible solution, the minimal n can be found by testing feasibility of the LP problem.*

Remark 6.6. *Keeping the parameters d, L and L_{max}, $\bar{m}$ constant, the number of bits n goes to ∞ according to Assumption 6.4, as the size of the graph M goes to ∞. A similar statement for quantized consensus problems with a fixed number of bits n can be found in [60].*

Remark 6.7. *Assumption 6.4 requires the knowledge of* $\|x^0 - x^\star\|$. *However, all theoretical properties are maintained if the quantity* $\|x^0 - x^\star\|$ *is replaced by an upper-bound of* $\|x^0 - x^\star\|$, *which can be obtained by initializing the algorithms with a feasible solution and estimating the size of the closed convex constraint* $\mathbb{C} := \mathbb{C}_1 \times \cdots \times \mathbb{C}_M$.

Lemma 6.4. *If Assumption 6.4 is satisfied and* $(1-\gamma) < \rho < 1$, *then the values of* x_i^k *and* ∇f_i^k *in Algorithm 16 fall inside of the quantization intervals of* $Q_{\alpha,i}^k$ *and* $Q_{\beta,i}^k$, *i.e.* $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ *and* $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$, *for all* $k \geq 0$.

Proof: We will prove Lemma 6.4 by induction.

- Base case: When $k = 0$, since C_α and C_β are positive numbers and $\hat{x}_i^{-1}$ and x_i^0 are initialized to zero, it holds that $\|x_i^0 - \bar{x}_{\alpha,i}^0\|_\infty = \|x_i^0 - \hat{x}_i^{-1}\|_\infty = 0 \leq \frac{l_{\alpha,i}^0}{2} = \frac{C_\alpha}{2}$ and $\|\nabla f_i^0 - \bar{\nabla} f_{\beta,i}^0\|_\infty = \|\nabla f_i^0 - \hat{\nabla} f_i^{-1}\|_\infty = \|\nabla f_i(\tilde{x}_{\mathcal{N}_i}^0) - \nabla f_i(\text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(0))\| = 0 \leq \frac{l_{\beta,i}^0}{2} = \frac{C_\beta}{2}$.

- Induction step: Let $g \geq 0$ be given and suppose that $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ and $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$ for $0 \leq k \leq g$. We will prove that

$$\|x_i^{g+1} - \bar{x}_{\alpha,i}^{g+1}\|_\infty \leq \frac{l_{\alpha,i}^{g+1}}{2} \tag{6.11}$$

and

$$\|\nabla f_i^{g+1} - \bar{\nabla} f_{\beta,i}^{g+1}\|_\infty \leq \frac{l_{\beta,i}^{g+1}}{2} \tag{6.12}$$

for $i = 1, \cdots, M$. We first show (6.11). From Algorithm 16, we know

$$\begin{aligned}
\|x_i^{g+1} - \bar{x}_{\alpha,i}^{g+1}\|_\infty &= \|x_i^{g+1} - \hat{x}_i^g\|_\infty \\
&\leq \|x^{g+1} - \hat{x}^g\|_\infty \\
&= \|x^{g+1} - x^g - \sum_{i=1}^M E_i^T F_{ii}^T \alpha_i^g\|_\infty \\
&\leq \|x^{g+1} - x^g\|_\infty + \|\sum_{i=1}^M E_i^T F_{ii}^T \alpha_i^g\|_\infty \\
&\leq \|x^{g+1} - x^\star\|_\infty + \|x^g - x^\star\|_\infty + \|\sum_{i=1}^M E_i^T F_{ii}^T \alpha_i^g\|_\infty \ .
\end{aligned}$$

Since E_i and F_{ii} are selection matrices, then $\|E_i\| = \|F_{ii}\| = 1$. The term above is upper-bounded by

$$\leq \|x^{g+1} - x^\star\|_2 + \|x^g - x^\star\|_2 + \sum_{i=1}^M \|\alpha_i^g\|_2 \ .$$

By the assumption of the induction, we know $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ and $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$ for $0 \leq k \leq g$. Then, using Lemma 6.3, we obtain that the

term above is upper-bounded by

$$\begin{aligned}&\leq \rho^{g+1}\left[\|x^0 - x^\star\| + \frac{(C_1 + \sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)(1-\gamma)}\right] \\ &\quad + \rho^g\left[\|x^0 - x^\star\| + \frac{(C_1 + \sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)(1-\gamma)}\right] + \frac{M\sqrt{\bar{m}}C_\alpha\rho^g}{2^{n+1}} \ .\end{aligned}$$

By substituting $C_1 = \frac{M\sqrt{\bar{m}}(L_{max}dC_\alpha + \sqrt{d}C_\beta)}{2^{n+1}}$ and $C_2 = \frac{\sqrt{2}}{2} \cdot \frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$ and using the parameters defined in Assumption 6.4, it follows that the expression above is equal to

$$= \rho^{g+1}\left[a_1 + a_2\frac{C_\alpha}{2^{n+1}} + a_3 \cdot \frac{C_\beta}{2^{n+1}}\right] \ .$$

By inequality (6.9) in Assumption 6.4, the term above is bounded by $\frac{C_\alpha}{2}\rho^{g+1}$. Thus, inequality (6.11) holds. In the following, we prove that inequality (6.12) is true.

$$\begin{aligned}\|\nabla f_i^{g+1} - \bar{\nabla} f_{\beta,i}^{g+1}\|_\infty &= \|\nabla f_i^{g+1} - \hat{\nabla} f_i^g\|_\infty \\ &= \|\nabla f_i(\tilde{x}_{\mathcal{N}_i}^{g+1}) - \nabla f_i(\tilde{x}_{\mathcal{N}_i}^g) + \beta_i^g\|_\infty \\ &\leq \|\nabla f_i(\tilde{x}_{\mathcal{N}_i}^{g+1}) - \nabla f_i(\tilde{x}_{\mathcal{N}_i}^g)\|_\infty + \|\beta_i^g\|_\infty \\ &\leq \|\nabla f_i(\tilde{x}_{\mathcal{N}_i}^{g+1}) - \nabla f_i(\tilde{x}_{\mathcal{N}_i}^g)\|_2 + \|\beta_i^g\|_2 \\ &\leq L_i\|\tilde{x}_{\mathcal{N}_i}^{g+1} - \tilde{x}_{\mathcal{N}_i}^g\| + \|\beta_i^g\| \\ &\leq L_i\|x_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^g\| + L_i\|\tilde{x}_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^{g+1}\| \\ &\quad + L_i\|\tilde{x}_{\mathcal{N}_i}^g - x_{\mathcal{N}_i}^g\| + \|\beta_i^g\|\end{aligned}$$

Since $x_{\mathcal{N}_i}^{g+1}, x_{\mathcal{N}_i}^g \in \mathbb{C}_{\mathcal{N}_i}$, $\tilde{x}_{\mathcal{N}_i}^{g+1} = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}_{\mathcal{N}_i}^{g+1})$ and $\tilde{x}_{\mathcal{N}_i}^g = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}_{\mathcal{N}_i}^g)$, Lemma 6.1 implies $\|\tilde{x}_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^{g+1}\| \leq \|\hat{x}_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^{g+1}\|$ and $\|\tilde{x}_{\mathcal{N}_i}^g - x_{\mathcal{N}_i}^g\| \leq \|\hat{x}_{\mathcal{N}_i}^g - x_{\mathcal{N}_i}^g\|$. Hence, the term above is upper-bounded by

$$\begin{aligned}&\leq L_i\|x_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^g\| + L_i\|\hat{x}_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^{g+1}\| + L_i\|\hat{x}_{\mathcal{N}_i}^g - x_{\mathcal{N}_i}^g\| + \|\beta_i^g\| \\ &\leq L_i\|x_{\mathcal{N}_i}^{g+1} - x_{\mathcal{N}_i}^g\| + L_i\sum_{j\in\mathcal{N}_i}(\|\alpha_j^{g+1}\| + \|\alpha_j^g\|) + \|\beta_i^g\| \\ &\leq L_i\|x^{g+1} - x^g\| + L_i\sum_{j\in\mathcal{N}_i}(\|\alpha_j^{g+1}\| + \|\alpha_j^g\|) + \|\beta_i^g\| \\ &\leq L_{\max}(\|x^{g+1} - x^\star\| + \|x^g - x^\star\|) + L_{\max}\sum_{j\in\mathcal{N}_i}(\|\alpha_j^{g+1}\| + \|\alpha_j^g\|) + \|\beta_i^g\| \ .\end{aligned}$$

Again by the assumption of the induction, we know $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ and $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$ for $0 \leq k \leq g$. Then, Lemma 6.3 implies that the term

above is upper-bounded by

$$\begin{aligned}
&\leq L_{\max}\rho^{g+1}\left(\|x^0-x^\star\|+\frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)}\right)\\
&\quad+L_{\max}\rho^{g}\left(\|x^0-x^\star\|+\frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)}\right)\\
&\quad+\frac{L_{\max}\sqrt{\bar{m}}\sum_{j\in\mathcal{N}_i}(l^{g+1}_{\alpha,j}+l^{g}_{\alpha,j})}{2^{n+1}}+\frac{\sqrt{d\bar{m}}l^g_{\beta,i}}{2^{n+1}}\\
&\leq L_{\max}\rho^{g+1}\left(\|x^0-x^\star\|+\frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)}\right)\\
&\quad+L_{\max}\rho^{g}\left(\|x^0-x^\star\|+\frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)}\right)\\
&\quad+\frac{L_{\max}\sqrt{d\bar{m}}C_\alpha(\rho^{g+1}+\rho^g)}{2^{n+1}}+\frac{\sqrt{d\bar{m}}C_\beta\rho^g}{2^{n+1}}\ .
\end{aligned}$$

By substituting $C_1=\frac{M\sqrt{\bar{m}}(L_{max}dC_\alpha+\sqrt{d}C_\beta)}{2^{n+1}}$ and $C_2=\frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$ and using the parameters defined in Assumption 6.4, it follows that the expression above is equal to

$$=\rho^{g+1}\cdot\left[b_1+b_2\cdot\frac{C_\alpha}{2^{n+1}}+b_3\cdot\frac{C_\beta}{2^{n+1}}\right]\ .$$

By inequality (6.10) in Assumption 6.4, the term above is bounded by $\frac{C_\beta}{2}\rho^{g+1}=\frac{l^{g+1}_{\beta,i}}{2}$. Thus, inequality (6.12) holds.

We conclude that by the principle of induction, the values of x_i^k and ∇f_i^k in Algorithm 16 fall inside of the quantization intervals of $Q^k_{\alpha,i}$ and $Q^k_{\beta,i}$, i.e. $\|x_i^k-\bar{x}^k_{\alpha,i}\|_\infty\leq\frac{l^k_{\alpha,i}}{2}$ and $\|\nabla f_i^k-\bar{\nabla}f^k_{\beta,i}\|_\infty\leq\frac{l^k_{\beta,i}}{2}$ for all $k\geq 0$. ■

After showing Lemma 6.2, Lemma 6.3 and Lemma 6.4, we are ready to present the main theorem.

Theorem 6.2. *If Assumptions 6.1, 6.3 and 6.4 hold and $(1-\gamma)<\rho<1$, then for $k\geq 0$ the sequence $\{x^k\}$ generated by Algorithm 16 converges to the optimum linearly with the constant ρ and satisfies*

$$\|x^{k+1}-x^\star\|\leq\rho^{k+1}\left[\|x^0-x^\star\|+\frac{(C_1+\sqrt{2L}C_2)\rho}{L(\rho+\gamma-1)(1-\gamma)}\right]\ , \tag{6.13}$$

with $C_1=\frac{M\sqrt{\bar{m}}(L_{max}dC_\alpha+\sqrt{d}C_\beta)}{2^{n+1}}$ and $C_2=\frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$.

Proof: Since Assumption 6.1, 6.3 and 6.4 hold, Lemma 6.4 states that for each iteration the values x_i^k and ∇f_i^k in Algorithm 16 fall inside of the quantization intervals of $Q^k_{\alpha,i}$ and $Q^k_{\beta,i}$. Then from Lemma 6.3, we know that the error sequences $\|e^k\|$ and $\sqrt{\epsilon^k}$ satisfy $\|e^k\|\leq C_1\rho^k$ and $\sqrt{\epsilon^k}\leq C_2\rho^k$, and by Lemma 6.2 the sequence x^k generated by Algorithm 16 satisfies inequality (6.13). ■

Remark 6.8. *In Assumption 6.1, the function f is assumed to be twice differentiable and strongly convex. These conditions guarantee that the parameter γ is a positive value smaller than* 1*. In particular, the convexity modulus σ_f and the Lipschitz constant of the gradient L are the lower and upper-bound of the Hessian of f, respectively. See [65] for details.*

Recalling the complexity bound in Proposition 2.6, we know that for the case without errors the algorithm converges linearly with the constant $1-\gamma$. After imposing quantization on the algorithm, it still converges to the global optimum linearly but with a larger constant $\rho > 1-\gamma$. We conclude that with the proposed quantization design, the linear convergence of the algorithm is preserved, but the constant of the convergence rate has to be enlarged in order to compensate for the deficiencies from limited communication.

6.3.4 Accelerated distributed algorithm with quantization refinement

In this section, we propose an accelerated variant of the distributed algorithm with quantization refinement in Algorithm 17 based on the inexact accelerated proximal gradient method in Algorithm 8. Compared to Algorithm 16, Algorithm 17 has an extra acceleration Step 5 $\tilde{y}^k_{\mathcal{N}_i} = \tilde{x}^k_{\mathcal{N}_i} + \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}(\tilde{x}^k_{\mathcal{N}_i} - \tilde{x}^{k-1}_{\mathcal{N}_i})$, and at each iteration the gradient ∇f_i^k is computed based on $\tilde{y}^k_{\mathcal{N}_i}$. The acceleration step improves the constant of the linear convergence rate of the algorithms from $1-\gamma$ to $\sqrt{1-\sqrt{\gamma}}$, and changes the condition on the quantization parameter ρ to $\sqrt{1-\sqrt{\gamma}} < \rho < 1$.

Algorithm 17 Accelerated distributed algorithm with quantization refinement

Require: Initialize $\hat{x}_i^{-1} = x_i^{-1} = x_i^0 = 0$, $\tilde{x}^{-1}_{\mathcal{N}_i} = 0$, $\hat{\nabla} f_i^{-1} = \nabla f_i(\mathrm{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(0))$, $\sqrt{1-\sqrt{\gamma}} < \rho < 1$ and $\tau < \frac{1}{L}$.

for $k = 0, 1, 2, \cdots$ **do**

For sub-system i, $i = 1, 2, \cdots, M$ do in parallel:

1: Update the parameters of quantizer $Q^k_{\alpha,i}$: $l^k_{\alpha,i} = C_\alpha \rho^k$ and $\bar{x}^k_{\alpha,i} = \hat{x}_i^{k-1}$

2: Quantize the local variable: $\hat{x}_i^k = Q^k_{\alpha,i}(x_i^k) = x_i^k + \alpha_i^k$

3: Send $\hat{x}_i^k$ to all the neighbours of sub-system i

4: Compute the projection of $\hat{x}^k_{\mathcal{N}_i}$: $\tilde{x}^k_{\mathcal{N}_i} = \mathrm{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}^k_{\mathcal{N}_i})$

5: Accelerating update: $\tilde{y}^k_{\mathcal{N}_i} = \tilde{x}^k_{\mathcal{N}_i} + \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}(\tilde{x}^k_{\mathcal{N}_i} - \tilde{x}^{k-1}_{\mathcal{N}_i})$ and $y_i^k = x_i^k + \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}(x_i^k - x_i^{k-1})$

6: Compute $\nabla f_i^k = \nabla f_i(\tilde{y}^k_{\mathcal{N}_i})$

7: Update the parameters of quantizer $Q^k_{\beta,i}$: $l^k_{\beta,i} = C_\beta \rho^k$ and $\bar{\nabla} f^k_{\beta,i} = \hat{\nabla} f_i^{k-1}$

8: Quantize the gradient: $\hat{\nabla} f_i^k = Q^k_{\beta,i}(\nabla f_i^k) = \nabla f_i^k + \beta_i^k$

9: Send $\hat{\nabla} f_i^k$ to all the neighbours of sub-system i

10: Update the local variable: $x_i^{k+1} = \mathrm{Proj}_{\mathbb{C}_i}(y_i^k - \tau \sum_{j \in \mathcal{N}_i} F_{ji} \hat{\nabla} f_j^k)$

end for

Lemma 6.5. *Consider the optimization problem with the two objectives $\phi = \sum_{i=1}^M f_i(x_{\mathcal{N}_i})$ and $\psi = \sum_{i=1}^M I_{\mathbb{C}_i}(x_i)$. Applying Algorithm 8 to the optimization problem results in*

Algorithm 17 with the error sequences defined as

$$e^k = \sum_{i=1}^{M} E_i^T \nabla f_i(\tilde{y}_{\mathcal{N}_i}^k) + \sum_{i=1}^{M} E_i^T \beta_i^k - \sum_{i=1}^{M} E_i^T \nabla f_i(y_{\mathcal{N}_i}^k) \ ,$$

and $\epsilon^k = \frac{1}{2}\|x^k - \tilde{x}^k\|^2$*, and upper-bounded by*

$$\|e^k\| \leq \sum_{i=1}^{M} L_i \cdot \sum_{j \in \mathcal{N}_i} \left(\frac{2}{1+\sqrt{\gamma}}\|\alpha_j^k\| + \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}\|\alpha_j^{k-1}\|\right) + \sum_{i=1}^{M} \|\beta_i^k\| \ , \tag{6.14}$$

and

$$\sqrt{\epsilon^k} \leq \frac{\sqrt{2}}{2} \sum_{i=1}^{M} \|\alpha_i^k\| \ . \tag{6.15}$$

Proof: The proof follows the same flow of the proof of Lemma 6.2. The only difference is that at each iteration the gradient ∇f_i^k is computed based on $\tilde{y}_{\mathcal{N}_i}^k$, which is a linear combination of $\tilde{x}_{\mathcal{N}_i}^k$ and $\tilde{x}_{\mathcal{N}_i}^{k-1}$. Hence, the upper-bound on the computational error of the gradient $\|e^k\|$ is a function of the linear combination of $\|\alpha_i^{k-1}\|$, $\|\alpha_i^k\|$ and $\|\beta_i^k\|$. ■

Lemma 6.6. *For any parameter* ρ *satisfying* $\sqrt{1-\sqrt{\gamma}} < \rho < 1$ *and* $k \geq 0$*, if for all* $0 \leq p \leq k$ *the values of* x_i^p *and* ∇f_i^p *generated by Algorithm 17 fall inside of the quantization intervals of* $Q_{\alpha,i}^p$ *and* $Q_{\beta,i}^p$*, i.e.* $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ *and* $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$*, then the error sequences satisfy*

$$\|e^p\| \leq C_3 \rho^p \ , \quad \sqrt{\epsilon^p} \leq C_4 \rho^p \ , \tag{6.16}$$

where $C_3 = \frac{M\sqrt{\bar{m}}(3L_{max}dC_\alpha + \rho\sqrt{d}C_\beta)}{\rho \cdot 2^{n+1}}$ *and* $C_4 = \frac{\sqrt{2}}{2} \cdot \frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$*, and* $\|x^{p+1} - x^\star\|$ *satisfies*

$$\|x^{p+1} - x^\star\| \leq \rho^{p+1}\left[\frac{2\sqrt{\Phi(x^0) - \Phi(x^\star)}}{\sqrt{\sigma_\phi}} + \frac{(2C_3 + 2\sqrt{2L}C_4 + \sqrt{2\sigma_\phi}C_4)\rho}{\sigma_\phi(\rho - \sqrt{1-\sqrt{\gamma}}) \cdot \sqrt{1-\sqrt{\gamma}}}\right] \ . \tag{6.17}$$

Proof: The proof follows the same flow of the proof of Lemma 6.3 by replacing the upper-bounds on $\|e^k\|$ and $\sqrt{\epsilon^k}$ in Lemma 6.2 and the upper-bound on $\|x^{p+1} - x^\star\|$ in Proposition 2.6 by the ones in Lemma 6.5 and Proposition 2.9. In addition, the proof requires the fact that $\sqrt{1-\sqrt{\gamma}} < \rho < 1$ and $1 < 1 + \sqrt{\gamma} < 2$. ■

Assumption 6.5. *We assume that the number of bits* n *and the initial quantization intervals* C_α *and* C_β *satisfy*

$$a_4 + a_5\frac{C_\alpha}{2^{n+1}} + a_6\frac{C_\beta}{2^{n+1}} \leq \frac{C_\alpha}{2} \tag{6.18}$$

$$b_4 + b_5\frac{C_\alpha}{2^{n+1}} + b_6\frac{C_\beta}{2^{n+1}} \leq \frac{C_\beta}{2} \ , \tag{6.19}$$

with

$$
\begin{aligned}
a_4 &= \frac{2(\rho+1)\sqrt{\Phi(x^0)-\Phi(x^\star)}}{\rho\sqrt{\sigma_\phi}} \ , \\
a_6 &= \frac{2M\sqrt{d\bar{m}}(\rho+1)}{\sigma_\phi(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}} \ , \\
a_5 &= \frac{6M\sqrt{\bar{m}}(\rho+1)dL_{\max}+M\sqrt{\bar{m}}\rho(\rho+1)(2\sqrt{L}+\sqrt{\sigma_\phi})}{\sigma_\phi\rho(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}} \\
&\quad + \frac{\sigma_\phi M\sqrt{\bar{m}}(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}{\sigma_\phi\rho(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}} \ , \\
b_4 &= \frac{2L_{\max}(2\rho^2+3\rho+1)\sqrt{\Phi(x^0)-\Phi(x^\star)}}{\rho^2\sqrt{\sigma_\phi}} \ , \\
b_5 &= \frac{L_{max}\sqrt{\bar{m}}(2\rho^2+3\rho+1)}{\rho^2}\left[d+\frac{6MdL_{max}+M\rho(2\sqrt{L}+\sqrt{\sigma_\phi})}{\sigma_\phi(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}\right] \ , \\
b_6 &= \frac{2L_{\max}M\sqrt{d\bar{m}}(2\rho^2+3\rho+1)+\sigma_\phi\sqrt{d\bar{m}}(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}{\sigma_\phi\rho(\rho-\sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}
\end{aligned}
$$

Lemma 6.7. *If Assumption 6.5 is satisfied and* $\sqrt{1-\sqrt{\gamma}} < \rho < 1$*, then for any* $k \geq 0$ *the values of* x_i^k *and* ∇f_i^k *in Algorithm 17 fall inside of the quantization intervals of* $Q_{\alpha,i}^k$ *and* $Q_{\beta,i}^k$*, i.e.* $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ *and* $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$.

Proof: The proof is similar to the proof of Lemma 6.4. The difference is that at each iteration the gradient ∇f_i^k is computed based on $\tilde{y}_{\mathcal{N}_i}^k$, which is a linear combination of $\tilde{x}_{\mathcal{N}_i}^k$ and $\tilde{x}_{\mathcal{N}_i}^{k-1}$. We therefore only show a brief proof for the second step, i.e. the inequality $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$ for any $k \geq 0$ by induction.

- Base case: When $k = 0$, since C_β is positive a number, $\tilde{x}_{\mathcal{N}_i}^{-1}$ and x_i^0 are initialized to zero and $\hat{\nabla} f_i^{-1} = \nabla f_i(\mathrm{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(0))$, it holds that $\|\nabla f_i^0 - \bar{\nabla} f_{\beta,i}^0\|_\infty = \|\nabla f_i^0 - \hat{\nabla} f_i^{-1}\|_\infty = \|\nabla f_i(\tilde{y}_{\mathcal{N}_i}^0) - \nabla f_i(\mathrm{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(0))\| = 0 \leq \frac{l_{\beta,i}^0}{2} = \frac{C_\beta}{2}$.

- Induction step: Let $g \geq 0$ be given and suppose that $\|x_i^k - \bar{x}_{\alpha,i}^k\|_\infty \leq \frac{l_{\alpha,i}^k}{2}$ and $\|\nabla f_i^k - \bar{\nabla} f_{\beta,i}^k\|_\infty \leq \frac{l_{\beta,i}^k}{2}$ for $0 \leq k \leq g$. We will prove

$$\|\nabla f_i^{g+1} - \bar{\nabla} f_{\beta,i}^{g+1}\|_\infty \leq \frac{l_{\beta,i}^{g+1}}{2} \ . \tag{6.21}$$

From the algorithm, we know

$$
\begin{aligned}
\|\nabla f_i^{g+1} - \bar{\nabla} f_{\beta,i}^{g+1}\|_\infty &= \|\nabla f_i^{g+1} - \hat{\nabla} f_i^g\|_\infty \\
&= \|\nabla f_i(\tilde{y}_{\mathcal{N}_i}^{g+1}) - \nabla f_i(\tilde{y}_{\mathcal{N}_i}^g) + \beta_i^g\|_\infty \\
&\leq L_i\|y_{\mathcal{N}_i}^{g+1} - y_{\mathcal{N}_i}^g\| + L_i\|\hat{y}_{\mathcal{N}_i}^{g+1} - y_{\mathcal{N}_i}^{g+1}\| \\
&\quad + L_i\|\hat{y}_{\mathcal{N}_i}^g - y_{\mathcal{N}_i}^g\| + \|\beta_i^g\| \ .
\end{aligned}
$$

By substituting $\hat{y}^g_{\mathcal{N}_i} = \frac{2}{1+\sqrt{\gamma}}\hat{x}^g_{\mathcal{N}_i} - \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}\hat{x}^{g-1}_{\mathcal{N}_i}$, $y^g_{\mathcal{N}_i} = \frac{2}{1+\sqrt{\gamma}}x^g_{\mathcal{N}_i} - \frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}x^{g-1}_{\mathcal{N}_i}$ and $L_{max} := \max_{1\leq i\leq M} L_i$, and using the fact that $\frac{2}{1+\sqrt{\gamma}} \leq 2$ and $\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}} \leq 1$, the expression above is upper-bounded by

$$\begin{aligned}\leq & L_{\max}(2\|x^{g+1} - x^\star\| + 3\|x^g - x^\star\| + \|x^{g-1} - x^\star\|) \\ & + L_{\max}\sum_{j\in\mathcal{N}_i}(2\|\alpha_j^{g+1}\| + 3\|\alpha_j^g\| + \|\alpha_j^{g-1}\|) + \|\beta_i^g\| \ .\end{aligned}$$

By the assumption of the induction and Lemma 6.6, we obtain that the above is upper-bounded by

$$\begin{aligned}\leq & L_{\max}(2\rho^{g+1} + 3\rho^g + \rho^{g-1})\left[\frac{2\sqrt{\Phi(x^0) - \Phi(x^\star)}}{\sqrt{\sigma_\phi}}\right. \\ & \left. + \frac{(2C_3 + 2\sqrt{2L}C_4 + \sqrt{2\sigma_\phi}C_4)\rho}{\sigma_\phi(\rho - \sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}\right] \\ & + \frac{L_{\max}\sqrt{\bar{m}}d(2l^{g+1}_{\alpha,j} + 3l^g_{\alpha,j} + l^{g-1}_{\alpha,j})}{2^{n+1}} + \frac{\sqrt{d\bar{m}}l^g_{\beta,i}}{2^{n+1}} \ .\end{aligned}$$

By substituting $C_3 = \frac{M\sqrt{\bar{m}}(3L_{max}dC_\alpha + \rho\sqrt{d}C_\beta)}{\rho\cdot 2^{n+1}}$ and $C_4 = \frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$ and using the parameters defined in Assumption 6.5, the expression becomes

$$= \rho^{g+1}\cdot\left[b_4 + b_5\cdot\frac{C_\alpha}{2^{n+1}} + b_6\cdot\frac{C_\beta}{2^{n+1}}\right] \ .$$

By inequality (6.19) in Assumption 6.5, the term above is bounded by $\frac{C_\beta}{2}\rho^{g+1} = \frac{l^{g+1}_{\beta,i}}{2}$. Thus, the inequality $\|\nabla f_i^{g+1} - \bar{\nabla} f^{g+1}_{\beta,i}\|_\infty \leq \frac{l^{g+1}_{\beta,i}}{2}$ holds. The proof of the induction step is complete.

By the principle of induction, we conclude that the inequality$\|\nabla f_i^k - \bar{\nabla} f^k_{\beta,i}\|_\infty \leq \frac{l^k_{\beta,i}}{2}$ holds for any $k \geq 0$. ■

Theorem 6.3. *If Assumptions 6.1, 6.3 and 6.5 hold and $\sqrt{1-\sqrt{\gamma}} < \rho < 1$, then for $k \geq 0$ the sequence $\{x^k\}$ generated by Algorithm 17 converges to the optimum linearly with the constant ρ and satisfies*

$$\|x^{k+1} - x^\star\| \leq \rho^{k+1}\left[\frac{2\sqrt{\Phi(x^0) - \Phi(x^\star)}}{\sqrt{\sigma_\phi}} + \frac{(2C_3 + 2\sqrt{2L}C_4 + \sqrt{2\sigma_\phi}C_4)\rho}{\sigma_\phi(\rho - \sqrt{1-\sqrt{\gamma}})\cdot\sqrt{1-\sqrt{\gamma}}}\right] \ , \quad (6.22)$$

with $C_3 = \frac{M\sqrt{\bar{m}}(3L_{max}dC_\alpha + \rho\sqrt{d}C_\beta)}{\rho\cdot 2^{n+1}}$ and $C_4 = \frac{\sqrt{2}}{2}\cdot\frac{M\sqrt{\bar{m}}C_\alpha}{2^{n+1}}$.

Proof: The proof follows directly from the proof of Theorem 6.2 by replacing Lemma 6.2, Lemma 6.3 and Lemma 6.4 by Lemma 6.5, Lemma 6.6 and Lemma 6.7. ■

6.4 Numerical Example

This section illustrates the theoretical findings of the chapter and demonstrates the performance of Algorithm 16 and Algorithm 17 for solving a distributed quadratic programming (QP) problem originating from the problem of regulating constrained distributed linear systems by model predictive control (MPC) in the form of Problem 3.2.

We use the same distributed MPC as in Section 5.4. The concatenation of the initial states of subsystem i and its neighbours is denoted by $\bar{x}_{\mathcal{N}_i}$. By eliminating all state variables distributed MPC problems of this class can be reformulated as a distributed QP of the form in Problem 6.4 with the local variables $x_i = u_i$ and the concatenations of the variables of subsystem i and its neighbours $x_{\mathcal{N}_i}$.

Problem 6.4.

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}} \quad & f(x) = \sum_{i=1}^{M} f_i(x_{\mathcal{N}_i}) = \sum_{i=1}^{M} x_{\mathcal{N}_i}^T H_i x_{\mathcal{N}_i} + \bar{x}_{\mathcal{N}_i}^T h_i^T x_{\mathcal{N}_i} \\ s.t. \quad & x_i \in \mathbb{C}_i \quad . \end{aligned}$$

The matrices are given by $H_i = \mathcal{B}_i^T \mathcal{Q}_i \mathcal{B}_i + \mathcal{R}_i$ and $h_i = \mathcal{A}_i^T \mathcal{Q}_i \mathcal{B}_i$, where $\mathcal{A}_i = \begin{bmatrix} A_{ii}^T & \cdots & A_{ii}^{N^T} \end{bmatrix}$,

$$\mathcal{Q}_i = \begin{bmatrix} I_N \otimes Q_{ii} & 0 \\ 0 & P_{ii} \end{bmatrix}, \quad \mathcal{B}_i = \begin{bmatrix} B_i & 0 & \cdots & 0 \\ A_{ii}B_i & B_i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{ii}^{N-1}B_i & A_{ii}^{N-2}B_i & \cdots & B_i \end{bmatrix},$$

and $\mathcal{R}_i = I_N \otimes R_i$ with $B_i = [B_{ij_1}, \cdots, B_{ij_{|\mathcal{N}_i|}}]$ and $R_i = blkdiag(R_{j_1 j_1}, \cdots, R_{j_{|\mathcal{N}_i|} j_{|\mathcal{N}_i|}})$, with $\mathcal{N}_i = \{j_1, \cdots, j_{|\mathcal{N}_i|}\}$. The constraint $\mathbb{C}_i = \mathbb{U}_i^N$ is a polytopic set. Note that for this example the matrix H_i is dense and positive definite, and vector h_i is dense.

Table 6.1 shows the parameters chosen in Algorithm 16 and Algorithm 17, including the constants of the convergence rate of the algorithms, i.e. $\gamma = \frac{\sigma_f}{L}$ and $\sqrt{1-\sqrt{\gamma}}$, the decrease rates of the quantization intervals ρ satisfying $1-\gamma \leq \rho \leq 1$ for Algorithm 16 and $\sqrt{1-\sqrt{\gamma}} \leq \rho \leq 1$ for Algorithm 17 and the minimum number of bits required for convergence n_{min}.

Fig. 6.2 shows the relationship between the number of bits n and the minimum initial quantization intervals C_α and C_β, which satisfy Assumption 6.4 for Problem 6.4. We see that the minimum number of bits required for convergence is equal to $n_{min} = 13$, and as the number of bits n increases, the required minimum C_α and C_β decrease.

Fig. 6.3 shows the performance of Algorithm 16 and Algorithm 17 for solving the distributed QP problem in Problem 6.4 originating from the distributed MPC problem. For Algorithm 16, n is set to 13 and 15, respectively, and the initial quantization intervals C_α and C_β are set to corresponding minimum values satisfying Assumption 6.4. For Algorithm 17, n is set to 19 and 23, and C_α and C_β to corresponding minimum values satisfying Assumption 6.5. In Fig. 6.3 we can observe that the proposed distributed algorithms with quantization converges to the global

Parameters	Algorithm 16	Algorithm 17
Rate constants	$1-\gamma = 0.8093$	$\sqrt{1-\sqrt{\gamma}} = 0.7505$
ρ	0.9333	0.7991
n_{min}	13	19

Table 6.1 – The parameters in Algorithm 16 and 17 for solving Problem 6.4.

optimum linearly and the performance is improved when the number of bits n is increased. Due to the acceleration step, Algorithm 17 converges faster than Algorithm 16. However, Algorithm 17 requires a larger number of bits n to guarantee the convergence.

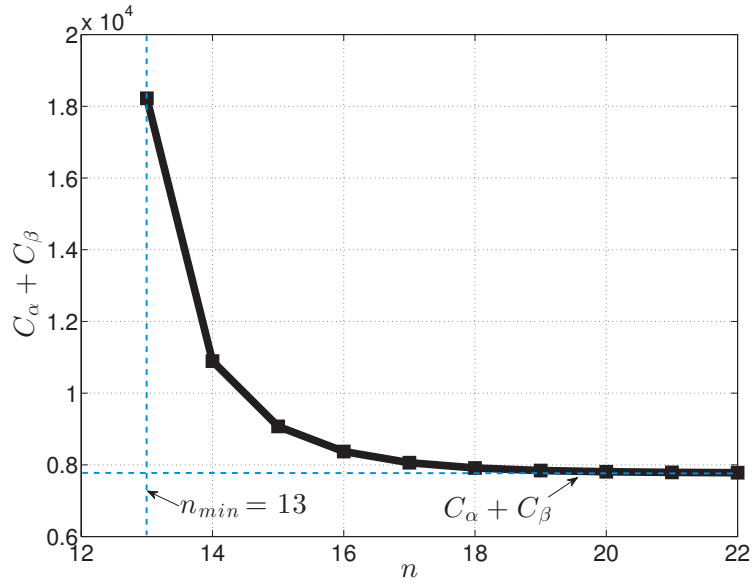


Figure 6.2 – Relationship between the number of bits n and the minimum initial quantization intervals C_α and C_β satisfying Assumption 6.4 for Problem 6.4.

6.5 Conclusion

In Chapter 6, we considered distributed optimization problems with limited communication rate and proposed two distributed optimization algorithms with an iteratively refining quantization based on the inexact proximal gradient method. It is shown that if the parameters of the quantizers satisfy certain conditions, then the quantization error decreases linearly and the convergence of the distributed algorithms is guaranteed. Fig. 6.4 illustrates the proposed distributed optimization algorithms with a iteratively refining quantization design.

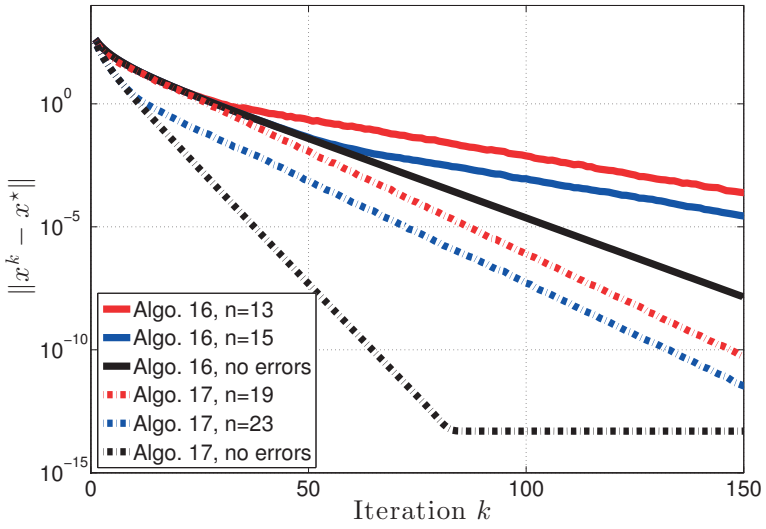


Figure 6.3 – Comparison of the performance of Algorithm 16 and 17 with different n and corresponding minimum C_α and C_β with the exact algorithms (no quantization errors) for Problem 6.4.

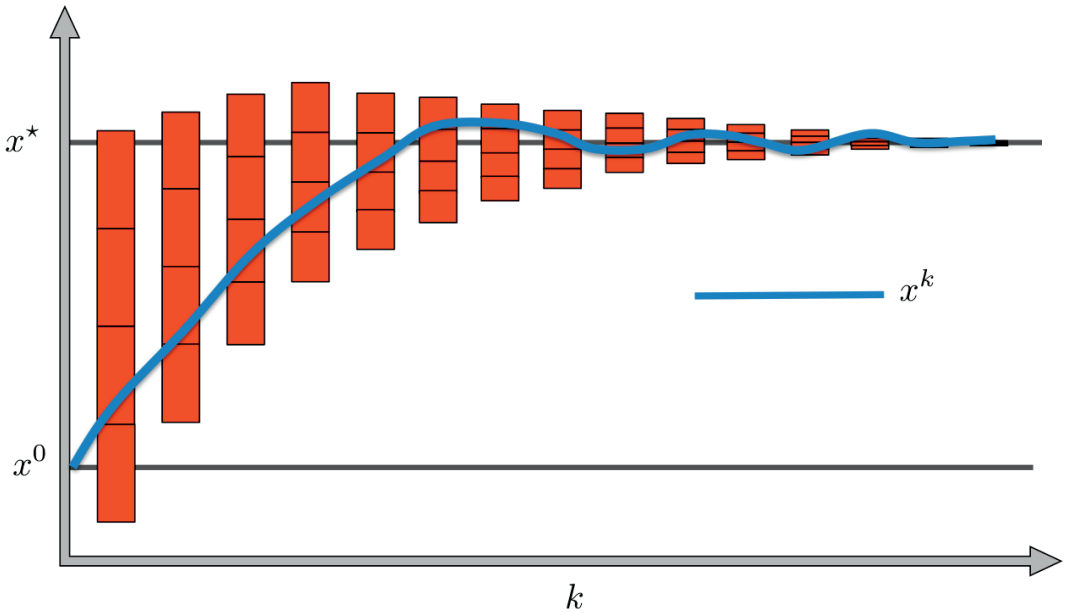


Figure 6.4 – Illustration of the proposed distributed optimization algorithms with a iteratively refining quantization design.

Quantization design for distributed optimization with time-varying parameters

7

The majority of the text and content in Chapter 7 has appeared in [66].

7.1 Introduction

In this chapter, we consider the problem of solving a sequence of distributed optimization problems parameterized by a time-varying parameter, which is central to many engineering problems, e.g. on-line resource allocation, distributed estimation and distributed optimal control problems. We develop distributed optimization methods considering the following two challenges: 1. solution of each problem in a distributed manner with only local communication, i.e. between neighbouring sub-systems and with limited communication bandwidth, where at each iteration only a limited number of bits can be transmitted; 2. optimization of the problems to a given accuracy sequentially and efficiently, i.e., to reduce the computation and communication required to achieve the desired level of accuracy.

We propose an optimization method with a progressive quantization scheme to solve the distributed optimization problems sequentially. The idea is to extend the progressive quantization scheme developed in Chapter 6 to a quantization design for parametric distributed optimization. By employing a warm-starting strategy, we leverage the solution at the previous time instance to improve the performance of the algorithm and show that there exists a trade-off between the accuracy and the number of iterations K. In particular, this chapter makes the following main contributions:

- We extend the progressive quantization design for distributed optimization in Chapter 6 to the problem of optimizing a sequence of distributed problems with time-varying parameters and present the conditions on the quantizers, which guarantee that for all steps the values exchanged in the network always fall inside the quantization intervals and the quantization errors decrease linearly.
- By employing a warm-starting strategy, we improve the convergence speed of the algorithm and present a relationship between the solution accuracy and the cost of computation and communication represented by the number of

iterations K. We show that for a given accuracy ϵ, there always exists a K guaranteeing that the sub-optimality of each solution in the sequence of distributed optimization problems is upper-bounded by ϵ.

- We demonstrate the proposed method for solving a distributed model predictive control problem by considering the initial state measurement at each sampling time as the varying parameters and compare the simulation results with the theoretical bound.

7.2 Preliminaries

7.2.1 Parametric distributed optimization problem

We consider a parametric distributed optimization problem on a network given in Problem 7.1, which is an extended problem of Problem 6.1 with a time-dependent parameter $\eta_i^t \in \Xi_i \subseteq \mathbb{R}^{n_{\eta_i}}$, for $i = 1, 2, \cdots, M$. We denote $\eta^t = [\eta_1^{t^T}, \cdots, \eta_M^{t^T}]$.

Problem 7.1.

$$\begin{aligned} \min_{x,\, x_{\mathcal{N}_i}} \quad & f(x, \eta^t) = \sum_{i=1}^{M} f_i(x_{\mathcal{N}_i}, \eta_i^t) \\ s.t. \quad & x_i \in \mathbb{C}_i \ , \ x_i = F_{ji} x_{\mathcal{N}_j} \ , \ j \in \mathcal{N}_i \ , \\ & x_{\mathcal{N}_i} = E_i x \ , \ i = 1, 2, \cdots, M \ . \end{aligned}$$

Assumption 7.1. *We assume that for all $\eta_i^t \in \Xi_i$ the global cost function $f(\cdot, \cdot)$ is strongly convex with respect to x with a convexity modulus σ_f.*

Assumption 7.2. *We assume that for all $\eta_i^t \in \Xi_i$ every local cost function $f_i(\cdot)$ has Lipschitz continuous gradient with a Lipschitz constant L_i, and denote L_{max} as the maximum Lipschitz constant of the local functions, i.e. $L_{max} := \max_{1 \le i \le M} L_i$.*

Remark 7.1. *If Assumption 7.2 holds, then the global cost function has a Lipschitz continuous gradient with a Lipschitz constant $L := \sum_{1 \le i \le M} L_i$.*

Assumption 7.3. *The local constraint $\mathbb{C}_i$ is a convex set, for $i = 1, \cdots, M$.*

Model predictive control is one application resulting in a parametric optimization problem, which generally satisfies Assumption 7.1, Assumption 7.2 and Assumption 7.3. This will be discussed in more detail in the example in Section 7.4.

7.2.2 Distributed optimization with limited communication

Algorithm 18 provides the distributed algorithm with the progressive quantization design proposed in Algorithm 16 for Problem 7.1 with a fixed parameter η^t. For every sub-system i, there are two uniform quantizers $Q_{\alpha,i}^{t,k}$ and $Q_{\beta,i}^{t,k}$ using the formulation introduced in (6.1) with a fixed number of bits n, changing quantization intervals $l_{\alpha,i}^{t,k}$ and $l_{\beta,i}^{t,k}$ and changing mid-values $\bar{x}_{\alpha,i}^{t,k}$ and $\bar{\nabla} f_{\beta,i}^k$ for transmitting $x_i^{t,k}$ and ∇f_i^k at every iteration k. At iteration k, the quantization intervals are set to be $l_{\alpha,i}^{t,k} = C_\alpha^t \kappa^k$ and $l_{\beta,i}^{t,k} = C_\beta^t \kappa^k$, and the mid-values are set to be the previous quantized values

$\bar{x}_{\alpha,i}^{t,k} = \hat{x}_i^{t,k-1}$ and $\bar{\nabla} f_{\beta,i}^k = \hat{\nabla} f_i^{k-1}$. The two parameters $C_\alpha^t = l_{\alpha,i}^{t,0}$ and $C_\beta^t = l_{\beta,i}^{t,0}$ denote the initial quantization intervals. Similar to Chapter 6, $\hat{\cdot}$ is used to denote a quantized value, e.g. $\hat{x}_i^{t,k} = Q_{\alpha,i}^{t,k}(x_i^{t,k})$ and $\tilde{\cdot}$ is used to denote a re-projected value, e.g. $\tilde{x}_{\mathcal{N}_i}^{t,k} = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}_{\mathcal{N}_i}^{t,k})$. The quantization errors are denoted by $\alpha_i^{t,k} = \hat{x}_i^{t,k} - x_i^{t,k}$ and $\beta_i^{t,k} = \hat{\nabla} f_i^k - \nabla f_i^k$.

Algorithm 18 Distributed algorithm with quantization refinement

Require: Initialize $\hat{x}_i^{t,-1} = x_i^0(\eta^t)$, $\hat{\nabla} f_i^{t,-1} = \nabla f_i(\text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(x_{\mathcal{N}_i}^0(\eta^t)))$, $(1-\gamma) < \kappa < 1$, $\tau < \frac{1}{L}$, the initial quantization intervals C_α^t and C_β^t and the number of iterations K.

for $k = 0, 1, 2, \cdots, K$ **do**

For sub-system $i = 1, 2, \cdots, M$ (in parallel):

1: Update the parameters of quantizer $Q_{\alpha,i}^{t,k}$: $l_{\alpha,i}^{t,k} = C_\alpha^t \kappa^k$ and $\bar{x}_{\alpha,i}^{t,k} = \hat{x}_i^{t,k-1}$

2: Quantize the state: $\hat{x}_i^{t,k} = Q_{\alpha,i}^{t,k}(x_i^{t,k}) = x_i^{t,k} + \alpha_i^{t,k}$.

3: Send $\hat{x}_i^{t,k}$ to all the neighbours of sub-system i

4: Compute the projection of $\hat{x}_{\mathcal{N}_i}^{t,k}$: $\tilde{x}_{\mathcal{N}_i}^{t,k} = \text{Proj}_{\mathbb{C}_{\mathcal{N}_i}}(\hat{x}_{\mathcal{N}_i}^{t,k})$

5: Compute $\nabla f_i^k = \nabla f_i(\tilde{x}_{\mathcal{N}_i}^{t,k})$

6: Update the parameters of quantizer $Q_{\beta,i}^{t,k}$: $l_{\beta,i}^{t,k} = C_\beta^t \kappa^k$ and $\bar{\nabla} f_{\beta,i}^k = \hat{\nabla} f_i^{k-1}$

7: Quantize the gradient: $\hat{\nabla} f_i^k = Q_{\beta,i}^{t,k}(\nabla f_i^k) = \nabla f_i^k + \beta_i^{t,k}$.

8: Send $\hat{\nabla} f_i^k$ to all the neighbours of sub-system i

9: Update the state: $x_i^{t,k+1} = \text{Proj}_{\mathbb{C}_i}(x_i^{t,k} - \tau \sum_{j \in \mathcal{N}_i} F_{ji} \hat{\nabla} f_j^k)$

end for

Assumption 7.4. *Consider the quantizers $Q_{\alpha,i}^{t,k}$ and $Q_{\beta,i}^{t,k}$ in Algorithm 18. We assume that the parameters of the quantizers, i.e. the number of bits n and the initial quantization intervals C_α^t and C_β^t satisfy*

$$a_1 \cdot \|x^0(\eta^t) - x^\star(\eta^t)\| + a_2 \frac{C_\alpha^t}{2^{n+1}} + a_3 \frac{C_\beta^t}{2^{n+1}} \leq \frac{C_\alpha^t}{2} \tag{7.1}$$

$$b_1 \cdot \|x^0(\eta^t) - x^\star(\eta^t)\| + b_2 \frac{C_\alpha^t}{2^{n+1}} + b_3 \frac{C_\beta^t}{2^{n+1}} \leq \frac{C_\beta^t}{2} \ , \tag{7.2}$$

where the constants are defined as

$$\begin{aligned}
a_1 =& \frac{(\kappa+1)}{\kappa} \ , \\
a_2 =& \frac{M\sqrt{\bar{m}}\kappa(\kappa+1)(dL_{\max} + \sqrt{L}) + M\sqrt{\bar{m}}L(\kappa+\gamma-1)(1-\gamma)}{L\kappa(\kappa+\gamma-1)(1-\gamma)} \ , \\
a_3 =& \frac{M\sqrt{d\bar{m}}(\kappa+1)}{L(\kappa+\gamma-1)(1-\gamma)} \ , \\
b_1 =& \frac{L_{\max}(\kappa+1)}{\kappa} \ , \\
b_2 =& \frac{L_{\max}M\sqrt{\bar{m}}\kappa(\kappa+1)(dL_{max} + \sqrt{L}) + L_{max}d\sqrt{\bar{m}}L(\kappa+1)(\kappa+\gamma-1)(1-\gamma)}{L\kappa(\kappa+\gamma-1)(1-\gamma)} \ ,
\end{aligned}$$

$$b_3 = \frac{L_{\max} M \sqrt{d\bar{m}} \kappa(\kappa+1) + L\sqrt{d\bar{m}}(\kappa+\gamma-1)(1-\gamma)}{L\kappa(\kappa+\gamma-1)(1-\gamma)} \ .$$

Remark 7.2. *The parameters of the quantizers n, C_α^t and C_β^t are all positive constants. Assumption 7.4 can always be satisfied by increasing n, C_α^t and C_β^t.*

Theorem 7.2. *[Theorem 6.2 with parameter η^t] For any $t \geq 0$, if Assumptions 7.1, 7.2 and 7.4 hold and $(1-\gamma) < \kappa < 1$, then for $0 \leq k \leq K$ the sequence $\{x^{t,k+1}\}$ generated by Algorithm 18 converges to the optimum linearly with the constant κ and satisfies*

$$\|x^{t,k+1} - x^\star(\eta^t)\| \leq \kappa^{k+1}\left[\|x^0(\eta^t) - x^\star(\eta^t)\| + \frac{(C_1^t + \sqrt{2L}C_2^t)\kappa}{L(\kappa+\gamma-1)(1-\gamma)}\right] \ .$$

with $C_1^t = \frac{M\sqrt{\bar{m}}(L_{max} d C_\alpha^t + \sqrt{d} C_\beta^t)}{2^{n+1}}$ *and* $C_2^t = \frac{\sqrt{2}}{2} \cdot \frac{M\sqrt{\bar{m}} C_\alpha^t}{2^{n+1}}$.

Proof. It follows directly from Theorem 6.2 by considering the parameter η^t as a constant in the optimization problem. □

Theorem 7.2 states that with the proposed quantization design, the linear convergence of the algorithm is preserved, but the constant of the convergence rate has to be enlarged from $1-\gamma$ to κ in order to compensate for the deficiencies arising from limited communication.

7.3 Parametric distributed optimization with limited communication

We extend Algorithm 18 to solve the parametric distributed optimization Problem 7.1. For parametric optimization problems with a slowly time-varying parameter, so-called warm-starting strategies initializing the solution at each time step with the solution obtained at the previous time-step have been observed to offer significant computational speedups [3]. One example is the solution of optimal control problems that are parameterized by the state measurement. The solution from the previous step offers a reasonable guess for the current solution, if the parameter, i.e. the state, does not change drastically from one time step to the next, and therefore reduces the number of iterations to optimality compared to starting from a fixed point, also known as cold-starting. In this section, we apply this warm-starting strategy to initialize the starting sequence at time t in Algorithm 19, i.e. $x^0(\eta^t) = x^K(\eta^{t-1})$. More importantly than improving practical performance, we employ the warm-starting strategy from a theoretical perspective to show that there exists a relationship between the number of iterations K and the accuracy ϵ of a suboptimal solution with respect to the optimal solution. For a given ϵ, we can always find a K guaranteeing that for all $t \geq 0$ the sub-optimal solution $x^K(\eta^t)$ satisfies the accuracy ϵ, i.e. $\|x^K(\eta^t) - x^\star(\eta^t)\| \leq \epsilon$. The distributed optimization algorithm with quantization refinement for the parametric optimization problem is presented in Algorithm 19.

Assumption 7.5. *We assume that the optimal solution satisfies*

$$\|x^\star(\eta^t) - x^\star(\eta^{t+1})\| \leq \rho \ , \tag{7.3}$$

for all $t \geq 0$.

Assumption 7.6. *We assume that at the first step* $t = 0$ *the initial solution of the algorithm is a sub-optimal solution satisfying*

$$\|x^0(\eta^0) - x^\star(\eta^0)\| \leq \epsilon \ . \tag{7.4}$$

Assumption 7.5 states a condition on the difference between the optimal solutions for time t and $t+1$. The difference is assumed to be upper-bounded by a constant for all $t \geq 0$. Assumption 7.6 shows a condition on the difference between the initial solution and optimal solution at each time t. The difference is required to be upper-bounded by a constant. The applications satisfying these two assumptions include distributed model predictive control problem and distributed moving horizon estimation problems by considering the state measurement at each sampling time as the time-varying parameter.

Remark 7.3. *A sub-optimal solution* $x^0(\eta^0)$ *at time* $t = 0$ *satisfying Assumption 7.6 can be computed off-line.*

Assumption 7.7. *We assume that the two parameters* C_α *and* C_β *in Algorithms 19 satisfy*

$$a_1 \cdot (\epsilon + \rho) + a_2 \frac{C_\alpha}{2^{n+1}} + a_3 \frac{C_\beta}{2^{n+1}} \leq \frac{C_\alpha}{2} \tag{7.5a}$$

$$b_1 \cdot (\epsilon + \rho) + b_2 \frac{C_\alpha}{2^{n+1}} + b_3 \frac{C_\beta}{2^{n+1}} \leq \frac{C_\beta}{2} \ . \tag{7.5b}$$

Remark 7.4. *The two conditions on the initial quantization intervals* C_α^t *and* C_β^t *in Assumption 7.7 do not vary with* t*, i.e., they are independent of the parameters* η^t*. Therefore, we can compute the initial intervals* C_α *and* C_β *satisfying (7.5) off-line and set* C_α^t *and* C_β^t *to the same values for all* $t \geq 0$.

Algorithm 19 Parametric distributed algorithm with quantization refinement

Require: Initial solution $x^0(\eta^0)$, the number of iterations K and the initial quantization intervals $C_\alpha^t = C_\alpha$ and $C_\beta^t = C_\beta$ for all $t \geq 0$.

for $t = 0, 1, 2, \cdots$ **do**

1: Solve Problem 7.1 with the parameter η^t by Algorithm 18 with the initial solution $x^0(\eta^t)$ and the number of iterations K.

2: $x^0(\eta^{t+1}) \leftarrow x^{K+1}(\eta^t)$ (warm-starting)

end for

Theorem 7.3. *For a given* $\epsilon > 0$*, if Assumptions 7.1, 7.2, 7.5, 7.6 and 7.7 hold,* $(1-\gamma) < \kappa < 1$ *and the number of iterations* K *satisfies*

$$K \geq \left\lceil \log_\kappa \frac{\epsilon(1-\kappa)}{\rho + \delta + (1-\kappa)(\epsilon + \delta)} \right\rceil - 1 \ , \tag{7.6}$$

with $\delta = \frac{\kappa(C_1 + \sqrt{2L}C_2)}{L(\kappa+\gamma-1)(1-\gamma)}$, $C_1 = \frac{M\sqrt{\bar{m}}(L_{max} d C_\alpha + \sqrt{d} C_\beta)}{2^{n+1}}$ *and* $C_2 = \frac{\sqrt{2}}{2} \cdot \frac{M\sqrt{\bar{m}} C_\alpha}{2^{n+1}}$*, then the sub-optimal solution* $x^{K+1}(\eta^t)$ *satisfies:*

$$\|x^{K+1}(\eta^t) - x^\star(\eta^t)\| \leq \epsilon \ , \tag{7.7}$$

for all $t \geq 0$.

Proof. We will prove Theorem 7.3 by induction.

- Base case: At $t = 0$, Assumption 7.6 and Assumption 7.7 imply that Assumption 7.4 holds. Then all assumptions required by Theorem 7.2 are satisfied and it follows that $\|x^{K+1}(\eta^0) - x^\star(\eta^0)\| \leq \kappa^{K+1}(\|x^0(\eta^0) - x^\star(\eta^0)\| + \delta) \leq \kappa^{K+1}(\epsilon + \delta)$. Using the condition in (7.6), we get $\kappa^{K+1}(\epsilon + \delta) \leq \epsilon$. Hence, it holds that $\|x^{K+1}(\eta^0) - x^\star(\eta^0)\| \leq \epsilon$.

- Induction step: Let $g \geq 0$ be given and suppose that $\|x^{K+1}(\eta^t) - x^\star(\eta^t)\| \leq \epsilon$ for $t \leq g$. We will prove that $\|x^{K+1}(\eta^{g+1}) - x^\star(\eta^{g+1})\| \leq \epsilon$. By the warm-starting step in Step 3 in Algorithm 19, we know

$$\begin{aligned}\|x^0(\eta^{g+1}) - x^\star(\eta^{g+1})\| =& \|x^{K+1}(\eta^g) - x^\star(\eta^{g+1})\| \\ \leq& \|x^{K+1}(\eta^g) - x^\star(\eta^g)\| + \|x^\star(\eta^g) - x^\star(\eta^{g+1})\| \ .\end{aligned}$$

  By the assumption of induction and Assumption 7.5 , we obtain

$$\|x^0(\eta^{g+1}) - x^\star(\eta^{g+1})\| \leq \epsilon + \rho \ .$$

  Then Assumption 7.7 implies that Assumption 7.4 holds for $g + 1$. It follows from Theorem 7.2 that

$$\|x^{K+1}(\eta^{g+1}) - x^\star(\eta^{g+1})\| \leq \kappa^{K+1}(\|x^0(\eta^{g+1}) - x^\star(\eta^{g+1})\| + \delta) \ .$$

  By the warm-starting step in Step 3 in Algorithm 19, the above is upper-bounded by

$$\begin{aligned}\leq& \kappa^{K+1}(\|x^{K+1}(\eta^g) - x^\star(\eta^{g+1})\| + \delta) \\ \leq& \kappa^{K+1}(\|x^{K+1}(\eta^g) - x^\star(\eta^g)\| + \|x^\star(\eta^g) - x^\star(\eta^{g+1})\| + \delta) \ ,\end{aligned}$$

  Assumption 7.5 implies

$$\leq \kappa^{K+1}(\|x^{K+1}(\eta^g) - x^\star(\eta^g)\| + \rho + \delta) \ .$$

  Again by the warm-starting step, Assumption 7.7 implies Assumption 7.4. It follows from Theorem 7.2 that the above is upper-bounded by

$$\begin{aligned}\leq& \kappa^{K+1}(\rho + \delta) + (\kappa^{K+1})^2 \cdot (\|x^0(\eta^g) - x^\star(\eta^g)\| + \delta) \\ \leq& \kappa^{K+1}(\rho + \delta) + (\kappa^{K+1})^2 \cdot (\|x^{K+1}(\eta^{g-1}) - x^\star(\eta^{g-1})\| + \rho + \delta) \ .\end{aligned}$$

  Sequentially, we get that the above is upper-bounded by

$$\leq \sum_{p=1}^{g+1} (\kappa^{K+1})^p \cdot (\rho + \delta) + (\kappa^{K+1})^{g+2} \cdot (\epsilon + \delta) \ .$$

  By Assumption 7.6 and the property of geometric series, we have

$$\leq (\rho + \delta) \cdot \kappa^{K+1} \cdot \frac{1 - (\kappa^{K+1})^{g+1}}{1 - \kappa^{K+1}} + (\epsilon + \delta) \cdot (\kappa^{K+1})^{g+2} \ .$$

Using the fact that $0 < \kappa < 1$, we get

$$\|x^{K+1}(\eta^{g+1}) - x^\star(\eta^{g+1})\| \leq \frac{\rho+\delta}{1-\kappa} \cdot \kappa^{K+1} + (\epsilon+\delta) \cdot \kappa^{K+1} \ .$$

Note that the inequality above holds for all $t \geq 0$. By the condition in (7.6), we obtain that

$$\|x^{K+1}(\eta^{g+1}) - x^\star(\eta^{g+1})\| \leq \epsilon \ .$$

We conclude that by the principle of induction for all $t \geq 0$ the solution $x^{K+1}(\eta^t)$ satisfies $\|x^{K+1}(\eta^t) - x^\star(\eta^t)\| \leq \epsilon$. □

For a given accuracy ϵ and parameter variation δ, Theorem 7.3 provides a lower bound on the number of iterations ensuring that for all $t \geq 0$ the sub-optimal solution provided by Algorithm 19 satisfies the accuracy ϵ.

Remark 7.5. *We add a brief discussion about why warm-starting is necessary for Theorem 7.3. With both warm- and cold-starting, the algorithm converges to the optimum at each step t, as the number of iterations k goes to infinity. However, warm-starting is required in order to derive an off-line condition on the number of iterations K to achieve a given fixed accuracy at all time steps. Without the warm-starting strategy, the number of iterations to achieve a given accuracy varies with the parameters and would need to be determined on-line.*

Remark 7.6. *By using the results in Theorem 7.3, we can also compute the best accuracy ϵ for a given number of iterations K, i.e., for a given communication data-rate and a given number of bits per iteration.*

7.4 Numerical Example

This section illustrates the theoretical findings of the chapter and demonstrates the performance of Algorithm 19. We consider a parametric distributed quadratic programming (QP) problem originating from the same problem of regulating constrained distributed linear systems by model predictive control (MPC) introduced in Section 6.4. In addition, the initial state $\bar{z}_i^t$ is considered to be the time-varying parameter.

Problem 7.4.

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}} \ f(x, \bar{z}^t) &= \sum_{i=1}^{M} f_i(x_{\mathcal{N}_i}, \bar{z}^t_{\mathcal{N}_i}) \\ &= \sum_{i=1}^{M} x_{\mathcal{N}_i}^T H_i x_{\mathcal{N}_i} + \bar{z}_{\mathcal{N}_i}^{t^T} h_i x_{\mathcal{N}_i} \\ s.t. \quad & x_i \in \mathbb{C}_i \quad . \end{aligned}$$

The matrices H_i and h_i and the constraint $\mathbb{C}_i$ can be found in Section 6.4. The parameter appears in the linear term $\bar{z}_{\mathcal{N}_i}^{t^T} h_i x_{\mathcal{N}_i}$.

For Problem 7.4, the constants in Algorithm 19 are $\gamma = \frac{\sigma_f}{L} = 0.1027$, the decrease rates of the quantization intervals $1 - \gamma \leq \kappa = 0.9692$, and the minimum number of bits required for convergence, i.e. the conditions in (6.10), $n_{min} = 13$.

In the simulation Fig. 7.1, we set the number of steps to $t = 50$ and the number of iterations K to 2, 10 and 30. The parameter $\bar{z}^t$ is randomly generated and satisfies $\|\bar{z}^t - \bar{z}^{t+1}\| \leq 3$. Fig. 7.1 shows the accuracy achieved by Algorithm 19 and Algorithm 19 without warm-starting strategy, i.e. setting Step 2 in Algorithm 19 to $x^0(\eta^{t+1}) = 0$ (cold-starting) for all $t \geq 0$. The results show that warm-starting achieves significantly better accuracy for the same number of iterations.

In Fig. 7.2, we compute the average accuracy achieved by Algorithm 19 over all steps $t_{max} = 50$ in Fig. 7.1 and calculate the corresponding number of iterations K satisfying the bound in Theorem 7.3. Note that the parameter ρ in Assumption **??** is approximated by randomly sampling 500 initial states satisfying $\|\bar{z}^v - \bar{z}^{v+1}\| \leq 3$, for $1 \leq v \leq 500$, computing the largest $\rho = \max_{1 \leq v \leq 500}\{\|x^\star(\bar{z}^v) - x^\star(\bar{z}^{v+1})\|\}$. We observe that the bound is relatively loose, when the accuracy requirement ϵ is larger than $\epsilon > 10^{-3}$. It gets tighter as the accuracy ϵ decreases below $\epsilon < 10^{-3}$.

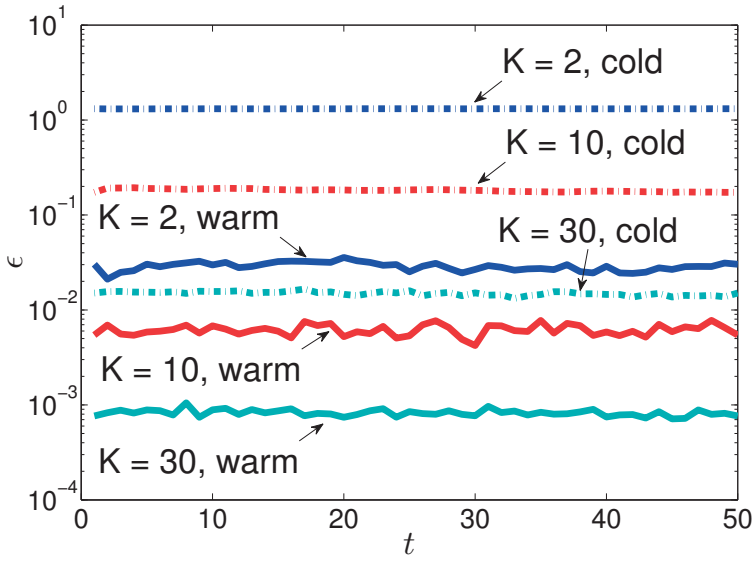


Figure 7.1 – Comparison of the accuracy achieved by Algorithm 19 and Algorithm 19 with a modified Step 2, i.e. $x^0(\eta^{t+1}) = 0$ (cold-starting), for solving Problem 7.4 for different numbers of iterations K.

7.5 Conclusion

In this chapter, we proposed an on-line algorithm for solving distributed optimization problems with time-varying parameters under two communication constraints, i.e. only neighbour-to-neighbour communication and a limited communication data-rate exchanged and show that there exists a trade-off between the number of iterations (communication data-rate) for solving the problem with the parameter at each time step and the accuracy achieved by the algorithm. We derived a lower-bound on the number of iterations K, which guarantees that the sub-optimal solution given by the algorithm at each time step satisfies a certain accuracy. We demonstrated the proposed method and the theoretical findings for solving a distributed model predictive control problem by considering the state measurement at each sampling time as

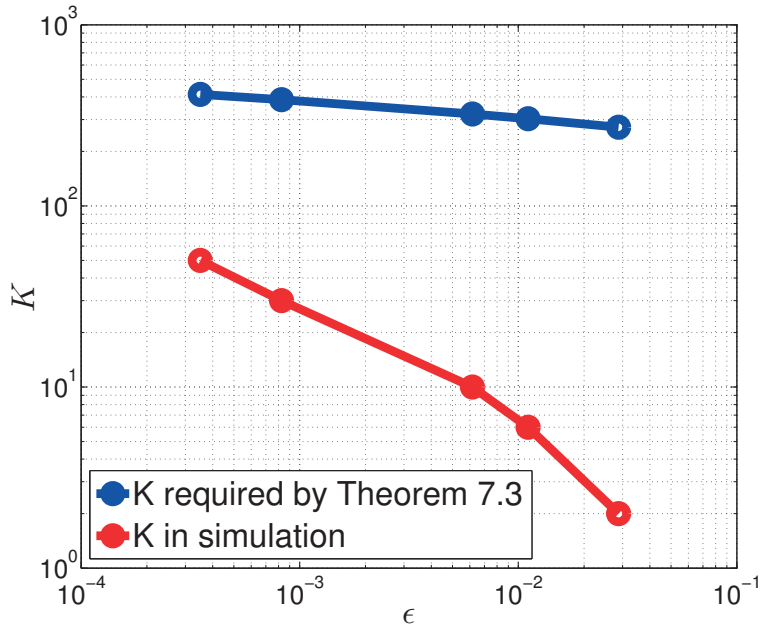


Figure 7.2 – Comparison of the number of iterations K required by Theorem 7.3 and obtained in simulation for solving Problem 7.4 for the same accuracy ϵ.

the time-varying parameter. Fig. 7.3 illustrates the proposed distributed algorithms with a iteratively refining quantization design for a distributed optimisation problem with time-varying parameters.

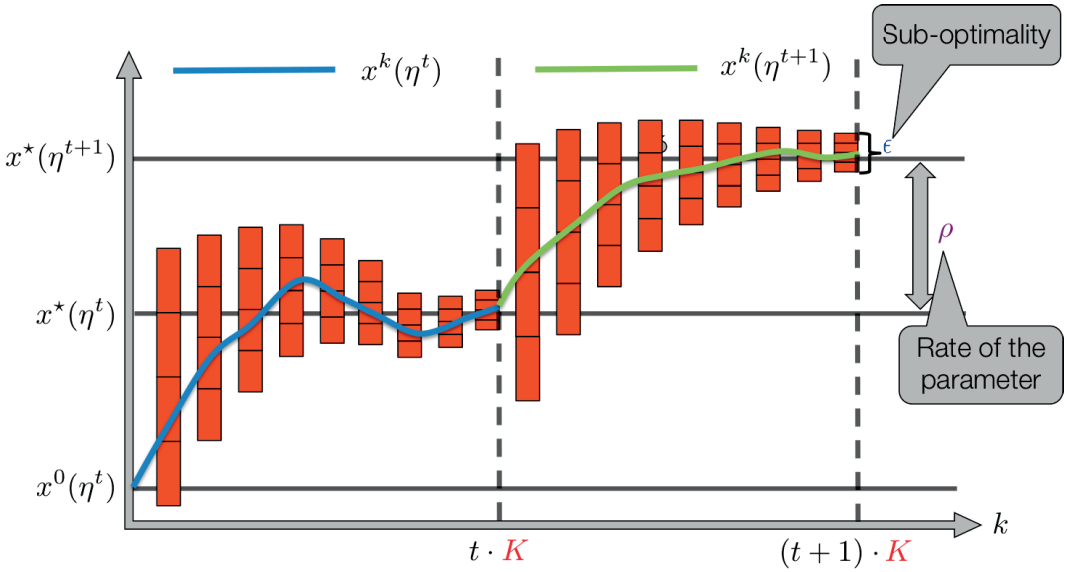


Figure 7.3 – Illustration of the proposed distributed algorithms with a iteratively refining quantization design for a distributed optimisation problem with time-varying parameters.

8 Conclusion and Outlook

In this work, we have focused on splitting methods, including their accelerated and inexact variants, and applied them to MPC problems and distributed optimization problems. The main contribution and future work are summarized in the following.

Splitting methods for fast MPC

Summary: In Chapter 4, we studied the fast alternating minimization algorithm and proposed efficient implementations for solving MPC problems with polytopic and second-order cone constraints. We derived complexity bounds on the number of iterations for both dual and primal variables, which are of particular relevance in the context of real-time MPC to bound the required online computation time, and further discussed the computation of the complexity bounds. For MPC problems with polyhedral and ellipsoidal constraints, we provided an off-line pre-conditioning method to further improve the convergence speed of FAMA by decreasing the complexity upper-bounds and enlarging the step-size of the algorithm.

Future research topics include:

- **Preconditioning techniques:** Splitting methods belong to the family of first-order methods. In general, first-order methods offer simple iteration schemes, but are very sensitive to the geometry data of an optimization problem, i.e., the Lipschitz continuity of the objective function and the shape of the constraints. One important approach to improve the performance of the algorithms is to use preconditioning techniques. In future work, an important problem to be addressed is to develop preconditioning methods using the complexity bounds for problems with general convex conic constraints, e.g. polyhedron, second-order cone and positive semi-definite cone.

- **Real-time MPC:**

  Complexity upper-bounds are of interest in the context of real-time MPC problems, as the bound provides a means to certify the sub-optimality of the solution resulting from the algorithms running for a fixed number of iterations. In order to use this sub-optimal solution for control, future work is to

consider the optimization algorithm as a dynamical system and apply system and control theory to analyse the connection to the physical system.

Splitting methods for distributed optimization

Summary: In Chapter 5, 6 and 7, we proposed inexact splitting optimization algorithms and utilized them to solve distributed optimization problems in the presence of computation and communication errors. In Chapter 5, the inexact alternating minimization algorithm, as well as its accelerated variant, was proposed to solve distributed optimization problems, allowing for local computation. In Chapter 6, we considered distributed optimization problems with limited communication rate and proposed two distributed optimization algorithms with an iteratively refining quantization based on the inexact proximal gradient method. It is shown that if the parameters of the quantizers satisfy certain conditions, then the quantization error decreases linearly and the convergence of the distributed algorithms is guaranteed. In Chapter 7, we extended the methods to distributed optimization problems with time-varying parameters, and show the trade-off between the accuracy of the sub-optimal solution given by the algorithm and the number of iterations for each time-step in the sequential realization of the parameter.

Future research topics include:

- **Complexity bound for distributed optimization:** An important open problem is to derive complexity upper-bounds for distributed algorithms and utilize this bound to study the questions: how does the topology of the network of a distributed optimization problem affect the convergence behavior of a distributed algorithm, and knowing these properties how could we design an efficient algorithm, as well as redesigning the network, to improve the convergence behavior? In a real-time framework, one open question is that distributed MPC problems need to be solved within limited computation time and communication source. Assuming that the computation time and communication only allow for implementing the algorithms for a fixed number of iterations, then how can we specify the sub-optimality of the solution by the complexity bound and use this sub-optimal solution for the control task?

- **Distributed optimization with synchronous updates:** The second future research topic is to develop a distributed algorithm with asynchronous updates. Most available distributed algorithms are synchronous and assume that computations at all nodes are performed simultaneously according to a global clock. However, compared to asynchronous algorithms, synchronous algorithms have drawbacks and are less flexible [67]. For asynchronous algorithms, each processor does not need to communicate to each other processor at each time instance. The processors may perform computations without having to wait until they receive the messages that have been transmitted to them. Such algorithms can relieve communication overloads and will not be significantly affected by either communication delays or the differences in computation time between different processors.

- **Distributed optimization with random errors:** An interesting field of research is to conduct research on stochastic optimization methods to develop

distributed algorithms admitting random errors in each iteration. In Chapter 5, we proposed an inexact fast alternating direction minimization algorithm, which allows inexact updates in each iteration. However, the condition on the error sequence for convergence can be conservative. The error sequence needs to converge to zero at a certain rate. It would be relevant to extend this result to the case that the error sequence does not necessarily converge but satisfies a certain distribution, in order to show probabilistic convergence of the algorithm by using stochastic splitting methods. This sacrifices the deterministic convergence of the algorithm, but relaxes the condition on the errors. Considering the communication schemes in Chapter 6, one direct extension is to show the required conditions on the quantization design for this case, or to develop a new design procedure, that guarantees a probabilistic convergence and allows for more general communication errors.

Bibliography

[1] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences, L. Magni, D. M. Raimondo, and F. Allgower, Eds. Springer Berlin Heidelberg, 2009, vol. 384, pp. 345–369.

[2] M. Zeilinger, C. Jones, and M. Morari, "Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization," *IEEE Trans. on Autom. Control*, vol. 56, pp. 1524–1534, 2011.

[3] R. Richter, C. N. Jones, and M. Morari, "Computational complexity certification for real-time MPC with input constraints based on the fast gradient method," *IEEE Transactions on Automatic Control*, vol. 57(6), pp. 1391–1403, 2012.

[4] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," in *Soviet Mathematics Doklady*, vol. 27, 1983, pp. 372–376.

[5] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. on Control Systems Techn.*, vol. 18, pp. 267–278, 2010.

[6] A. Domahidi, A. U. Zgraggen, M. N. Zeilinger, M. Morari, and C. N. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *Proc. of the 51st IEEE Conf. on Decision and Control*, 2012, pp. 668–674.

[7] M. Kogel and R. Findeisen, "Fast predictive control of linear systems combining nesterov's gradient method and the method of multipliers," in *Proc. of the 50th IEEE Conf. on Decision and Control*, 2011, pp. 501–506.

[8] P. Giselsson, M. D. Doan, T. Keviczky, B. D. Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, pp. 829–833, 2013.

[9] H. I. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit mpc," *International Journal of Robust and Nonlinear Control*, vol. 18, pp. 816–830, 2008.

[10] R. A. Bartlett and L. T. Biegler, "QPSchur: a dual, active-set, schur-complement method for large-scale and structured convex quadratic programming," *Optimization and Engineering*, vol. 7, pp. 5–32, 2006.

[11] T. Goldstein, B. O'Donoghue, and S. Setzer, "Fast alternating direction optimization methods," *CAM report*, pp. 12–35, 2012.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 2011.

[13] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, ser. Springer Optimization and Its Applications. Springer New York, 2011, pp. 185–212.

[14] C. Conte, N. R. Voellmy, M. N. Zeilinger, M. Morari, and C. N. Jones, "Distributed synthesis and control of constrained linear systems," in *American Control Conference*, 2012, pp. 6017–6022.

[15] W. B. Dunbar, "Distributed receding horizon control of dynamically coupled non-linear systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 7, pp. 1249–1263, 2007.

[16] M. Farina and R. Scattolini, "Distributed non-cooperative MPC with neighbor-to-neighbor communication," in *18th World Congress of the International Federation of Automatic Control*, 2011, pp. 404–409.

[17] D. Westwick, *Power plants and power systems control 2006*. Elsevier, 2007.

[18] A. Venkat, I. Hiskens, J. Rawlings, and S. Wright, "Distributed mpc strategies with application to power system automatic generation control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192–1206, 2008.

[19] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, Massachusetts, 1997.

[20] I. Necoara and V. Nedelcu, "Rate analysis of inexact dual first order methods: Application to distributed MPC for network systems," *arXiv:1302.3129 [math]*, Feb. 2013, arXiv: 1302.3129.

[21] Q. T. Dinh, I. Necoara, and M. Diehl, "Fast inexact decomposition algorithms for large-scale separable convex optimization," *Optimization*, in press, 2015.

[22] M. Schmidt, N. L. Roux, and F. Bach, "Convergence rates of inexact proximal-gradient methods for convex optimization," in *25th Annual Conference on Neural Information Processing Systems*, 2011, pp. 6819–6824.

[23] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1990.

[24] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex analysis and optimization*. Athena Scientific Belmont, 2003.

[25] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[26] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011.

[27] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms II*. Springer Berlin Heidelberg, 1993.

[28] A. Beck and M. Teboulle, "A fast iterative shrinkage thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, pp. 183–202, 2009.

[29] P. Tseng, "Applications of a splitting algorithm to decomposition in convex programming and variational inequalities," *SIAM Journal on Control and Optimization*, vol. 29, pp. 119–138, 1991.

[30] V. Acary, O. Bonnefon, and B. Brogliato, *Nonsmooth Modeling and Simulation for Switched Circuits*, ser. Lecture Notes in Electrical Engineering. Springer Netherlands, 2011, vol. 69.

[31] R. Glowinski and A. Marrocco, "Sur l'approximation par elements nis dordre un, et la resolution par penalisation-dualite dune classe de problemes de dirichlet nonlineaires," *Revue Francaise d'Automatique, Informatique, et Recherche Operationelle*, vol. R24176, 1975.

[32] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers: Quadratic problems," *IEEE Trans. on Autom. Control*, vol. 60, pp. 644–658, 2015.

[33] J. M. Maciejowski, *Predictive control: with constraints*. Pearson Education, 2002.

[34] J. B. Rawlings and D. Q. Mayne, *Model predictive control: theory and design*. Nob Hill Pub., 2009.

[35] R. Scattolini, "Architectures for distributed and hierarchical model predictive control – a review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[36] C. Conte, T. Summers, M. Zeilinger, M. Morari, and C. Jones, "Computational aspects of distributed optimization in model predictive control," in *51th IEEE Conference on Decision and Control*, 2012, pp. 6819–6824.

[37] Y. Pu, M. N. Zeilinger, and C. N. Jones, "Fast alternating minimization algorithm for model predictive control," in *19th World Congress of the International Federation of Automatic Control*, 2014.

[38] ——, "Complexity certication of the fast alternating minimization algorithm for linear model predictive control," *IEEE Transactions on Automatic Control*, accepted for publication in March 2016. [Online]. Available: http://infoscience.epfl.ch/record/207029/files/FAMAMPC_jrnl.pdf?version=1

[39] B. O'Donoghue, G. Stathopoulos, and S. Boyd, "A splitting method for optimal control," *IEEE Trans. on Control Systems Techn.*, p. 1, 2013.

[40] B. He and X. Yuan, "On the $o(1/n)$ convergence rate of the douglas–rachford alternating direction method," *SIAM Journal on Numerical Analysis*, vol. 50, pp. 700–709, 2012.

[41] P. Giselsson and S. Boyd, "Preconditioning in fast dual gradient methods," in *IEEE 53rd Conf. on Decision and Control*, 2014, pp. 5040–5045.

[42] A. Bemporad and M. Panaglotis, "Simple and certifiable quadratic programming algorithms for embedded linear model predictive control," in *4th IFAC Nonlinear Model Predictive Control Conf.*, 2012, pp. 14–20.

[43] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for embedded linear model predictive control," *IEEE Trans. on Automatic Control*, vol. 59, pp. 18–33, 2014.

[44] J. Demmel, J. Nie, and V. Powers, "Representations of positive polynomials on noncompact semialgebraic sets via KKT ideals," *Journal of Pure and Applied Algebra*, vol. 209, pp. 189–200, 2007.

[45] G. Calafiore and M. C. Campi, "Uncertain convex programs: randomized solutions and confidence levels," *Mathematical Programming*, pp. 25–46, 2005.

[46] P. Giselsson, "Optimal preconditioning and iteration complexity bounds for gradient-based optimization in model predictive control," in *American Control Conf.*, 2013, pp. 358–364.

[47] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, ser. Studies in Applied Mathematics. Philadelphia, PA: SIAM, Jun. 1994, vol. 15.

[48] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conf. on Robotics and Automation*, 2011, pp. 2520–2525.

[49] Y. Pu, M. Zeilinger, and C. N. Jones, "Inexact fast alternating minimization algorithm for distributed model predictive control," in *53th IEEE Conference on Decision and Control*, 2014, pp. 5915–5921.

[50] ——, "Inexact alternating minimization algorithm for distributed optimization with an application to distributed mpc," submitted in Mar. 2016.

[51] P. Giselsson, "Execution time certification for gradient-based optimization in model predictive control," in *51th IEEE Conference on Decision and Control*, Dec. 2012, pp. 3165–3170.

[52] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*. Dover Publications, Incorporated, 1974.

[53] Y. Pu, M. Zeilinger, and C. N. Jones, "Quantization design for unconstrained distributed optimization," in *American Control Conference*, 2015.

[54] ——, "Quantization design for distributed optimization,," *IEEE Transactions on Automatic Control*, accepted for publication in March 2016. [Online]. Available: http://infoscience.epfl.ch/record/207085/files/pu_quantization.pdf?version=1

[55] T. Erseghe, "Distributed optimal power flow using admm," *IEEE Transactions on Power Systems*, vol. 29, pp. 2370–2380, 2014.

[56] R. Carli and G. Notarstefano, "Distributed partition-based optimization via dual decomposition," in *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*, 2013, pp. 2979–2984.

[57] R. Carli, F. Fagnani, P. Frasca, T. Taylor, and R. Zampieri, "Average consensus on networks with transmission noise or quantization," in *Proceedings of European Control Conference*, 2007.

[58] A. Kashyap, T. Basar, and R. Srikant, "Quantized consensus," *Automatica*, vol. 43, pp. 1192–1203, 2007.

[59] D. Thanou, E. Kokiopoulou, Y. Pu, and P. Frossard, "Distributed average consensus with quantization refinement," *IEEE Transactions on Signal Processing*, vol. 61, pp. 194–205, 2013.

[60] T. Li, M. Fu, L. Xie, and J.-F. Zhang, "Distributed consensus with limited communication data rate," *IEEE Transactions on Automatic Control*, vol. 56, pp. 279–292, 2011.

[61] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. Tsitsiklis, "On distributed averaging algorithms and quantization effects," *IEEE Transactions on Automatic Control*, vol. 54, pp. 2506–2517, 2009.

[62] ——, "Distributed subgradient methods and quantization effects," in *47th IEEE Conference on Decision and Control*, 2008, pp. 4177–4184.

[63] M. Rabbat and R. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 798–808, 2005.

[64] M. El Chamie, L. J., and T. Basar, "Design and analysis of distributed averaging with quantized communication," in *Proc. of the 53st IEEE Conf. on Decision and Control*, 2014, pp. 3860–3865.

[65] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

[66] Y. Pu, M. N. Zeilinger, and C. N. Jones, "Quantization design for distributed optimization with time-varying parameters," in *54th IEEE Conference on Decision and Control*, 2015, pp. 2037–2042.

[67] H. T. Kung, "Synchronized and asynchronous parallel algorithms for multiprocessors," pp. 153–200, 1976.

Ye Pu

Address EPFL-STI-IGM-LA Station 9
Lausanne 1015, Switzerland

website http://people.epfl.ch/y.pu
Email y.pu@epfl.ch

Education

02.2012 - present PhD in progress, School of Electrical Engineering
École Polytechnique Fédéral de Lausanne, Lausanne, Switzerland

04.2009 - 06.2011 Master of Science, Electrical Engineering
Technische Universität Berlin, Berlin, Germany

09.2004 - 07.2008 Bachelor of Science, Electronic, Information and Electrical Engineering
Shanghai Jiao Tong University, Shanghai, China

Research experience

02.2012- present Automatic Control Lab, EPFL
Doctoral assistant, supervised by Prof. Colin Jones and Prof. Melanie Zeilinger

02.2014- 04.2014 Hybrid System Lab, University of California at Berkeley
Visiting Researcher, supervised by Prof. Melanie Zeilinger and Prof. Claire Tomlin

09.2011- 01.2012 Signal Processing Lab 4, EPFL
Intern, supervised by Prof. Pascal Frossard

01.2011 - 06.2011 Control Systems Group, TU Berlin
Master thesis, supervised by Dr. Naim Bajcinca and Prof. Jörg Raisch

02.2010 - 03.2011 Medical Technology Division Group
Fraunhofer Institute for Production Systems and Design Technology, Berlin
Research assistant, supervised by Dr. Christian Winne and Prof. Erwin Keeve

Honors

Excellent Scholarship of Shanghai Jiao Tong University	2005, 2006, 2007
Excellent Graduate of Shanghai Jiao Tong University	2008
Swiss National Science Foundation - Early Postdoc. Mobility Fellowship	09.2016 - 03.2018

Languages

Chinese	Native
English	Full professional proficiency
German	Professional working proficiency

Publications

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Distributed Optimization, March 2016, accepted for publication in IEEE Transactions on Automatic Control. `http://infoscience.epfl.ch/record/207085/files/pu_quantization.pdf?version=1`
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Complexity Certification of the Fast Alternating Minimization Algorithm for Linear Model Predictive Control, March 2016, accepted for publication in IEEE Transactions on Automatic Control. `http://infoscience.epfl.ch/record/207029/files/FAMAMPC_jrnl.pdf?version=1`
- F. F. C. Rego, **Y. Pu**, A. Alessandretti, A. P. Aguiar and C. N. Jones. Design of a Distributed Quantized Luenberger Filter for Bounded Noise. American Control Conference, 2016.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Distributed Optimization with time-varying parameters. 54rd IEEE Conference on Decision and Control, Osaka, 2015.
- F. F. C. Rego, **Y. Pu**, A. P. Aguiar and C. N. Jones. A Consensus Algorithm for Networks with Process Noise and Quantization Error. 53rd Annual Allerton Conference 2015.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Quantization Design for Unconstrained Distributed Optimization. American Control Conference, Chicago, 2015.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Fast Alternating Minimization Algorithm for Model Predictive Control. 19th IFAC World Congress, Cape Town, 8-24, 2014.
- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Inexact Fast Alternating Minimization Algorithm for Distributed Model Predictive Control. 53rd IEEE Conference on Decision and Control, Los Angeles, 2014, December 15-17, 2014.
- G. Stathopoulos, A. Szücs, **Y. Pu** and C. N. Jones. Splitting methods in control. 13th European Control Conference, Strasbourg, June 24-27 2014.
- M.N. Zeilinger, **Y. Pu**, S. Riverso, G. Ferrati-Trecate, C. N. Jones. Plug and Play Distributed Model Predictive Control based on Distributed Invariance and Optimization. 52rd IEEE Conference on Decision and Control, Florence, 2013
- D. Thanou, E. Kokiopoulou, **Y. Pu** and P. Frossard. Distributed Average Consensus with Quantization Refinement. Transactions on Signal Processing, p. 194-205, 2013.
- N. Bajcinca and **Y. Pu** A Symbolic Approach to Decentralized Supervisory Control of Hybrid Systems. accepted to 51st Annual Allerton Conference 2013

Preprints

- **Y. Pu**, M. N. Zeilinger and C. N. Jones. Inexact Alternating Minimization Algorithm for Distributed Optimization with an Application to Distributed MPC, March. 2016, submitted to IEEE Transactions on Automatic Control
- L. Ferranti, **Y. Pu**, C. N. Jones, and T. Keviczky. Asynchronous Splitting Design for Model Predictive Control. submitted to 55th IEEE Conference on Decision and Control, 2016
- G. Stathopoulos, H. Shukla, A. Szücs, **Y. Pu** and C. N. Jones. Operator splitting methods in control. Sep. 2015 submitted to Foundations and Trends in Systems and Control.